LNCS 3696

Włodzisław Duch
Janusz Kacprzyk
Erkki Oja
Sławomir Zadrożny (Eds.)

Artificial Neural Networks: Biological Inspirations – ICANN 2005

15th International Conference
Warsaw, Poland, September 2005
Proceedings, Part I

1 Part I

Springer

Lecture Notes in Computer Science 3696

Włodzisław Duch   Janusz Kacprzyk
Erkki Oja   Sławomir Zadrożny (Eds.)

Artificial Neural Networks: Biological Inspirations – ICANN 2005

15th International Conference
Warsaw, Poland, September 11-15, 2005
Proceedings, Part I

Springer

Volume Editors

Włodzisław Duch
Nicolaus Copernicus University
Department of Informatics
ul. Grudziądzka 5, 87-100 Toruń, Poland
and
Nanyang Technological University
School of Computer Engineering
Division of Computer Science
Blk N4-02c-114, Nanyang Avenue, Singapore, 639798
E-mail: wduch@phys.uni.torun.pl

Janusz Kacprzyk
Sławomir Zadrożny
Polish Academy of Sciences
Systems Research Institute
ul. Newelska 6, 01–447 Warsaw, Poland
E-mail: {kacprzyk,zadrozny}@ibspan.waw.pl

Erkki Oja
Helsinki University of Technology
Laboratory of Computer and Information Science
P.O.Box 5400, 02015 Hut, Finland
E-mail: erkki.oja@hut.fi

Library of Congress Control Number: 2005931603

CR Subject Classification (1998): F.1, I.2, I.5, I.4, G.3, J.3, C.2.1, C.1.3

ISSN 0302-9743
ISBN-10 3-540-28752-3 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-28752-0 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11550822 06/3142 5 4 3 2 1 0

Preface

This volume is the first part of the two-volume proceedings of the International Conference on Artificial Neural Networks (ICANN 2005), held on September 11–15, 2005 in Warsaw, Poland, with several accompanying workshops held on September 15, 2005 at the Nicolaus Copernicus University, Toruń, Poland.

The ICANN conference is an annual meeting organized by the European Neural Network Society in cooperation with the International Neural Network Society, the Japanese Neural Network Society, and the IEEE Computational Intelligence Society. It is the premier European event covering all topics concerned with neural networks and related areas. The ICANN series of conferences was initiated in 1991 and soon became the major European gathering for experts in those fields.

In 2005 the ICANN conference was organized by the Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland, and the Nicolaus Copernicus University, Toruń, Poland.

From over 600 papers submitted to the regular sessions and some 10 special conference sessions, the International Program Committee selected – after a thorough peer-review process – about 270 papers for publication. The large number of papers accepted is certainly a proof of the vitality and attractiveness of the field of artificial neural networks, but it also shows a strong interest in the ICANN conferences. Because of their reputation as high-level conferences, the ICANN conferences rarely receive papers of a poor quality and thus their rejection rate may be not as high as that of some other conferences. A large number of accepted papers meant that we had to publish the proceedings in two volumes. Papers presented at the post-conference workshops will be published separately.

The first of these volumes, *Artificial Neural Networks: Biological Inspirations*, is primarily concerned with issues related to models of biological functions, spiking neurons, understanding real brain processes, development of cognitive powers, and inspiration from such models for the development and application of artificial neural networks in information technologies, modeling perception and other biological processes. This volume covers dynamical models of single spiking neurons, their assemblies, population coding, models of neocortex, cerebellum and subcortical brain structures, brain–computer interfaces, and also the development of associative memories, natural language processing and other higher cognitive processes in human beings and other living organisms. Papers on self-organizing maps, evolutionary processes, and cooperative biological behavior, with some applications, are also included. Natural perception, computer vision, recognition and detection of faces and other natural patterns, and sound and speech signal analysis are the topics of many contributions in this volume. Some papers on bioinformatics, bioengineering and biomedical applications are also included in this volume.

The second volume, *Artificial Neural Networks: Formal Models and Their Applications*, is mainly concerned with new paradigms, architectures and formal models of artificial neural networks that can provide efficient tools and techniques to model a great array of non-trivial real-world problems. All areas that are of interest to the neural network community are covered, although many computational algorithms discussed in this volume are only remotely inspired by neural networks. A perennial question that the editors and reviewers always face is: how to define the boundary or the limits of a field? What should still be classified as an artificial neural network and what should be left out as a general algorithm that is presented in the network form? There are no clear-cut answers to these questions. Support vector machines and kernel-based methods are well established at neural network conferences although their connections with neural networks are only of a historical interest. Computational learning theory, approximation theory, stochastic optimization and other branches of statistics and mathematics are also of interest to many neural network experts. Thus, instead of asking: Is this still a neural method?, we have rather adopted a policy of accepting all high-quality papers that could be of interest to the neural network community.

A considerable part of the second volume is devoted to learning in its many forms, such as unsupervised and supervised learning, reinforcement learning, Bayesian learning, inductive learning, ensemble learning, and their applications. Many papers are devoted to the important topics in classification and clustering, data fusion from various sources, applications to systems modeling, decision making, optimization, control, prediction and forecasting, speech and text analysis and processing, multimedia systems, applications to various games, and other topics. A section on knowledge extraction from neural networks shows that such models are not always opaque, black boxes. A few papers present also algorithms for fuzzy rule extraction using neural approaches. Descriptions of several hardware implementations of different neural algorithms are also included. Altogether this volume presents a variety of theoretical results and applications covering most areas that the neural network community may be interested in.

We would like to thank, first of all, Ms. Magdalena Gola and Ms. Anna Wilbik for their great contribution in the preparation of the proceedings. Moreover, Ms. Magdalena Gola, Ms. Anna Wilbik, and Ms. Krystyna Warzywoda, with her team, deserve our sincere thanks for their help in the organization of the conference. Finally, we wish to thank Mr. Alfred Hofmann, Ms. Anna Kramer and Ms. Ursula Barth from Springer for their help and collaboration in this demanding publication project.

July 2005 W. Duch, J. Kacprzyk, E. Oja, S. Zadrożny

Table of Contents – Part I

Biological Inspirations

Modeling the Brain and Cognitive Functions

Special Session: The Development of Cognitive Powers in Embodied Systems

Spiking Neural Networks

Associative Memory Models

Models of Biological Functions

Special Session: Projects in the Area of NeuroIT

Evolutionary and Other Biological Inspirations

Self-organizing Maps and Their Applications

Computer Vision

Face Recognition and Detection

Sound and Speech Recognition

Bioinformatics

Biomedical Applications

Special Session: Information-Theoretic Concepts in Biomedical Data Analysis

Table of Contents – Part II

Formal Models and Their Applications

New Neural Network Models

Supervised Learning Algorithms

Ensemble-Based Learning

Unsupervised Learning

Recurrent Neural Networks

Reinforcement Learning

Bayesian Approaches to Learning

Learning Theory

Artificial Neural Networks for System Modeling, Decision Making, Optimalization and Control

Special Session: Knowledge Extraction from Neural Networks
Organizer and Chair: D. A. Elizondo

Temporal Data Analysis, Prediction and Forecasting

Support Vector Machines and Kernel-Based Methods

Soft Computing Methods for Data Representation, Analysis and Processing

Special Session: Data Fusion for Industrial, Medical and Environmental Applications

Organizers and Chairs: D. Mandic, D. Obradovic

Special Session: Non-linear Predictive Models for Speech Processing

Organizers and Chairs: M. Chetouani, M. Faundez-Zanuy, B. Gas, A. Hussain

Special Session: Intelligent Multimedia and Semantics

Organizers and Chairs: Y. Avrithis, S. Kollias

Applications to Natural Language Proceesing

Various Applications

Special Session: Computational Intelligence in Games
Organizer and Chair: J. Mańdziuk

Issues in Hardware Implementation

Novelty Analysis in Dynamic Scene for Autonomous Mental Development

Sang-Woo Ban and Minho Lee

School of Electrical Engineering and Computer Science,
Kyungpook National University,
1370 Sankyuk-Dong, Puk-Gu, Taegu 702-701, Korea
swban@ee.knu.ac.kr, mholee@knu.ac.kr
http://abr.knu.ac.kr

Abstract. We propose a new biologically motivated novelty analysis model that can give robust performance for natural scenes with affine transformed field of view as well as noisy scenes in dynamic visual environment, which can play important role for an autonomous mental development. The proposed model based on biological visual pathway uses a topology matching method of a visual scan path obtained from a low level top-down attention model in conjunction with a bottom-up saliency map model in order to detect a novelty in an input scene. In addition, the energy signature for the corresponding visual scan path is also considered to decide whether a novelty is occurred in an input scene or not. The computer experimental results show that the proposed model successfully indicates a novelty for natural color input scenes in dynamic visual environment.

1 Introduction

Reinforcement learning is a natural way for developmental models. Reinforcement signals come from two sources; one is extrinsic sources such as human teachers and the other is intrinsic sources such as pain, novelty, boredom, and curiosity. Among these sources for reinforcement learning, internal novelty perception plays important role for intrinsic motivation so as to extend knowledge and adapt to the changing world through external human interaction, which is one of the most important function of the autonomous mental development model.

According to physiological experiments, the removal procedure of the two hippocampi does not seriously affect the person's memory for information stored in the brain prior to removal of the hippocampi [1]. However, after removal, these persons have very little capacity for storing verbal and symbolic types of memories in long-term memory, or even in short-term memory lasting longer than a few minutes [1]. Therefore, these persons are unable to establish new long-term memories of those types of information that are the basis of intelligence. The probable answer, why the hippocampus is so important in helping the brain to store new memories, is one of the important output pathways from the reward and punishment areas of the limbic system. Those sensory stimuli, or even thoughts, that cause pain or aversion excite the

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 1 – 6, 2005.

punishment centers, whereas those stimuli that cause pleasure, happiness, or a sense of reward excite the reward centers [1]. All of these together provide the background mood and motivations of the person [1]. The hippocampus especially and to a lesser degree the dorsal medial nuclei of the thalamus, another limbic structure, have proved especially important in making a decision about which of our thoughts are important enough on a basis of reward or punishment to be worthy of memory [1]. From these experiments, we may conclude that a decision of novelty is an important internal motivation for perceiving new things. A number of physiology researches have shown the novelty detection mechanism and the critical roles of the hippocampus for novelty detection [2-3].

Based on these understanding we developed a low level top-down attention model that can generate a human plausible scan path through reinforcing or inhibiting visual information based on human preference, which can be considered as a specific function related with the roles of the reward and punishment centers in the limbic system. Moreover, we developed a novelty scene detection model that can indicate the novelty of the input scene compared with previously experienced ones. This novelty detection model can generate internal self motivation which initiates further higher level processing such as high level scene perception and knowledge representation using information obtained from scene perception, which is essential for a developmental system.

In section 2, the proposed scene novelty analysis model will be described. We describe the experiments of the proposed model in section 3. Conclusion will be made in section 4.

2 Scene Novelty Analysis Model

In a dynamic environment, it is hard to get an input scene aligned with the previous experienced scene. Therefore, in order to develop a scene novelty detection model, it is essential to develop a robust scene description model with tolerance for noise and for slight affine transformed field of view such as translation, rotation, zoom in and zoom out in dynamic vision environment. Such a model can be used to detect novelty by comparing the description for the current scene with that for the experienced scene like the function of hippocampus in our brain. In the proposed model we developed a relative geometrical topology matching method using visual scan paths which is obtained from the low level top-down attention model together with the bottom-up saliency map(SM) model to indicate whether any novelty occurs in a current input scene or not. We can describe an input scene very robustly using the visual scan path because it can preserve the characteristics of input scene with affine transformation as well as with noise in dynamic environment. In addition, the energy signatures obtained from the saliency map are also considered to decide whether a novelty is occurred in the input scene or not. Fig. 1 shows the proposed novelty detection model for an autonomous mental development. In the proposed model, we did not consider the functions of the FEF and pulvinar related with attention and V4/IT and prefrontal cortex related with object perception and higher intelligent process.

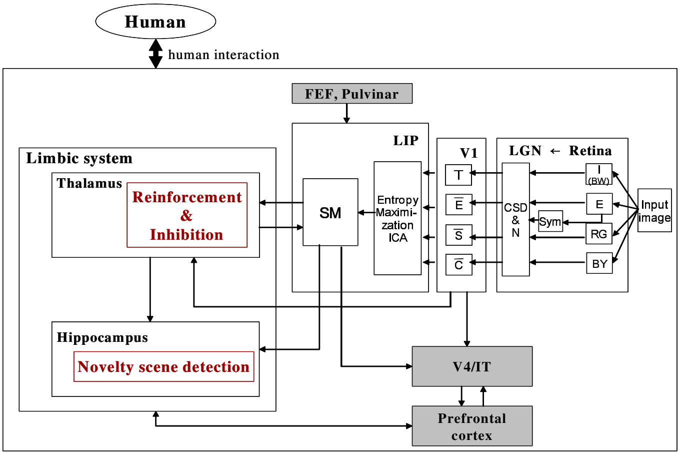


Fig. 1. The proposed biologically motivated novelty scene detection model for an autonomous mental development (I: intensity, E: edge, RG: red-green color opponent, BY: blue-yellow color opponent, sym: symmetry, CSD & N: center-surround difference and normalization, $\overline{I}$: intensity feature map, $\overline{E}$: edge feature map, $\overline{S}$: symmetry feature map, $\overline{C}$: color feature map, ICA : independent component analysis, SM : saliency map, LGN: lateral geniculate nucleus, LIP: lateral interaparietal cortex, FEF: frontal eye field, IT: infero-temporal area).

2.1 Scan Path Generation by a Selective Attention Model

As shown in Fig. 1, the SM is generated by low level top-down reinforcement and inhibition model in conjunction with a bottom-up saliency map model. The bottom-up saliency map model generates a saliency map based on the primitive 4 input features such as intensity, color opponency, edge, and symmetry, which mimics the brain visual pathway from retina to primary visual cortex(V1) and lateral intra parietal cortex(LIP) thorough lateral geniculate nucleus(LGN) [4,5]. The feature maps ($\overline{I}$, $\overline{E}$, $\overline{S}$, and $\overline{C}$) are constructed by center surround difference and normalization (CSD & N) of 4 bases, which mimics the on-center and off-surround mechanism in our brain, and then are integrated by an ICA algorithm [4]. The hierarchical fuzzy ART model can generate more plausible human like saliency map by reflecting human interests incrementally through reinforcing wanted areas and inhibiting unwanted areas [6]. Moreover, it is well known that a fuzzy ART network can be easily trained for additional input patterns without catastrophic forgetting and also can solve the stability-plasticity dilemma in a conventional multi-layer neural network [7]. During the training process, the fuzzy ART network learns and memorizes the characteristics of uninteresting areas and/or interesting areas decided by a human supervisor. After successful training of the fuzzy ART network, an unwanted salient area is inhibited and a desired area is reinforced by the vigilance value of the fuzzy ART network. However, as the number of training patterns increases, the fuzzy ART network becomes time consuming model to reinforce or inhibit some selected areas. For faster analysis to find an inhibition and/or a reinforcement area, we employed the hierarchical structure of the fuzzy ART network. The hierarchical fuzzy ART network consists of a 5 concatenate layer structure in which each layer represents a different hierarchical abstract level of information [6]. The highest level of the model stores the most abstract information that represents a highly abstract cluster. The lowest level of the model stores more detailed information. Fig. 2 shows the architecture of the proposed trainable selective attention model during testing mode.

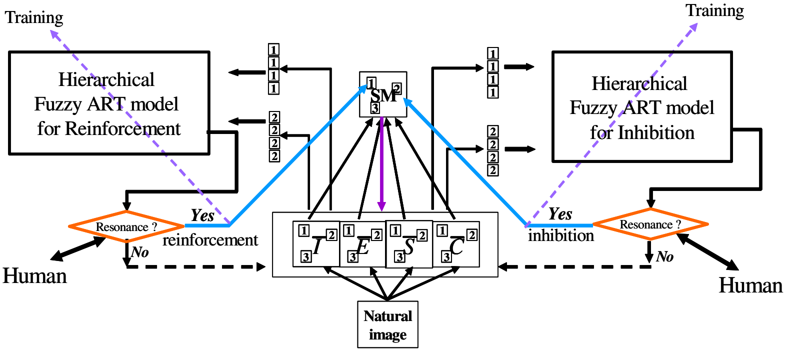


Fig. 2. The architecture of the proposed trainable selective attention model during testing mode; the square blocks 1, 2 and 3 in SM are salient areas.

2.2 Scene Novelty Analysis Using Scan Path Topology and Energy Signatures

The proposed model compares two scan path topologies obtained from the current scene and the experienced scene, of which the procedures are as follows;

1) Input scene is represented as a relative scan path topology vector, T^r_{scene}, composed of 5 center vectors $s_1^{'},\cdots,s_5^{'}$ for 5 dominant salient areas obtained from SM. Each center vector is relatively represented according to the selected reference vector representing the most left-top area among 5 salient areas, where $s_{ref}^{'}$ is a reference vector.

$$T^r_{scene}[s_1^{'}, s_2^{'}, s_{ref}^{'}, s_4^{'}, s_5^{'}] \tag{1}$$

2) Compute a measurement score for the topology.

$$Topology\_Score_{scene} = \sum_{i=1}^{5} \frac{d(s_{ref}^{'}, s_i^{'})}{\max\limits_{d}(d(s_{ref}^{'}, s_i^{'}))}, \quad d : Euclidean\, distance \tag{2}$$

3) Detect novelty by comparing topology score of the current input scene with that of the previously experienced scene.

In order to verify novelty detection in the input scene, the proposed model compares the energy signatures of two scenes. The following procedure describes the procedure for novelty detection using energy signature comparison.

4) Energy signature is represented as a vector with 5 components, $s_{a1}^{'},\cdots,s_{a5}^{'}$,

$$E_{scene}[s_{a1}^{'}, s_{a2}^{'}, s_{a3}^{'}, s_{a4}^{'}, s_{a5}^{'}] \quad , s_{ai}^{'} = \frac{1}{N_{ai}} \sum_{j \in s_{ai}}^{N_{ai}} SM_j \tag{3}$$

where N_{ai} is the number of pixels in the i^{th} salient area s_{ai} and SM_j is the degree of saliency at pixel j in SM.

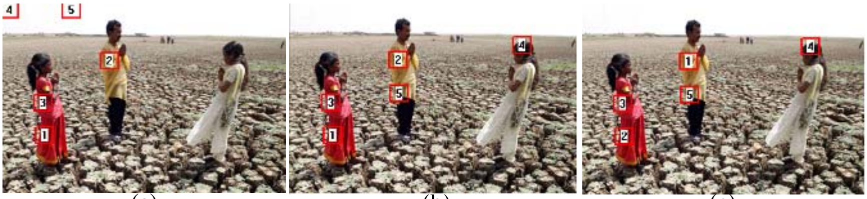
5) Compute energy score using relative energy for the energy signature.

$$Energy\_Score_{scene} = \sum_{i=1}^{5} \frac{s'_{ai}}{\max(s'_{ai})} \tag{4}$$

6) Detect novelty by comparing energy score of the current input scene with that of the previously experienced one.

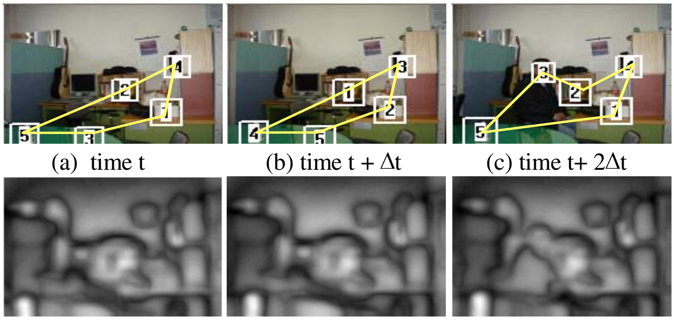
3 Experiments

Fig. 3 shows the simulation results of the low level top-down attention model together with the bottom-up attention model. Fig. 3 (a) shows the scan path generated only by the bottom-up attention model. The numbers in Fig. 3 (a) represent the order of the scan path according to the degree of saliency. Fig. 3 (b) shows the generated scan path after the hierarchical fuzzy ART network for inhibition successfully trained the 4th and 5th salient areas in Fig. 3 (a). After training for reinforcement of the 2nd salient areas in Fig. 3 (b), the proposed low-level top-down attention model can generate more plausible scan path, as shown in Fig. 3 (c). The proposed trainable selective attention model can successfully inhibit an unwanted area and reinforce a desired area through interaction with the human supervisor.



(a) (b) (c)

Fig. 3. Scan paths comparison; (a) by the bottom-up saliency map model, (b) by the low level top-down attention model after inhibition of the 4th and 5th salient areas in (a), (c) by the low level top-down attention model after reinforcement of the 2nd salient area in (b)



(a) time t (b) time t + Δt (c) time t+ 2Δt

(d) time t (e) time t + Δt (f) time t+ 2Δt

Fig. 4. Scan path topologies (a, b and c) and corresponding saliency maps (d, e and f) for three time different scenes assumed as the same scene having a little affine transformed field of view

Fig. 4 shows that the proposed novelty scene analysis model can successfully indicate a scene novelty in dynamic environment. Figs. 4 (a), (b) and (c) show the successive scenes with similar field of view. Fig. 4 (c) has a novelty, while Figs. 4 (a) and (b) do not have any novelty information except slight change of field of view. In dynamic environment, it is hard to get the scenes with the exactly same field of view, rather than we can get the scenes with affine transformed field of view. Also, Figs. 4 (a) and (b) show that topology of scan path can be preserved for an affine transformed scene, while Fig. 4 (c) has different topology of scan path because of novelty. The scan paths in Figs. 4 (a), (b), and (c) are obtained from the corresponding saliency maps in Figs. 4 (d), (e), and (f), respectively. Moreover, Figs. 4 (d) and (e) show that we can get similar energy signatures in two corresponding SMs. Thus, by considering energy signature together with the topology of scan path, we can not only generate the correctness of novelty detection but also discriminate two different scenes having the similar visual scan path topologies. The proposed model showed good performance for indicating the novelty scene in dynamic environments.

4 Conclusion

We proposed a biologically motivated model to detect a novelty in natural color input scenes getting from an affine transformed field of view. The proposed novelty detection model might be successfully applied to an autonomous mental development model that can mentally grow incrementally through human interaction.

Acknowledgement

This research was funded by the Brain Neuroinformatics Research Program of the Ministry of Commerce, Industry and Energy (2005).

References

1. Guyton A.C.: Textbook of medical physiology. 8th edn. W.B. Saunders Co. USA (1991)
2. Yamaguchi, S., Hale, L.A., D'Esposito, M., Knight, R.T.: Rapid prefrontal-hippocampal habituation to novel events. The Journal of Neuroscience 24 23 (2004) 5356-5363
3. Singh, S., Markou, M.: An approach to novelty detection applied to the classification of image regions. IEEE Trans. on Knowledge and Data Eng. 16 4 (2004) 396-407
4. Park, S.J., An, K.H., Lee, M.: Saliency map model with adaptive masking based on independent component analysis. Neurocomputing 49 (2002) 417-422
5. Lanyon, L.J., Denham, S.L.: A model of active visual search with object-based attention guiding scan paths. Neural Networks Special Issue: Vision & Brain. 17 5-6 (2004) 873-897
6. Ban, S.W., Niitsuma, H., Lee, M.: Biologically inspired autonomous mental development model based on visual selective attention mechanism. Working Notes: Developmental Robotics AAAI 2005 Spring Symposium (2005) 99-105
7. Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., Rosen, D.B.: Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. IEEE Trans. on Neural Networks 3 5 (1992) 698-713

The Computational Model to Simulate the Progress of Perceiving Patterns in Neuron Population

Wen-Chuang Chou and Tsung-Ying Sun

National Dong Hwa University, Department of Electrical Engineering,
1, Sec.2, Da Hsueh Rd., Shou-Feng, Hualien, Taiwan
b87501043@ntu.edu.tw, sunty@mail.ndhu.edu.tw

Abstract. We set out here, in an effort to extend the capacities of recent neurobiological evidence and theories, to propose a computational framework, which gradually accumulates and focuses transited energy as a distribution of incitation in the cortex by means of the interaction and communication between nerve cells within different attributes. In our attempts to simulate the human neural system, we found a reproduction of the corresponding perception pattern from that which is sensed by the brain. The model successfully projects a high-dimensional signal sequence as a lower-dimensional unique pattern, while also indicating the significant active role of nerve cell bodies in the central processing of neural network, rather than a merely passive nonlinear function for input and output.

1 Introduction

A phenomenon for proposing the delineation of the mammalian olfactory system and demonstrating some biological evidence of the receptor diversity and specificity is indicated by Buck and Axel [1]. They suggest a model in which each individual subfamily of receptors binds distinct structural types of odorant and the sensory neurons then accept significantly diverse signals at the receptor expression level. The sensory system may use the spatial segregation of sensory inputs to encode the identity of the afferent stimulus and by recalling the odorant memories in the brain, transduce the eventual sensitive information into a specific perception such as the smell of lilac flowers.

Recent neurobiological researchers explicitly indicate the narrow field in which the response of one neuron is confided, and the particular receivable range for the stimulus is known as "receptive field." In other words, every neuron has its own preference for being maximally activated by the certain input signal. Meanwhile, the outline that describes the bell-shaped relationship between external stimuli and fire rate is called as "tuning curve." The phenomena for neurons characterized by distinct receptive fields are generally discovered in all neural systems, such as diverse responses to the chemical compounds of scents in the olfactory system [2], the orientations and contours of objects in the vision system [3], and the moving directions of a arm in the motor system [4].

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 7–12, 2005.

Not only do neurobiologists show great interests on the behaviors of single neurons, but they indicate the assumption of population coding [2][4], that the perception information is coded by a neuron ensemble. In the view of the neural representation of perception Gilbert et al. give a reasonable idea based on the procedures of rate code, line label, vector summation, and probability summation [5]. Lledo et al. also propose the mechanism that in the olfactory system the pattern dynamics oscillates between reciprocal and lateral synapses [2]. And they also agree the sparse coding of information items and the dynamic coding implying correlation and decorrelation. Taking the above conceptions of neurobiological mechanism as our theoretical bases, in this study we would depend on the following two assumptions: (1) In the neuron array, nerve cells with distinct receptive fields can be partially activated by specific stimulus; (2) In the way of the diversity processing of units the results of neuron activations can be combined together to express comprehensive patterns.

2 Computational Model

Motivated by the activities of the human cerebral cortex in neurobiology, many researchers continued to design computational neural models based on the self-organization of afferent connections to the cortex, forced by exterior stimuli, for generating the topographic maps [6],[7]. The model we propose also intends to imitate the workings of human brain by combining the ideas inspired by the neurobiological mechanism of the sensory system. Like a typical self-organizing network, as shown in Fig.1(a), these neurons are placed on the nodes of a lattice. Via two-way connections, they are all analogous to the phenomenon of synaptic interactions' linking to neighbors for laterally exchanging messages after received signals [8]. Moreover, to avoid the neurons' sinking into multifarious information explosions, it is apparently not possible for each of them to connect to input sources as well as the typical framework of the self-organizing map.

In a two-dimension neural map let the vector $\mathbf{i}_{i,j}$ denotes the sum of the overall signal vectors received by the neuron $n_{i,j}$ either from afferent of lateral connections, where the subindices i and j serve to the coordinate on the map. The preference vector $\mathbf{p}_{i,j}$ of the neuron $n_{i,j}$ indicates that the stimulation $\mathbf{i}_{i,j}$ can maximize the activation (in the apex of the tuning curve in Fig.1(b)) of the neuron $n_{i,j}$ if $\mathbf{i}_{i,j}$ is equal to $\mathbf{p}_{i,j}$. As Fig.1(c) indicates, the smaller the angle θ between input signal vector $\mathbf{i}_{i,j}$ and preference vector $\mathbf{p}_{i,j}$, the higher the neuron will be fired. The vicinity of the preference vector in vector space can be also regarded as a receptive field. Because a nerve cell has a maximal limitation of response to any stimuli, according to the magnitude of the stimulus, the norm of the input vector $\mathbf{i}_{i,j}$, the activation of the neuron $a_{i,j}$ can be calculated by

$$a_{i,j} = \mathcal{F}(\mathbf{i}_{i,j}, \mathbf{p}_{i,j}) = \begin{cases} \frac{\mathbf{i}_{i,j} \cdot \mathbf{p}_{i,j}}{|\mathbf{i}_{i,j}||\mathbf{p}_{i,j}|} & \ldots |\mathbf{i}_{i,j}| > |\mathbf{p}_{i,j}| \\ \frac{\mathbf{i}_{i,j} \cdot \mathbf{p}_{i,j}}{|\mathbf{p}_{i,j}|^2} & \ldots |\mathbf{i}_{i,j}| < |\mathbf{p}_{i,j}| \end{cases} \tag{1}$$

where || means the norm operator and the result is between +1 and −1. If the strength of input vector is within the maximal limitation to which neurons can

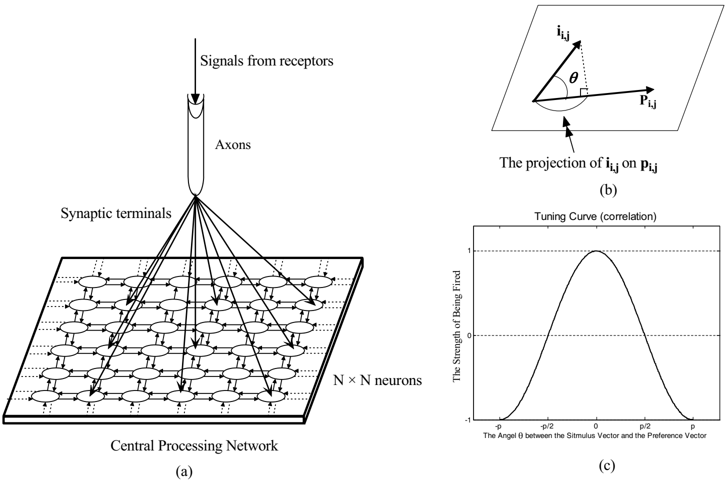

Signals from receptors
Axons
Synaptic terminals
N × N neurons
Central Processing Network
(a)
$\mathbf{i}_{\mathbf{i,j}}$
θ
$\mathbf{P}_{\mathbf{i,j}}$
The projection of $\mathbf{i}_{\mathbf{i,j}}$ on $\mathbf{p}_{\mathbf{i,j}}$
(b)
Tuning Curve (correlation)
The Strength of Being Fired
1
0
-1
-p
-p/2
0
p/2
p
The Angel θ between the Sitmulus Vector and the Preference Vector
(c)

Fig. 1. (a)Model architecture (b)The relationship between two vectors (c)Tuning curve

respond, the activation of neuron is calculated with being multiplied by a linear diminution.

After the original signal activating neurons, the residual energy is consecutively transited to stimulate other immediate neurons so that the neuron population can oscillate and exchange information, if they contain similar receptive fields [3], and then synthesize the distribution of resonant energy to produce a perceptual pattern. The concept to retain the portion of input stimulus in accordance with the attribute of the nerve cell is by projection operation. Because the neuron's ability to transfer energy is limited, the norm of the residual signal $\mathbf{r}_{i,j}$ transmitted to lateral neurons can not exceed the norm of the preference vector $\mathbf{p}_{i,j}$. If the above condition is not satisfied, the residual signal $\mathbf{r}_{i,j}$ will be decreased to the preference vector $\mathbf{p}_{i,j}$. The residual signal $\mathbf{r}_{i,j}$ is defined as follows and $\tilde{\mathbf{r}}_{i,j}$ is a dummy variable:

$$\begin{aligned} \tilde{\mathbf{r}}_{i,j} &= \mathcal{G}(\mathbf{i}_{i,j}, \mathbf{p}_{i,j}) = \frac{\mathbf{i}_{i,j} \cdot \mathbf{p}_{i,j}}{|\mathbf{p}_{i,j}|^2} \mathbf{p}_{i,j} \\ \mathbf{r}_{i,j} &= \begin{cases} \mathbf{p}_{i,j} \cdots |\tilde{\mathbf{r}}_{i,j}| > |\mathbf{p}_{i,j}| \\ \tilde{\mathbf{r}}_{i,j} \cdots |\tilde{\mathbf{r}}_{i,j}| < |\mathbf{p}_{i,j}| \end{cases} \end{aligned} \tag{2}$$

It is considerable that after obtaining one external signal in one epoch, the neural system must push each afferent stimulus to thoroughly vibrate all of the neurons on the map. In this article, we suppose that the oscillation frequency "F" of each neuron with neighbors in the time interval between one input stimulus and the next is the number of synaptic connections between adjacent input nodes. While external information flows sequentially into the network, the old residual signals simultaneously participate in the work of activating neurons as well. The residual activity of energy distribution can resonate with the following stimulus if the sequence comprises significant relation to the temporal domain. This phenomenon also indicates that the resultant pattern generated by the neural network embraces the temporal information of sequence order. The whole computation steps can be described as:

- When the nth stimulus $\mathbf{s}(n)$ injects into map ($t = 1$)

$$\mathbf{i}_{i,j}(n,t) = \mathbf{s}_{i,j}(n) + \sum \mathbf{r}_{m,n}(n-1,F) \tag{3}$$
$$\alpha_{i,j}(n,t) = \alpha_{i,j}(n-1,F) + \mathcal{F}(\mathbf{i}_{i,j}(n,t), \mathbf{p}_{i,j}) \tag{4}$$
$$\mathbf{r}_{i,j}(n,t) = \mathcal{G}(\mathbf{i}_{i,j}(n,t), \mathbf{p}_{i,j}) \tag{5}$$

- After the nth stimulus injecting, during the interval of oscillation ($t = 2 \sim F$)

$$\mathbf{i}_{i,j}(n,t) = \sum \mathbf{r}_{m,n}(n,t-1) \tag{6}$$
$$\alpha_{i,j}(n,t) = \alpha_{i,j}(n,t-1) + \mathcal{F}(\mathbf{i}_{i,j}(n,t), \mathbf{p}_{i,j}) \tag{7}$$
$$\mathbf{r}_{i,j}(n,t) = \mathcal{G}(\mathbf{i}_{i,j}(n,t), \mathbf{p}_{i,j}) \tag{8}$$

The subindex "m, n" means the location nearby the neuron $n_{i,j}$, and α represents the accumulated value of neuron activation during the active period.

Table 1. The attributes of neuron subfamilies and test odorants

	The extent of preference regarding those signals from ten types of odorant receptors									
Subfamilies	I	II	III	IV	V	VI	VII	VIII	IX	X
The 1st subfamily	0.2	0.8	1	0	0	0	0	0	0	0
The 2nd subfamily	0	0	0	0	0.5	1	0.5	0	0	0
The 3rd subfamily	0	0	0	0	0	0	0	0.6	0.6	0.8
	The activating extent of odors to ten types of odorant receptors									
Test odorants	I	II	III	IV	V	VI	VII	VIII	IX	X
The 1st test odorant	0	0	0	0.2	0.2	0.2	0	0	0	0
The 2nd test odorant	0	0	0	0	0	0.001	0.001	0.001	0	0

3 Experiment

In this section, we implement the proposed neural model to simulate the human olfactory perception system as an example, and this model can be also applied to any other sensory systems. Here we emphasize the interaction of cortical neurons for the population coding of patterns. The scheme of the olfactory sensory system is showed in the fig.2(a), the diversity of receptors contribute the distinct activations of corresponding chemical molecules to the cortex[2]. As showed in Table1, the three subfamilies of cortical neurons possess different preference vectors $\mathbf{p}_{i,j}$, purposely assumed a high-dimension to exhibit the generalization of this model, for the ten types of the signals from receptors. For each subfamily we can, again, create five similar neurons of which the lengths of preference vectors $\mathbf{p}_{i,j}$ based on normal distribution. These five neurons are arranged in row order, and their center means are the values in the corresponding row of Table 1, and the variance is always “one”. However, all the directions of vectors are the same in a identical subfamily because the directional attributes determine the preferred characters of neurons. Meanwhile, the attributes of the neurons near the border are influence by two subfamilies. In order to make the result of the perceptual pattern more obvious and intuitive we extend neural map to the size 45 × 45 with the 1 × 15 row basis. Finally, we allow some neurons connect to afferent axons and be spaced apart by four nodes in such a way that the frequency “F” is fixed to five. Because our map is well developed, similar neurons are arranged together as like a brick wall in Fig.2(b).

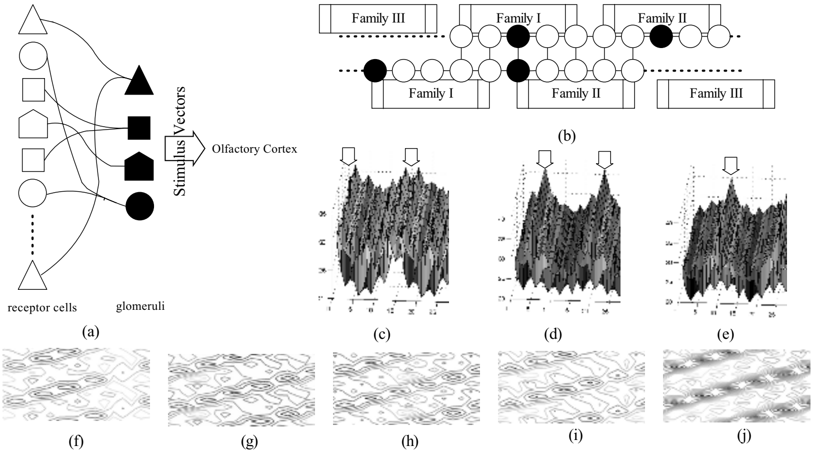


Fig. 2. (a)In sensory system the glomeruli gather distinct corresponding signals from a large number of receptors within the same receptive fields to olfactory cortex. (b)The brick organization of neurons; the black nodes present the afferent connections. (c)-(e)The distribution of activation in the z-axis are showed as curved surfaces upon 45 by 45 neurons placed in x and y-axis directions (parts of the graphs). (f)-(j)The contour maps show the variations of being fired distribution in height, and please note the differences in the outlines representing the valleys of the activation distribution (parts of the graphs).

At first we attempt to use three test odors whose ability to activate the ten receptors are identical to the attribute of the three subfamilies showed in Table1 to stimulate the neural map. The result of the patterns is illustrated respectively in Fig. 2(c)-(e). They show the distribution of the accumulated activations of vital neurons, and the z-axis describes the strength of being fired according to the tuning curve in Fig.1(c). It can be easily observed that the locations of the peaks indicated by the arrowheads move form left to right reflecting the same order of the neuron arrangements. The evidence intensely supports that in those regions the neurons with the receptive fields similar with the identical input stimuli are synchronous fired. Moreover, we choose two random odors to test our map, and the different contours of patterns stimulated by all referred five odors are presented in Fig.2(f)-(j).

4 Conclusions

In numerous trials, the artificial neural network succeeded to be established in a simple presentation of mathematics by approximately utilizing the recent neuroscience findings and theories of the human sensory system and to reproduce the spatiotemporal perceptual pattern aroused by exterior stimulus. The degrees of the activations of neuron population are determined by the correlation between the diversity attribute of cells and the afferent or lateral stimuli in the form of vectors. Our model perfectly conforms to the fact that if an input signal is near the receptive field of the nerve cell, it is possible that the cell would be fired. In the aspect of the neuron interactions the projection function plays a significant role as a mediator who brings the opinions coherent with the attribute of the former neuron to the latter neuron. A synchronous firing would occur when it satisfies the condition that pairs of cell have similar viewpoints on a certain topic. Eventually, like other self-organizing maps the high dimensional world signals through the interactions of neurons can be projected onto the one- or two-dimensional energy distribution map.

References

1. Buck, L.B., Axel, R.: A novel multigene family may encode odorant receptors: a molecular basis for odor recognition. Cell **65** (1991) 175–187
2. Lledo, P., Gheusi, G., Vincent, J.: Information processing in the mammalian olfactory system. Physiological Reviews **85** (2005) 281–317
3. Usrey, W., Reid, R.: Synchronous activity in the visual system. Annual Review of Physiology **61** (1999) 435–456
4. Maynard, E., Hatsopoulos, N., Ojakangas, C., Acuna, B., Sanes, J., Normann, R., Donoghue, J.: Neuronal interactions improve cortical population coding of movement direction. Journal of Neuroscience **19** (1999) 8083–8093
5. Gilbert, C.D., Sigman, M.: The neural basis of perceptual learning. Neuron **31** (2001) 681–697
6. Kohonen, T.: The self-organizing map. Proceedings of the IEEE **7** (1990) 1464–1480
7. Amari, S.I.: Topographic organization of nerve fields. Bulletin of Mathematical Biology **42** (1980) 339–364
8. Haines, D.E.: Fundamental Neuroscience. 2nd edn. Churchill Livingstone, New York (2002)

Short Term Memory and Pattern Matching with Simple Echo State Networks

Georg Fette and Julian Eggert

[1] Technische Universität München, Boltzmannstr. 3, 85748 Garching/München, Germany
fette@in.tum.de
[2] HONDA Research I. Europe GmbH, Carl-Legien-Str.30, 63073 Offenbach/Main, Germany
julian.eggert@honda-ri.de

Abstract. Two recently proposed approaches to recognize temporal patterns have been proposed by Jäger with the so called Echo State Network (ESN) and by Maass with the so called Liquid State Machine (LSM). The ESN approach assumes a sort of "black-box" operability of the networks and claims a broad applicability to several different problems using the same principle. Here we propose a simplified version of ESNs which we call Simple Echo State Network (SESN) which exhibits good results in memory capacity and pattern matching tasks and which allows a better understanding of the capabilities and restrictions of ESNs.

1 Introduction

Both, ESN and LSM, are three-layered networks consisting of input/hidden/output layers, which are used in matching tasks using input-output sequences. To solve the task the input signal is fed into a highly recurrent hidden layer. During the presentation of the signals the state-vector of the network's hidden layer is logged. After the complete input vector was fed into the system the logged activity is used to adjust the output weights in a way that the network's output matches the desired output pattern. Whereas the ESN proposed by Jäger (see e.g. [1]) is a discrete time, nonlinear, recurrent network, the LSM proposed by Maass (see e.g. [2]) is a biologically inspired model with integrate-and-fire neurons and dynamically adapting synapses. Both approaches were used to detect temporal patterns with different time lags [1]-[4] between input and output. For reasons of brevity, in the following we just want to describe the main properties of the two models and refer to the specific publications for the details.

1) They both use a network with an input layer which is connected with a hidden layer. Maass connects about 30% of the hidden layer with the input layer, Jäger fully connects the input towards the hidden layer. 2) The hidden layer is heavily recurrent. Jäger uses a fully recurrent hidden layer, whereas Maass uses a spatial connection topology in which the neurons are more densely connected to their local neighbors than to remote neurons and about 10% of the connections are inhibitory. 3) The connectivity in both approaches is stochastic. 4) The weights to the output layer are set with a one-step learning rule, adjusting them in a way that the difference between the actual output and the desired output over the whole training period is minimized. To achieve this goal, both approaches use linear regression on the logged states of the hidden layer and the desired outputs.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 13–18, 2005.

2 SESN

The coarse structure of a SESN is similar to that of an ESN, also consisting of three layers. Input layer I contains just a single neuron u. We define $u(t)$ as the input to the network and $u = [u(0), .., u(t_{max})]^T$ as the vector of inputs in the time interval $t \in [0, t_{max}]$. u is connected to every neuron x_v in the hidden layer $H \quad (|H| = n)$ with weight $i_v = 1 \quad (v \in [1, n])$. Each unit x_v in H is just connected to itself with weight $d_v \in]0, 1[$ randomly taken from a uniform distribution $(d_i \neq d_j, \forall i, j \in [1, n], i \neq j)$. All hidden units are fully connected to the output layer O. The output layer just contains a single neuron. The output weights $\mathbf{w} \in \mathcal{R}^n$ from H to O are set in a single learning step by linear regression as explained later. There are no connections from O to H or to I, so further readout neurons can be added and analyzed in parallel without affecting the rest of the system. All neurons have a linear transfer function. The state of the system is described by the vector $\mathbf{x}(t) = [x_1(t), .., x_v(t)]^T$ of the hidden layer. Every unit x_v is updated according to $x_v(t) = d_v\, x_v(t-1) + u(t)$. The output unit calculates its value by $o(t) = \mathbf{x}\,\mathbf{w}^T$. The desired output at every time step is defined as $\hat{o}(t)$. The vector $\hat{\mathbf{o}} = [\hat{o}(0), .., \hat{o}(t_{max})]^T$ describes the desired outputs for the time interval $t \in [0, t_{max}]$. When the proper output weights have to be calculated, the input signal $u(t)$ is fed into the system and the state vectors $\mathbf{x}(t)$ are logged forming the matrix $X \in \mathcal{R}^{n \times (t_{max}+1)}$

$$X_{v,t} = x_v(t) \tag{1}$$

The pseudoinverse $X^\dagger$ of this matrix is then used to calculate the weights by

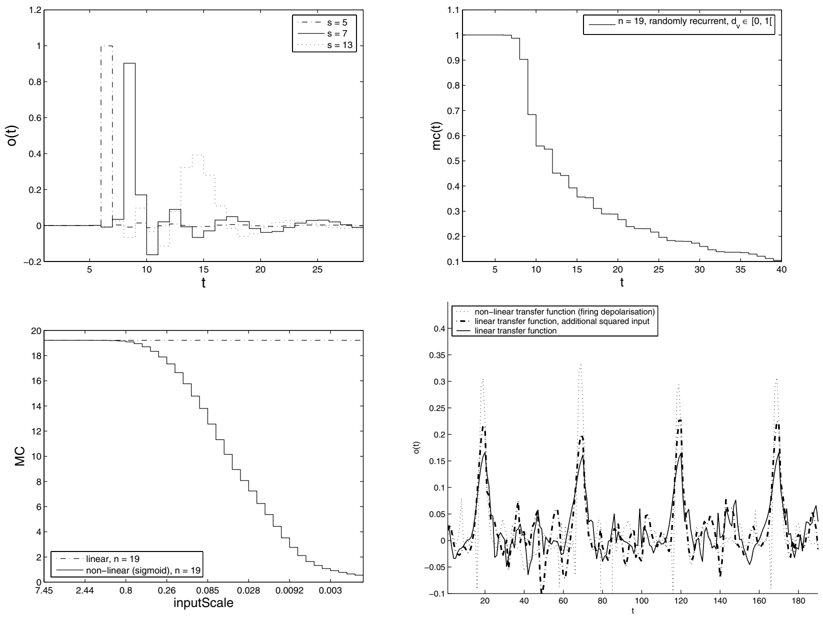
$$\mathbf{w} = X^\dagger\, \hat{\mathbf{o}}. \tag{2}$$

3 SESNs for Memory Reconstruction

We now want to find the output weights $\mathbf{w}_s$ that allow to recover the past input $u(t-s)$ (shifted by s) at $o(t)$ for arbitrary inputs, so that the overall network acts as a sort of delay line. This task can be differently expressed by: The system shall map the input impulse $\mathbf{u} = [1, 0, .., 0]^T \in \mathcal{R}^{t_{max}+1}$ to the desired output sequence $\hat{\mathbf{o}}_s = [0, .., 0, 1, 0, .., 0]^T, (o_{s,s} = 1) \in \mathcal{R}^{t_{max}+1}$. This is a feasible approach because a continuous linear system can be completely described by its pulse response and the signals of an arbitrary input are processed by the system by superposition of the signal's individual composing pulses. For the time-discrete SESNs from (2) this still holds approximately as long as s is of the same order of magnitude as n. With the above assumptions on $\mathbf{u}$ and $\hat{\mathbf{o}}_s$ the matrix X simplifies to $X_{v,t} = {d_v}^t$. The output weights $\mathbf{w}_s$ are then calculated using (2). We call the discrete response $p_s(t)$ of the system on a discrete impulse $\delta(t)$ ($\delta(1) = 1;\ \delta(x) = 0,\ x \neq 1$) its kernel, with

$$p_s(t) = \mathbf{d}^t\, \mathbf{w}_s \tag{3}$$

and $\mathbf{d}^t$ the elementwise exponentiation of $\mathbf{d}$. When we now feed the system with an arbitrary input signal $u(t)$ the output unit's response can be directly calculated by folding the input with the kernel so that $o_s(t) = (p_s * u)(t)$ and training time can be



reduced by several orders of magnitude as compared to standard ESNs. Furthermore, we can easily calculate the partial ($mc(s)$) and total (MC) memory capacities for our network configuration. Jäger defined the total memory capacity [3] as the ability of the network to extract the variation of former input from the system when it is fed with white noise input $u(t)$[1]:

$$mc(s) = \frac{(cov_t(\hat{o}_s(t), o_s(t)))^2}{\sigma_t^2(\hat{o}_s(t))\, \sigma_t^2(o_s(t))} \qquad MC = \sum_{s=0}^{\infty} mc(s) \tag{4}$$

For SESNs it holds that $MC = n$ and

$$p_s(s) = mc(s) \tag{5}$$

(for the proof see appendix of [4]) which means that the kernel p_s at time step s indicates how much of the signal $u(t-s)$ can still be retrieved by $o(t)$. We observe that the maximum peak $p_s(s)$ of the kernel response gets smaller with increasing s, meaning the memory capacity decreases when the time lag grows. This relationship can be seen in the top left plot of the figure at the end of the paper, were we show 3 kernels $p_s(t)$ that resulted from the system. When we equip the hidden neurons with a nonlinear transfer function like $f(x) = \tanh(x)$ the proper kernel p_s cannot be computed in the easy way explained above, but must be calculated extensively by propagating a white noise signal $u(t)$ through the network and using formulas (1), (2) and (3), since (5) does not

[1] $\mu_x(f(x)) = \langle f(i)\rangle_i \, \sigma_x^2(f(x)) = \langle (f(i) - \mu_x(f(x)))^2\rangle_i \, cov_x(f(x), h(x)) = \langle f(i)\, h(i)\rangle_i$.

hold any more. The nonlinearities lower the memory capacity of SESNs, because larger inputs are squashed and cannot be retrieved by the linear output weights in the same way smaller input values are retrieved. As a sigmoid behaves almost linearly for inputs around 0, memory capacity in this case gets dependent on the scaling of the input signal, being maximal for small scale inputs (linear limit) and decreasing for larger inputs. The reduction in memory capacity by nonlinearities was already described by Jäger [3] and can be observed in the bottom left plot of the figure, where we show the memory capacity as a function of the input scaling for linear and nonlinear SESNs. In the top right quadrant of our figure we also show a typical partial memory capacity plot which exhibits a characteristic drop at $t \approx n/4$, so that if a network with almost complete memory reconstruction capability of the last, say, $\hat{s}$ time steps is desired, one would have to use a network of size $n \approx 4\hat{s}$.

4 SESNs for Pattern Matching

With the presented mechanism we now want the network to detect an arbitrary binary temporal pattern $g(t) \in \{0, 1\} \quad (t \in [0, t_g])$. We feed the network with the pattern. At time t_g, after the pattern presentation ends, we want $\hat{o}(t_g) = 1$ at the output if the learned pattern was presented and otherwise $\hat{o}(t_g) = 0$. To accomplish this task we take a network of proper size (considering the arguments gained from the empirical results for memory reconstruction from the previous section we use $n = 4t_g$), so that the ability to recover the pattern signals at the entire length of the presented patterns is almost 1. We now superpose the output weights $\mathbf{w}_s$, for every time shift $s \in [0, t_g]$ for which we want to map the input vector $\mathbf{u} = [1, 0, .., 0]^T$ to the output vector $\hat{\mathbf{o}}_{s,g} = [0, .., 0, k_s, 0, .., 0]^T$, with $k_s = \frac{2\,g(s)-1}{t_g}$. By performing this superposition also the different kernels p_s are summed to a pattern-detecting kernel $\hat{p}$. The summed weight vector $\mathbf{w}_g = \sum_{s=0}^{t_g} \mathbf{w}_s$ now reacts on g with the output $o(t_g) = 1$ and on every other pattern with a lower excitation. In addition, the systems output is continuous so that small changes from the original pattern only lead to small reductions in the output excitation.

Nevertheless in this mode of operation the system performs approximately as a linear filter after learning. To overcome this penalty without adding nonlinearities into the system itself, we can add another pool of hidden units of the same size as the first one, which is connected to a further input neuron, which supplies the square $(u(t))^2$ of the input signal. We also need a further bias unit u_b supplying a permanent bias signal $u_b(t) = 1$ which is connected to the output unit with the weight w_b. If we set the kernel of the first hidden pool to the negative inverse pattern, i.e., $p_1(t) = -2\,g(t - t_g)$ the second pool to $p_2(t) = 1$ and the bias weight to the summed squared pattern values $(w_b = \sum_{i=0}^{t_g} g(i)^2)$, we get the following output:

$$o(t) = (u * p_1)(t) + (u^2 * p_2)(t) + \sum_{i=0}^{t_g} g(i)^2 = \sum_{i=0}^{t_g} u(t-i)\,(-2)\,g(t-i) + \sum_{i=0}^{t_g} u(t)^2\,1 + \sum_{i=0}^{t_g} g(t-i)^2 = \sum_{i=0}^{t_g} (u(t-i) - g(t-i))^2 = \sum_{i=0}^{t_g} (u(i) - g(i))^2 \quad (6)$$

The resulting total kernel calculates the summed squared distance between the input pattern and the learned pattern. Thus only the learned pattern produces a maximum excitation at the output and every other pattern produces less activity.

Another possibility to make SESN capable to detect arbitrary patterns is to add nonlinearities into the system itself by introducing nonlinear activation functions. To train the network with $n = 100$ the system is fed by white noise in which every 50 time steps a pattern $g(t) = -(-2 + \frac{4t}{t_g})^2$ with $t_g = 20$ is inserted. Again only at the end of a pattern presentation the output shall be 1 and otherwise 0. The size of the training set was 2000 time steps. The calculation of $\mathbf{w}_s$ has to be done by formulas (1) and (2). As can be observed in the figure bottom right where as nonlinearity was introduces by a transfer function which depolarizes the neurons after the threshold 2 is surpassed performance rises significantly. This suggests that the enhanced capabilities of the nonlinear network originate from exploiting the nonlinearities in a way that the network computes higher orders of the input signal internally and computes a similar distance measure as in (6).

5 Comparing SESN to ESN

When we want to compare ESNs with SESNs it is better to describe the differences between the two models, because they are very similar: 1) In standard ESNs the hidden units have a nonlinear, sigmoid transfer function. As we have seen, by introducing nonlinearities in SESNs, the memory capacity is reduced with respect to the input scaling, but on the other hand the ability to match patterns is enhanced. In our experiments we have seen that pattern matching performance again decreases, when the input is drastically scaled down [4]. 2) In ESNs the input weights are drawn randomly from a normal distribution. In linear SESNs the input weights to each hidden neuron can be multiplied into the output weight and the input weight itself can be set to 1. 3) In ESNs the hidden units are highly interconnected with each other. A linearized version of the ESN can straightforwardly be transformed into a SESN by diagonalizing the ESNs hidden unit connectivity matrix and multiplying the resulting matrices D and D^{-1} which occur during this process into the input-output-signals. In SESNs, there is no interconnectivity between the units. 4) In ESNs all recurrent connections are drawn randomly from a normal distribution, whereas SESNs get their recurrent connections as specified in section 2.

We have seen that linear SESNs have the same (maximal) memory capacity as the (also linear) ESNs, as shown in the figure bottom left and proved in [3] and [4]. We have also seen that nonlinear SESNs and nonlinear ESNs behave very similarly in pattern matching tasks, suggesting that the ESN learning procedure may select the weights in a way that the ESNs are effectively reduced to the simplified network structures as suggested in this paper.

6 Comparing SESN to LSM

LSMs are based on a continuous model with biologically inspired integrate-and-fire neurons. Therefore they are influenced by heavy nonlinearities from different sources (e.g. depolarization after firing or saturation of membrane potentials). As the nonlinear transformations of incoming signals can only be examined with great difficulties, our conclusions about LSM are based mostly on empirical results.

For comparison, we have implemented SESNs using a continuous time model in which the hidden layer is replaced by a layer of neurons which are just connected with their input and their output and no other (also no recurrent) connections. Each neuron was equipped with a membrane constant that matched that of typical integrate-and-fire neurons from the LSMs (since the recurrent connection weights d define a kind of membrane constant τ in the discrete, linear SESN model). Nonlinear effects lead to a memory capacity reduction. Therefore connections between the neurons with their nonlinear signal exchanging mechanism would result in a loss of memory capacity. On the other hand, as nonlinearities help in pattern matching tasks, a small number of nonlinearities proved to be very useful. In figure bottom right we can see that the introduction of a nonlinearity by depolarizing the neurons membrane potential when a threshold is surpassed is useful to improve the capability to recognize patterns. Again, a SESN implementation with a hidden layer of integrate-and-fire neurons which aren't connected at all with each other performed quite well in memorization as well as in pattern matching tasks in our experiments [4], compared to simulations with LSMs.

7 Conclusions

To better understand the working principles of networks of the ESN and LSM type, we have introduced a recurrent network model called Simple Echo State Network which in comparison has a very reduced complexity and learning costs. In particular the emphasized role of the recurrent connectivity and the nonlinearities can be nicely studied with SESNs. Since SESNs perform comparably well to both ESNs and LSMs on memorization tasks in the linear operation mode and on pattern matching tasks in the nonlinear operation mode, we suggest that they provide an insight into understanding the properties of the two other, much more complex models, and that they give some hints on detecting the main mechanisms leading to their performance.

References

1. H. Jäger, The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, GMD - German National Research Institute for Computer Science, 2001.
2. W. Maass, T. Natschläger, H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural Computation 14, 2002.
3. H. Jäger, Short term memory in echo state networks. GMD Report 152, GMD - German National Research Institute for Computer Science, 2002.
4. G. Fette, Signalverarbeitung in Neuronalen Netzen vom Typ Echo State Networks" diploma thesis (german), 2004.

Analytical Solution for Dynamic of Neuronal Populations

Wentao Huang[1,2], Licheng Jiao[1], Shiping Ma[2], and Yuelei Xu[2]

[1] Institute of Intelligent Information Processing and Key Laboratory
of Radar Signal Processing, Xidian University,
Xi'an 710071, China
wthuang@mail.xidian.edu.cn

[2] Signal and Information Processing Laboratory, Avionic Engineering Department,
College of Engineering, Air Force Engineering University,
Xi'an 710038, China

Abstract. The population density approach is a viable method to describe the large populations of neurons and has generated considerable interest recently. The evolution in time of the population density is determined by a partial differential equation. Now, the discussion of most researchers is based on the population density function. In this paper, we propose a new function to characterize the population of excitatory and inhibitory spiking neurons and derive a novel evolution equation which is a nonhomogeneous parabolic type equation. Moreover, we study the stationary solution and give the firing rate of the stationary states. Then we solve for the time dependent solution using the Fourier transform, which can be used to analyze the various behavior of cerebra.

1 Introduction

In many areas of the brain neurons are organized in populations of units with similar properties. Prominent examples are columns in the visual cortex and somatosensory, and pools of motor neurons. Given a large number of neurons within such a column or pool it is sensible to describe the mean activity of the neuronal population rather than the spiking of individual neurons. Each cubic millimeter of cortical tissue contains about 10^5 neurons. This impressive number also suggests that a description of neuronal dynamics in terms of a population activity is more appropriate than a description on the single-neuron level. Knight *et al* [1], [2] introduce a novel approach to the modeling and simulation of the dynamics of interacting populations of neurons. In this approach, the dynamics of individual neurons, which are described by a state vector $\vec{v}$, determines the evolution of a density function $\rho(\vec{v},t)$. The density function characterizes the behavior of the whole population. The evolution equation in this approach is a partial differential integral (PDE) equation, which describes the evolution of $\rho(\vec{v},t)$ under the influence of neuronal dynamics and a synaptic input. So far, most approaches to solve this PDE numerically are based on finite difference schemes [3], [4], [5]. Sirovich [6] discussed the solutions of some solvable cases. This paper presents a novel view to the population evolution equation. We derive a new population evolution equation and give an analytical solution to analyze the firing rate.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 19 – 24, 2005.

2 The Population Density Model

The population model on which this study is based derives from a neuronal dynamics described based on the simple integrate-and-fire equation

$$\frac{dv}{dt} = -\lambda v(t) + s(t)\ ; \quad 0 \le v \le 1\ , \tag{1}$$

where the trans-membrane potential, *v*, has been normalized so that *v* = 0 marks the rest state, and *v* = 1 the threshold for firing. When the latter is achieved *v* is reset to zero. λ, a frequency, is the leakage rate and *s*(*t*), also having the dimensions of frequency, is the normalized current due to synaptic arrivals at the neuron.

Under the statistical approach one considers a population of *N* neurons, each following (1), so that $N\rho(\vec{v},t)dv$ specifies the probable number of neurons, at time *t*, in the range of states (*v*, *v* + *dv*). $\rho(\vec{v},t)$, the probability density, may be shown to be governed by

$$\begin{aligned}\frac{\partial\rho}{\partial t} &= -\frac{\partial}{\partial v}J + \delta(v)r(t) = -\frac{\partial}{\partial v}[-\lambda v\rho + \sigma(t)\int_{v-h}^{v}\rho(v',t)dv'] + \delta(v)r(t-\tau) \\ &= -\frac{\partial}{\partial v}(-\lambda v\rho) + \sigma(t)(\rho(v-h,t) - \rho(v,t)) + \delta(v)r(t-\tau)\end{aligned}\ , \tag{2}$$

where *h* is the membrane voltage jump due to a spike arrival, τ is the refractory period, $\sigma(t)$ is the external input rate of spikes, and *J* is the neuronal flux in the state space and *r*(*t*) is the firing rate of the population and is given by the flux of neurons leaving at the threshold value of the membrane potential

$$r(t) = J(\rho, t-\tau)_{v=1} = \sigma(t)\int_{1-h}^{1}\rho(v',t)dv'\ . \tag{3}$$

The boundary conditions and initial data are

$$\begin{cases}\int_0^1 \rho(v,t)dv + \int_{t-\tau}^{t} r(t')dt' \equiv 1 \\ \rho(v=1,t) = 0 \\ \rho(v,t=0) = q(v)\end{cases}\ . \tag{4}$$

Since the number of neurons is preserved, the flux of neurons leaving the interval must equal that entering at the resting state

$$J(\rho,t)_{v=0} = -\lambda v\rho + \sigma(t)\int_{v-h}^{v}\rho(v',t)dv'\Big|_{v=0} = \sigma(t)\int_{0^-}^{0^+}\rho(v',t)dv' = J(\rho,t-\tau)_{v=1} = r(t)\ . \tag{5}$$

This model may be extended to include inhibition, membrane dynamics, a richer set of reversal potentials and stochastic effects, as well as more complicated neuronal models [2], [3], [4].

3 The Modified Population Equation and Its Analytical Solution

Sometimes we are more interest in firing rate $r(t)$ than $\rho(v,t)$. If we solved the firing rate $r(t)$ by equation (2) and equation (3), the computational process would be discommodious and complicated. In order to overcome the difficulties, we use following transformation first:

$$P(v,t)=\int_{-\infty}^{v}\rho(v',t)dv',\quad P(v,0)=Q(v)=\int_{-\infty}^{v}q(v')dv'. \tag{6}$$

Integrating equation (2) from 0 to *v* on two side, and substituting (6) into (2) derives

$$\frac{\partial P}{\partial t}-\lambda v\frac{\partial P}{\partial v}=\sigma(t)(P(v-h,t)-P(v,t))+H(v)r(t-\tau),\quad v\in[0,1]. \tag{7}$$

Here *H*(*v*) is the Heaviside step function, which equate 0 when *v* <0, otherwise equate 1. The firing rate *r*(*t*) is

$$r(t)=\sigma(t)[P(1,t)-P(1-h,t)]. \tag{8}$$

From above, moreover, we can get boundary conditions

$$P(0,t)=\frac{r(t-\tau)}{\sigma(t)},\qquad P(1,t)=1-\int_{t-\tau}^{t}r(t')dt',\qquad \frac{\partial P(v,t)}{\partial v}\Big|_{v=1}=0. \tag{9}$$

When $v\notin[0,1]$, $P(v,t)=0$. Generally, for the following quasilinear first order partial differential equations:

$$\frac{\partial\phi(v,t)}{\partial t}-av\frac{\partial\phi(v,t)}{\partial v}=-b(t)\phi(v,t)+g(v,t), \tag{10}$$

where $\phi(v,0)=\Theta(v)$. We can get its analytical solution:

$$\phi(v,t)=\Theta(ve^{at})e^{-\eta(t)}+\int_{0}^{t}g(ve^{a(t-t')},t')e^{\eta(t')-\eta(t)}dt', \tag{11}$$

where $\eta(t)=\int_0^t b(t')dt'$. Then, from (7), we have

$$P(v,t)=Q(ve^{\lambda t})e^{-\eta(t)}+\int_0^t[\sigma(t')P(ve^{\lambda(t-t')}-h,t')+r(t'-\tau)]e^{\eta(t')-\eta(t)}dt', \tag{12}$$

where $\eta(t)=\int_0^t\sigma(t')dt'$. Equation (12) gives an iterative approach to solve $P(v,t)$ for us. The value of current state $P(v,t)$ in each fixed position (*v*, *t*) is determined by the integrate value of previous subinterval. From the point of view of signal processing, this can be regarded as a spatio-temporal recursion filter.

However, the above method does not give an immediate analytical solution, and is inconvenient to applications for us.

When $h\to 0$, let us consider the Taylor expansion

$$P(v-h,t)\approx P(v,t)-h\frac{\partial P(v,t)}{\partial v}+\frac{h^2}{2}\frac{\partial^2 P(v,t)}{\partial v^2}. \tag{13}$$

But, when $v<h$, $P(v-h,t)=0$, the equation (13) can not be used for estimating $P(v-h,t)$. We adopt

$$\begin{cases} P(v-h,t)\approx P(v,t)-h\dfrac{\partial P(v,t)}{\partial v}+\dfrac{h^2}{2}\dfrac{\partial^2 P(v,t)}{\partial v^2}+h(v,t) \\ h(v,t)=-H_h(v)P(0)=-H_h(v)\dfrac{r(t-\tau)}{\sigma(t)} \\ H_h(v)=H(v)-H(v-h) \end{cases}, \tag{14}$$

where $h(v,t)$ can regard as compensate function when $v \in (0,h)$. Substituting (14) into (7) yields:

$$\begin{cases} \frac{\partial P}{\partial t} - \lambda v \frac{\partial P}{\partial v} = -h\sigma(t)\frac{\partial P(v,t)}{\partial v} + \frac{h^2\sigma(t)}{2}\frac{\partial^2 P(v,t)}{\partial v^2} + f(v,t) \\ f(v,t) = H(v-h)r(t-\tau) \end{cases}, \quad v \in [0,1] . \tag{15}$$

First, let us consider the stationary states. In this case, $\sigma(t) = \sigma_0$, $\frac{\partial P(v,t)}{\partial t} = 0$, $P(v,t) = P_0(v)$, $r(t) = r_0$, then, from (15) we get

$$\frac{h^2\sigma_0}{2}\frac{\partial^2 P_0(v)}{\partial v^2} + (\lambda v - h\sigma_0)\frac{\partial P_0(v)}{\partial v} + f(v) = 0 . \tag{16}$$

where $f(v) = H(v-h)r_0$. The solution of (16) is

$$P_0(v) = -\int_0^v \exp(-\frac{(v_2-b)^2}{2a^2})\int_0^{v_2}\frac{2r_0}{h^2\sigma_0}H(v_1-h)\exp(\frac{(v_1-b)^2}{2a^2})dv_1 dv_2 + C_1\int_0^v \exp(-\frac{(v_1-b)^2}{2a^2})dv_1 + C_2 , \tag{17}$$

where $a = \sqrt{h^2\sigma_0 / 2\lambda}$, $b = h\sigma_0 / \lambda$,C_1, C_2 are constant and satisfy the boundary conditions (9), finally, we obtain

$$C_1 = \frac{2r_0}{h^2\sigma_0}\int_0^1 H(v_1-h)\exp(\frac{(v_1-b)^2}{2a^2})dv_1, \qquad C_2 = \frac{r_0}{\sigma_0}H(v) , \tag{19}$$

and

$$r_0^{-1} = \begin{Bmatrix} \tau + \sigma_0^{-1} + \frac{2}{h^2\sigma_0}[\int_0^1 H(v_1-h)\exp(\frac{(v_1-b)^2}{2a^2})dv_1 \\ -\int_0^1 \exp(-\frac{(v_2-b)^2}{2a^2})\int_0^{v_2} H(v_1-h)\exp(\frac{(v_1-b)^2}{2a^2})dv_1 dv_2] \end{Bmatrix} . \tag{20}$$

Next, we discuss how to solve the equation (15). It can be expressed as following a nonhomogeneous parabolic type equation

$$\begin{cases} \frac{\partial P}{\partial t} - \lambda v \frac{\partial P}{\partial v} = -h\sigma(t)\frac{\partial P(v,t)}{\partial v} + \frac{h^2\sigma(t)}{2}\frac{\partial^2 P(v,t)}{\partial v^2} + H(v-h)r(t-\tau), v \in (0,1) \\ P(v,0) = Q(v), \quad P(0,t) = \frac{r(t-\tau)}{\sigma(t)}, \quad P(1,t) = 1 - \int_{t-\tau}^{t} r(t')dt' \end{cases} \tag{21}$$

This is a mixed problem that possesses initial value and boundary value simultaneously. It is a challenge for us to solve. We adopt the following assume

$$\begin{cases} Y(v,t) = P(v,t)[H(v) - H(v-1)] = \begin{cases} P(v,t), & v \in [0,1] \\ 0, & v \notin [0,1] \end{cases} \\ P_v'(0,t) = \frac{\partial P(0,t)}{\partial v} = 0, \quad P_v'(1,t) = \frac{\partial P(1,t)}{\partial v} = 0 \end{cases} , \tag{22}$$

From (22) we have

$$\begin{cases} \frac{\partial Y(v,t)}{\partial v} = \frac{\partial P(v,t)}{\partial v}[H(v) - H(v-1)] + [P(0,t)\delta(v) - P(1,t)\delta(v-1)] \\ \frac{\partial^2 Y(v,t)}{\partial v^2} = \frac{\partial^2 P(v,t)}{\partial v^2}[H(v) - H(v-1)] + [P(0,t)\delta'(v) - P(1,t)\delta'(v-1)] \end{cases} \tag{23}$$

Substituting (23) into (21) yields:

$$\begin{cases} \dfrac{\partial Y(v,t)}{\partial t} - \lambda v \dfrac{\partial Y(v,t)}{\partial v} = -h\sigma(t)\dfrac{\partial Y(v,t)}{\partial v} + \dfrac{h^2\sigma(t)}{2}\dfrac{\partial^2 Y(v,t)}{\partial v^2} + F(v,t) \\ Y(v,0) = Y_0(v) = Q(v)[H(v) - H(v-1)] \end{cases}, \tag{24}$$

where

$$\begin{cases} F(v,t) = [H(v-h) - H(v-1)]r(t-\tau) + g(v,t) \\ g(v,t) = (h\sigma(t) - \lambda v)[P(0,t)\delta(v) - P(1,t)\delta(v-1)] - \dfrac{h^2\sigma(t)}{2}[P(0,t)\delta'(v) - P(1,t)\delta'(v-1)] \end{cases} \tag{25}$$

Applying Fourier transform to (24) yields

$$\frac{\partial \tilde{Y}(s,t)}{\partial t} + \lambda s \frac{\partial \tilde{Y}(s,t)}{\partial s} = (-\lambda - h\sigma(t)js - \frac{h^2\sigma(t)}{2}s^2)\tilde{Y}(s,t) + \tilde{F}(s,t)\ . \tag{26}$$

where $\tilde{Y}(s,0) = \tilde{Y}_0(s)$. Solving the quasilinear first order partial differential equations obtains

$$\begin{cases} \tilde{Y}(s,t) = \tilde{Y}_0(se^{-\lambda t})e^{\eta(s,t)} + \int_0^t \tilde{F}(se^{-\lambda(t-t')},t')e^{\eta(s,t)-\eta(s,t')}dt' \\ \eta(s,t) = -\lambda t - jshe^{-\lambda t}\int_0^t \sigma(l)e^{\lambda l}dl - s^2\dfrac{h^2e^{-2\lambda t}}{2}\int_0^t \sigma(l)e^{2\lambda l}dl \end{cases} \tag{27}$$

The inversion of Fourier transform of $\tilde{Y}(s,t)$, i.e. $Y(v,t) = \Gamma^{-1}[\tilde{Y}(s,t)]$ is

$$\begin{cases} Y(v,t) = Y_0(ve^{\lambda t}) * \Gamma^{-1}[e^{\eta(s,t)}] + \int_0^t F(ve^{\lambda(t-t')},t') * \Gamma^{-1}[e^{\eta(s,t)-\eta(s,t')}]dt' \\ \eta(s,t) = -\lambda t - jshe^{-\lambda t}\int_0^t \sigma(l)e^{\lambda l}dl - s^2\dfrac{h^2e^{-2\lambda t}}{2}\int_0^t \sigma(l)e^{2\lambda l}dl \end{cases}, \tag{28}$$

where $*$ is convolution operator. From (28), we have

$$Y(v,t) = e^{-\lambda t}Y_0(ve^{\lambda t}) * U(v + c_1(t),t) + \int_0^t e^{-\lambda(t-t')}F(ve^{\lambda(t-t')},t') * U'(v + c_1(t) - c_1(t'),t,t')dt'\ , \tag{29}$$

and the firing rate $r(t)$

$$\begin{aligned} r(t) = \sigma(t)[Y(1,t) - Y(1-h,t)] = \sigma(t)[\int_{-\infty}^{\infty} e^{-\lambda t}[Y_0((1-v')e^{\lambda t}) - Y_0((1-h-v')e^{\lambda t})]U(v' + c_1(t),t)dv' + \\ \int_0^t \int_{-\infty}^{\infty} e^{-\lambda(t-t')}[F((1-v')e^{\lambda(t-t')},t') - F((1-h-v')e^{\lambda(t-t')},t')]U'(v' + c_1(t) - c_1(t'),t,t')dv'dt'] \end{aligned} \tag{30}$$

where

$$\begin{cases} U(v,t) = \Gamma^{-1}[e^{-s^2c_2(t)}] = \dfrac{1}{\sqrt{4\pi c_2(t)}}e^{-\frac{v^2}{4c_2(t)}} \\ U'(v,t,t') = \Gamma^{-1}[e^{-s^2(c_2(t)-c_2(t'))}] = \dfrac{1}{\sqrt{4\pi(c_2(t) - c_2(t'))}}e^{-\frac{v^2}{4(c_2(t)-c_2(t'))}} \\ c_1(t) = -he^{-\lambda t}\int_0^t \sigma(l)e^{\lambda l}dl, \qquad c_2(t) = \dfrac{h^2e^{-2\lambda t}}{2}\int_0^t \sigma(l)e^{2\lambda l}dl \end{cases}, \tag{31}$$

4 Conclusion

In this paper we have presented a novel analytical approach to study the population evolution equation. For computing the firing rate directly, we adopt a transformation to the density function $\rho(\vec{v},t)$ and obtain the state function $P(v,t)$. We derive a new evolution equation from the original equation, and give the approach that deduces its analytical solution. We partition the state v into successive subintervals and can solve the value of the current state $P(v,t)$ from the value of previous state $P(v-h,t)$, which can be regarded as a spatio-temporal recursion filter. Then we discuss a method to solve for the approximative solution, which derive a nonhomogeneous parabolic type equation and get a relation of computing firing rate.

References

1. Knight, B.: Dynamics of Encoding in Neuron Populations: Some General Mathematical Features. Neural Computation, 12 (2000) 473–518
2. Omurtag, A., Knight, B., Sirovich, L.: On the Simulation of Large Populations of Neurons. Journal of Computational Neuroscience, 8 (2000) 51–63
3. Nykamp, D.Q., Tranchina, D.: A Population Density Approach that Facilitates Large-Scale Modeling of Neural Networks: Analysis and an Application to Orientation Tuning. Journal of Computational Neuroscience, 8 (2000) 19–50
4. Casti, A., Omurtag, A., Sornborger, A., Kaplan, E., Knight, B., Victor, J., Sirovich, L.: A Population Study of Integrate-and-Fire-or-Burst Neurons. Neural Computation, 14 (2002) 947–986
5. de Kamps, M.: A Simple and Stable Numerical Solution for the Population Density Equation. Neural Computation, 15(2003) 2129–2146
6. Sirovich, L.: Dynamics of Neuronal Populations: Eigenfunction Theory; Some Solvable Cases. Network: Computation in Neural Systems. 14 (2003) 249–272

Dynamics of Cortical Columns – Sensitive Decision Making

Jörg Lücke[1,2]

[1] Institut für Neuroinformatik, Ruhr-Universität, 44780 Bochum, Germany
[2] Gatsby Computational Neuroscience Unit, UCL, London WC1N 3AR, UK

Abstract. Based on elementary assumptions on the interconnectivity within a cortical macrocolumn we derive a differential equation system which models the mean neural activities of its minicolumns. A stability analysis shows a rich diversity of stationary points and sensitive behavior with respect to a parameter of inhibition. If this parameter is continuously changed, the system shows the same types of bifurcations as the macrocolumn model presented in [1] which is based on explicitly defined interconnectivity and spiking neurons. Due to this behavior the macrocolumn is able to make very sensitive decisions with respect to external input. The decision making process can be used to induce self-organization of receptive fields as is shown in [2].

Keywords: cerebral cortex, cortical columns, non-linear dynamics, stability analysis, bifurcations.

1 Introduction

In [1] a model of a cortical neural module called macrocolumn or *segregate* [3] was defined which is based on spiking neurons and columnar interconnectivity. The model showed neuroscientifically desirable properties and far reaching functional capabilities such as high sensitivity to external input and fast reaction times. In this paper we show how a continuous neural dynamics with qualitatively the same properties can be derived from few elementary assumptions about macrocolumnar connectivity.

2 Dynamics of Minicolumn Activities

Motivated by neuroanatomical findings (see, e.g., [4,5] or [6,7] for an overview) we assume a macrocolumn to consist of equal minicolumns and we take each minicolumn to be equally and inhibitorily coupled to the mean activities p_α of all minicolumns in the macrocolumn, i.e., we assume the dynamics to be invariant under permutations of minicolumns. An equation system which models such a macrocolumn is given by[1]:

$$\frac{d}{dt}p_\alpha = f(p_\alpha, h(\boldsymbol{p}))\,, \quad \alpha = 1, \ldots, k\,, \tag{1}$$

[1] Note that we neglect external input to the minicolumns for the moment.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 25–30, 2005.

with functions $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ and $h : \mathbb{R}^k \to \mathbb{R}$. p_α is the mean activity of minicolumn α and h is invariant under all permutations of its arguments $(I = \{1, \dots, k\})$:

$$\forall \boldsymbol{x} \in \mathbb{R}^k \; \forall \sigma : I \to I \text{ permutations} : \; h(x_1, \dots, x_k) = h(x_{\sigma(1)}, \dots, x_{\sigma(k)}) \,. \tag{2}$$

The function h models the inhibitory input to a minicolumn, i.e, it models the effect of inhibitory postsynaptic potentials (IPSPs) on currently active neurons of a minicolumn. As explicit inhibitory coupling between the minicolumns we choose motivated by the inhibition function in [1]:

$$h(\boldsymbol{p}) = \nu \max_{\beta=1,\dots,k} \{p_\beta\} \,. \tag{3}$$

$\nu \in \mathbb{R}$ is an inhibitory gain factor which will play the role of a bifurcation parameter. Note that (3) satisfies the assumption in (2)[2].

Stationary points and stability

To analyze the dynamic behavior of (1) with (3) we first look for stationary points of the system. Consider the set Q of phase space points for fixed $\nu \in \mathbb{R}$ defined as follows:

$$\left.\begin{array}{ll}
\mathcal{P}_1^0 & := \max\{q \in \mathbb{R} \,|\, 0 = f(q, \nu q)\}, \\
\mathcal{P}_i^0 & := \max\{q \in \mathbb{R} \,|\, 0 = f(q, \nu q) \,\wedge\, \forall j < i : \, q \neq \mathcal{P}_j^0 \,\}, \\
\mathcal{P}_i^j & := \max\{q \in \mathbb{R} \,|\, 0 = f(q, \nu \mathcal{P}_i^0) \,\wedge\, q < \mathcal{P}_i^0 \,\wedge\, (\forall r < j : \, q \neq \mathcal{P}_i^r)\}, \\
 & \\
Q_i & := \{\boldsymbol{q} \in \mathbb{R}^k \,|\, \max\limits_{r \in I}\{q_r\} = \mathcal{P}_i^0 \,\wedge\, (\forall r \in I \, \exists j \in \mathbb{N}_o : \, q_r = \mathcal{P}_i^j) \,\}, \\
Q & := \bigcup_i Q_i,
\end{array}\right\} \tag{4}$$

where $I = \{1, \dots, k\}$, $\mathbb{N}_o = \mathbb{N} \cup \{0\}$, $i \in \mathbb{N}$, $j \in \mathbb{N}_o$, and $r \in I$. Note that $\mathcal{P}_i^j$ does not necessarily exist for all j. It can be shown that, for a large class[3] of functions f, the set Q contains all the stationary points of (1) with (3). An element of Q, e.g. $\boldsymbol{q} \in Q_i$, is of the form:

$$\boldsymbol{q} \,=\, (\underbrace{\mathcal{P}_i^0, \dots, \mathcal{P}_i^0}_{l(\boldsymbol{q})}, \underbrace{\mathcal{P}_i^1, \dots, \mathcal{P}_i^1}_{m_1(\boldsymbol{q})}, \dots, \underbrace{\mathcal{P}_i^J, \dots, \mathcal{P}_i^J}_{m_J(\boldsymbol{q})}), \quad l(\boldsymbol{q}) + \sum_{j=1}^{J} m_j(\boldsymbol{q}) = k \,, \tag{5}$$

or any permutation. For a given $\boldsymbol{q} \in Q$ a stability analysis results because of the symmetries in (1) with (3) in the following eigenvalues of the Jacobian:

$$\left.\begin{array}{rll}
\lambda_1 & = (\frac{\partial}{\partial x_1} f)_{(\mathcal{P}_i^0, \nu\, \mathcal{P}_i^0)} \,+\, \nu\, (\frac{\partial}{\partial x_2} f)_{(\mathcal{P}_i^0, \nu\, \mathcal{P}_i^0)} & \text{multiplicity } 1 \\
\lambda_2 & = (\frac{\partial}{\partial x_1} f)_{(\mathcal{P}_i^0, \nu\, \mathcal{P}_i^0)} & \text{multiplicity } (l(\boldsymbol{q}) - 1) \\
\lambda_{2+j} & = (\frac{\partial}{\partial x_1} f)_{(\mathcal{P}_i^j, \nu\, \mathcal{P}_i^0)} & \text{multiplicity } m_j(\boldsymbol{q})
\end{array}\right\} \tag{6}$$

[2] Note that using suitable coordinate transformation a similar analysis is also possible with a larger class of functions satisfying (2).

[3] Essentially f has to be continuous, continuously differentiable, and has to possess a finite number of zero points but weaker assumptions are also possible.

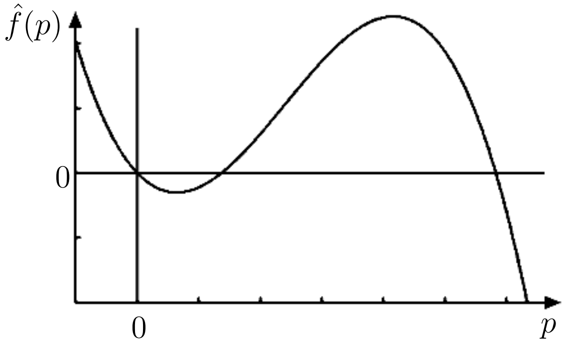


Fig. 1. Activation function $\hat{f}(p) = f(p, \nu p)$ of an isolated minicolumn

An explicit minicolumn activation function
We now choose a specific function $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ for dynamics (1). For $k = 1$ we expect (1) to model the activity dynamics of an isolated minicolumn. Self-excitation due to excitatory connectivity within a minicolumn (see [7] for a review) and bounded activity due to self-inhibition and neural refraction times suggest an activation function $\hat{f}(p) = f(p, \nu p)$ as displayed in Fig. 1. Given very low activity in a minicolumn without input we expect the activity to decay to zero because of finite neural thresholds. For neural activity above a certain level we expect the activity to increase until neural refractoriness and self-inhibition compensate for self-excitation.

A simple choice for f which is consistent with these expectations and Fig. 1 is given by:

$$f(p_\alpha, h(\boldsymbol{p})) = a\, p_\alpha\, (p_\alpha \,-\, h(\boldsymbol{p}) \,-\, \Theta \,-\, b\, p_\alpha^2)\,, \tag{7}$$

where $a, b > 0$, $\Theta \geq 0$. Note that for $k = 1$ the function $f(p, \nu p)$ is a polynomial of order 3. A special case is to choose $b = 1$ and $\Theta = 0$ such that we get the dynamics:

$$\frac{d}{dt} p_\alpha \;=\; a\, p_\alpha\, (p_\alpha \,-\, \nu \max_{\beta=1,\ldots,k} \{p_\beta\} - p_\alpha^2), \quad \text{where } a > 0, \text{ and } \nu \in [0, 1]. \tag{8}$$

Note that the inhibition by other minicolumns cannot drive the activities to non-biological negative values. Other functions $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ are also possible but (7) is especially well analyzable.

For $\nu > \frac{1}{2}$ we get using definitions (4) $\mathcal{P}_1^0 = 1 - \nu$, $\mathcal{P}_2^0 = 0$, $\mathcal{P}_1^1 = 0$. If $\nu < \frac{1}{2}$ we compute $\mathcal{P}_1^1 = \nu$ (instead of zero) and additionally $\mathcal{P}_1^2 = 0$. Thus, for $\nu < \frac{1}{2}$, the stationary points of the system are given by

$$\begin{aligned} Q_1 &= \{(\underbrace{\mathcal{P}_1^0, \ldots, \mathcal{P}_1^0}_{l}, \underbrace{\mathcal{P}_1^1, \ldots, \mathcal{P}_1^1}_{m_1}, \underbrace{0, \ldots, 0}_{m_2}) \text{ and permutations} \,|\, l \geq 1,\, m_{1,2} \geq 0\}\,, \\ Q_2 &= \{(0, \ldots, 0)\}\,, \end{aligned} \tag{9}$$

and for $\nu > \frac{1}{2}$ by

$$\begin{aligned} Q_1 &= \{(\underbrace{\mathcal{P}_1^0, \ldots, \mathcal{P}_1^0}_{l}, \underbrace{0, \ldots, 0}_{m}) \text{ and permutations} \,|\, l \geq 1,\, m \geq 0\}\,, \\ Q_2 &= \{(0, \ldots, 0)\}\,. \end{aligned} \tag{10}$$

Note that, by applying elementary combinatorics to (9) and (10), we get a number of $(3^k - 2^k + 1)$ stationary points for $\nu < \frac{1}{2}$ and 2^k stationary points for $\nu > \frac{1}{2}$. Using (6) the stabilities of the points in Q_1 and Q_2 are for $\nu < \frac{1}{2}$ given by the eigenvalues (together with their multiplicities):

$$\left.\begin{array}{lll} \lambda_1 = a\,(2(1-\nu)\mathcal{P}_1^0 \,-\, 3(\mathcal{P}_1^0)^2) & = -a\,(1-\nu)^2 & \text{mult. } 1 \\ \lambda_2 = a\,((2-\nu)\mathcal{P}_1^0 \,-\, 3(\mathcal{P}_1^0)^2) & = a\,(1-\nu)\,(2\nu-1) & \text{mult. } (l-1) \\ \lambda_3 = a\,(2\mathcal{P}_1^1 \,-\, \nu\mathcal{P}_1^0 \,-\, 3(\mathcal{P}_1^1)^2) & = a\,\nu\,(1-2\nu) & \text{mult. } m_1 \\ \lambda_4 = -a\,\nu\mathcal{P}_1^0 & = -a\,\nu\,(1-\nu) & \text{mult. } m_2 \end{array}\right\} \tag{11}$$

For $\nu > \frac{1}{2}$ we get the same eigenvalues except for λ_3 which does not exist. The stationary point $(0, \ldots, 0)$ of Q_2 has as only eigenvalue $\lambda = 0$ and it turns out to be unstable with polynomial behavior in the vicinity of $(0, \ldots, 0)$. Because of (11) we know that, e.g. for $k = 2$, the set of points $Q^{++} := \{(\mathcal{P}_1^0, 0),\, (0, \mathcal{P}_1^0)\}$ exists and is stable for all $\nu \in (0, 1)$ and that the stable stationary point $Q^{+} := \{(\mathcal{P}_1^0, \mathcal{P}_1^0)\}$ exists for all $\nu \in (0, 1)$ but is only stable for $\nu < \frac{1}{2}$. The set of points in $Q^{-} := \{(\mathcal{P}_1^0, \mathcal{P}_1^1),\, (\mathcal{P}_1^1, \mathcal{P}_1^0)\}$ only exists for $\nu < \frac{1}{2}$ and the points are unstable. The stationary points in Q^{-} define the subcritical branches with respect to Q^{+}.

In Fig. 2 we plotted the phase velocity of (8) for $k = 2$ and two different values of ν. For $\nu < \frac{1}{2}$ we get as non-zero stationary points the three stable points of Q^{++} and Q^{+} and the two unstable points of Q^{-}. If ν is increased to a value greater than $\frac{1}{2}$, the unstable points in Q^{-} merge with the stable point in Q^{+} in the point of structural instability $\nu_c = \frac{1}{2}$ and we get an unstable symmetric stationary point $(\mathcal{P}_1^0, \mathcal{P}_1^0)$ for $\nu > \nu_c$. This dynamic behavior exactly matches the behavior of the macrocolumn model with $k = 2$ minicolumns as it is described in [1]. For higher dimensions we know because of the multiplicities in (11) that all stationary points in a generalized Q^{+} (points in Q^{+} have $l(\boldsymbol{q}) \geq 2$) loose their stability for the same value ν_c ($\nu_c = \frac{1}{2}$ in this case). The dynamics, therefore, generalizes to higher dimensions as the macrocolumn dynamics in [1]. Using (9), (10) and (11) it can further be shown that $(2^k - k - 1)$ non-trivial stable stationary points loose their stability in ν_c.

3 Conclusion

We derived a neural dynamics motivated by cortical connectivity. In contrast to [1], in which an explicit connectivity and a time-discrete neuron model was used, we here derived a dynamics from a small set of more abstract assumptions on macrocolumn connectivity. The resulting system of differential equations (8) represents a continuous time version of the difference equation system discussed

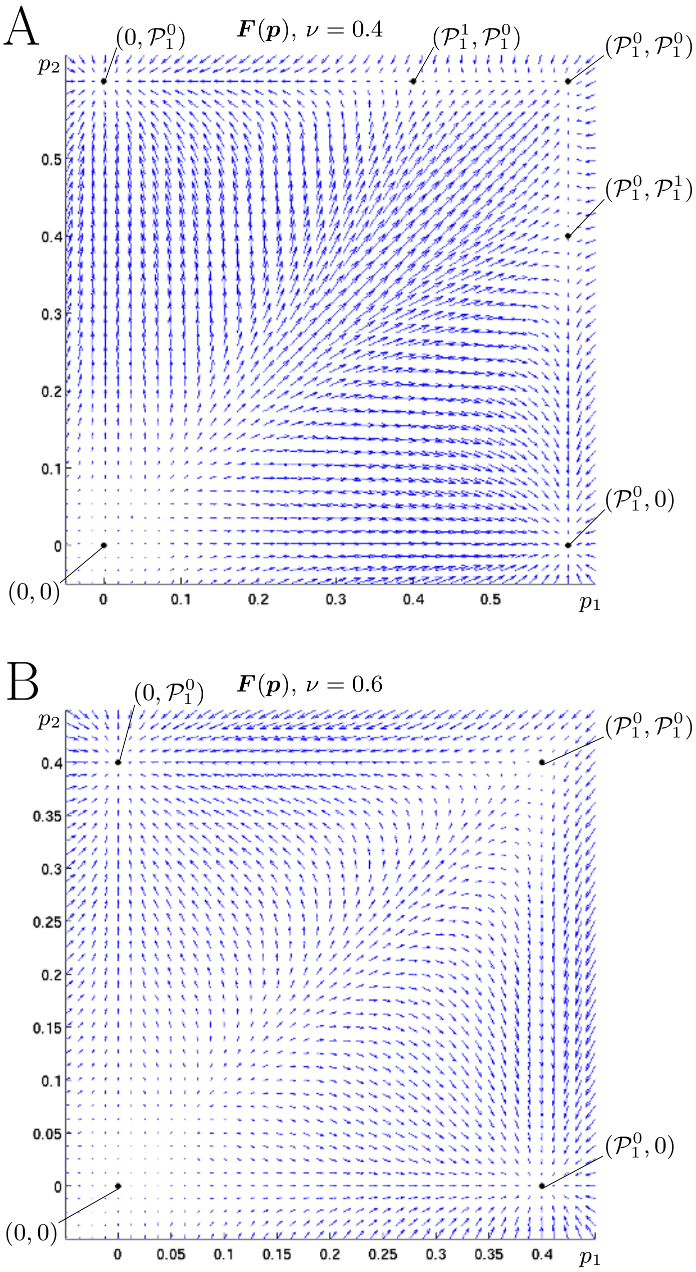


Fig. 2. Phase velocities $\boldsymbol{F}(\boldsymbol{p})$, $F_\alpha(\boldsymbol{p}) = f(p_\alpha, h(\boldsymbol{p}))$, of dynamics (8). **A** Phase velocity for $\nu = 0.4 < \nu_c$. Black points mark stationary points as given in (9). **B** Phase velocity for $\nu = 0.6 > \nu_c$. Black points mark stationary points as given in (10).

in [1]. Dynamics (8) has proven to capture the essential dynamical features of the model in [1], i.e., it spontaneously breaks the symmetry of minicolumn activities if the proportionality factor of inhibition ν is increased. If input to the minicolumns is considered as perturbation of the dynamics, the system breaks the symmetry on the basis of small input differences. Thus, the system is theoretically infinitely sensitive to external input. Using an oscillating ν the dynamics can make sensitive decisions during each oscillation (compare [1]). This behavior is further exploited in [2] where the dynamics is used to enable self-organization of RFs of minicolumns with far reaching computational capabilities.

Compared to the system [1] the dynamics presented in this paper is continuous, more compact and easier to handle than its predecessor. At the same time, it was derived from few assumptions on interconnectivity and is, in a sense, more independent of the concept of minicolumns and macrocolumn, i.e., any neural entities and connectivities giving rise to such an equation system possess equivalent information processing capabilities.

Acknowledgment

Funding by the Gatsby Charitable Foundation is gratefully acknowledged.

References

1. J. Lücke and C. von der Malsburg. Rapid processing and unsupervised learning in a model of the cortical macrocolumn. *Neural Computation*, 16:501 – 533, 2004.
2. J. Lücke and J. D. Bouecke. Dynamics of cortical columns – self-organization of receptive fields. ICANN 2005, accepted.
3. O. V. Favorov and M. Diamond. Demonstration of discrete place-defined columns, segregates, in cat SI. *Journal of Comparative Neurology*, 298:97 – 112, 1990.
4. A. Peters and C. Sethares. The organization of double bouquet cells in monkey striate cortex. *J. of Neurocytology*, 26:779 – 797, 1997.
5. J. M. L. Budd and Z. F. Kisvarday. Local lateral connectivity of inhibitory clutch cells in layer 4 of cat visual cortex. *Exp. Brain Res.*, 140(2):245 – 250, 2001.
6. V. B. Mountcastle. The columnar organization of the neocortex. *Brain*, 120:701 – 722, 1997.
7. D. P. Buxhoeveden and M. F. Casanova. The minicolumn and evolution of the brain. *Brain, Behavior and Evolution*, 60:125–151, 2002.

Dynamics of Cortical Columns – Self-organization of Receptive Fields

Jörg Lücke[1,2] and Jan D. Bouecke[1]

[1] Institut für Neuroinformatik, Ruhr-Universität, 44780 Bochum, Germany
[2] Gatsby Computational Neuroscience Unit, UCL, London WC1N 3AR, UK

Abstract. We present a system of differential equations which abstractly models neural dynamics and synaptic plasticity of a cortical macrocolumn. The equations assume inhibitory coupling between minicolumn activities and Hebbian type synaptic plasticity of afferents to the minicolumns. If input in the form of activity patterns is presented, self-organization of receptive fields (RFs) of the minicolumns is induced. Self-organization is shown to appropriately classify input patterns or to extract basic constituents form input patterns consisting of superpositions of subpatterns. The latter is demonstrated using the bars benchmark test. The dynamics was motivated by the more explicit model suggested in [1] but represents a much compacter, continuous, and easier to analyze dynamic description.

Keywords: cerebral cortex, cortical columns, non-linear dynamics, self-organization, receptive fields.

1 Introduction

The minicolumn is believed to be the smallest neural module consisting of roughly a hundred neurons which are stacked orthogonal to the cortical surface. Axons and dendrites of pyramidal cells in the same minicolumn bundle together and are assumed to be strongly interconnected [2,3]. Connectivity of inhibitory cells suggests inhibition between the minicolumns. Minicolumns combine together to what is called a *macrocolumn* or *segregate* [4]. Minicolumns of a macrocolumn receive input from the same source, e.g. a patch of the body surface, but in the adult brain they react differently to different types of stimuli, i.e., the minicolumns possess different RFs. At birth afferents to cortical columns are found to be relatively unspecific (see, e.g., [5]). The subsequent specialization is believed to be mainly driven by synaptic plasticity and to crucially depend on sensory input.

In this paper a dynamical system is presented which models the neural dynamics of minicolumn activities in a macrocolumn and the specialization of minicolumnar RFs on the basis of Hebbian plasticity.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 31–37, 2005.

2 Activity of Minicolumns

Instead of explicitly modeling the neural connectivity within a macro- and minicolumn [1] we consider an abstract dynamics of the activity p_α of minicolumn α in a macrocolumn of k minicolumns[1]:

$$\frac{d}{dt}\, p_\alpha = a\, p_\alpha\, \left(p_\alpha - h(\boldsymbol{p}) - p_\alpha^2\right) \,+\, \kappa\, I_\alpha \,+\, \sigma\, \eta_t\,, \tag{1}$$

where a is a time constant, κ the coupling strength to external input I_α, and where σ^2 is the variance of zero-mean Gaussian white noise[2] η_t. h is a function of the activities of all k minicolumns of the macrocolumn $\boldsymbol{p} = (p_1, \ldots, p_k)$. Dynamics (1) is a simple choice for modeling mini- and macrocolumn properties. An abstract derivation of (1) and a non-linear analysis of its dynamical properties can be found in [6]. The different summands on the right-hand-side (rhs) of (1) can be considered as modeling different neuro-dynamical aspects:

$a\,(p_\alpha)^2$ models self-excitation by excitatory interconnectivity within a minicolumn.

$-a\,(p_\alpha)^3$ models negative feed-back due to neural refraction times which naturally limit the minicolumn activity.

$-a\,p_\alpha\,h(\boldsymbol{p})$ models inhibition by the minicolumns of the macrocolumn. The function $h(\boldsymbol{p})$ is greater than zero. Note that because of the multiplication with p_α this term cannot drive the activity to non-biological negative values.

If we choose as inhibition function $h(\boldsymbol{p}) = \nu \max_{\beta=1,\ldots,k}\{p_\beta\}$, it can be shown that for zero input and zero noise the system possesses exponentially many stationary points [6]. $\nu \in (0,1)$ plays the role of a bifurcation parameter. For $\nu \in (0, \frac{1}{2})$ there are $(3^k - 2^k + 1)$ and for $\nu \in (\frac{1}{2}, 1)$ there are 2^k stationary points. In the point of structural instability, $\frac{1}{2} = \nu_c$, $2^k - k - 1$ non-trivial stable stationary points loose their stability in subcritical bifurcations. Analytical expressions for all stationary points and for their stabilities can be derived [6]. The system qualitatively reproduces the bifurcations observed in the explicit model defined in [1]: if ν is increased from a value $\nu < \frac{1}{2}$ to $\nu > \frac{1}{2}$ and if the macrocolumn has been in its symmetric stable state (all minicolumns are equally active) the minicolumns are deactivated via a process of symmetry breakings[3] (compare [1]). For non-zero input the symmetry is broken on the basis of small differences between the inputs I_α to the minicolumns. The smallest value of ν for which the deactivation of a minicolumn occurs is a measure for the input strength relative to the other inputs.

[1] $p_\alpha(t)$ can be thought of as the fraction of neurons in minicolumn α that have spiked during a short fixed time-interval around t.

[2] Which is taken to be different for each α.

[3] Note that for symmetry breakings infinitesimal perturbations are required, e.g., using a non-zero noise term with very small standard deviation.

3 Self-organization of Receptive Fields

We operate the system with oscillating inhibitory gain factor ν. An oscillation or *ν-cycle* starts with a $\nu = 0$ interval during which the system stabilizes the symmetric stable stationary state under the influence of noise ($\sigma > 0$). Subsequently, ν is increased from a value $\nu_{\min}$ to a maximal value $\nu_{\max}$ (see Fig. 1).

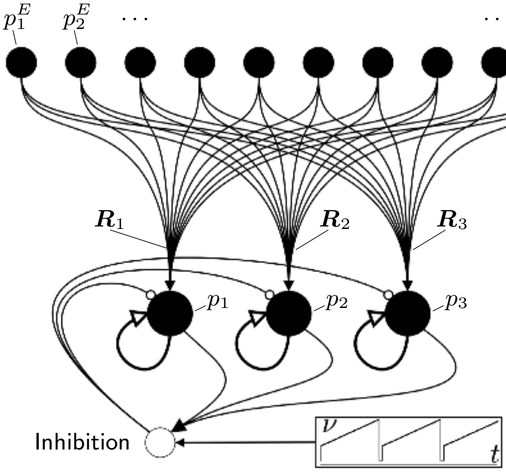


Fig. 1. Sketch of a macrocolumn with $k = 3$ minicolumns. Interactions are indicated using arrows. The inhibition between minicolumns is visualized using a symbolic inhibitory neuron.

The input to the minicolumns originates from a set of input units p_1^E to p_N^E ($p_j^E \in [0,1]$) and is mediated by afferent connections $R_{\alpha j}$: $I_\alpha = \sum_{j=1}^{N} R_{\alpha j} p_j^E$. The afferent fibers we take to be subject to synaptic plasticity of the form:

$$\frac{d}{dt} R_{\alpha j} = \left(\mathcal{E}\, p_\alpha\, p_j^E \;-\; (\sum_{l=1}^{N} \mathcal{E} p_\alpha\, p_l^E) R_{\alpha j} \right) \mathcal{H}(\chi - A(t))\,, \tag{2}$$

where $\mathcal{H}(x) = 0$ for $x \leq 0$ and $\mathcal{H}(x) = 1$ for $x > 0$ is a step function and where $\mathcal{E}$ is a synaptic growth factor. The positive term on the rhs of (2) models Hebbian type synaptic plasticity. It is only greater than zero if minicolumn α and input unit j are simultaneously active. The negative term insures that $\sum_j R_{\alpha j}$ converges to one for all α. The RFs, $\boldsymbol{R}_\alpha = (R_{\alpha 1}, \ldots, R_{\alpha N})$, are only modified if the over-all activity $A(t) = \sum_{\alpha=1}^{k} p_\alpha$ falls below a threshold χ which ensures that learning takes place only after a number of minicolumns have been deactivated.

The system is sketched in Fig. 1 and the complete dynamics now reads:

$$\nu(t) = \begin{cases} 0 & \text{if } \tilde{t} < T_{\text{init}} \\ (\nu_{\max} - \nu_{\min})\, \frac{\tilde{t} - T_{\text{init}}}{T - T_{\text{init}}} \;+\; \nu_{\min} & \text{if } \tilde{t} \geq T_{\text{init}} \end{cases}, \tag{3}$$

$$\frac{d}{dt}\,p_\alpha = a\,p_\alpha\,\left(p_\alpha - \nu(t)\max_{\beta=1,\ldots,k}\{p_\beta\} - p_\alpha^2\right) + \kappa\sum_{j=1}^{N} R_{\alpha j}p_j^E + \sigma\,\eta_t\,, \tag{4}$$

$$\frac{d}{dt}\,R_{\alpha j} = \left(\mathcal{E}\,p_\alpha\,p_j^E - (\sum_{l=1}^{N}\mathcal{E}p_\alpha\,p_l^E)R_{\alpha j}\right)\,\mathcal{H}(\chi - A(t))\,,\quad \mathcal{E} = \frac{\epsilon}{N}\,, \tag{5}$$

where $\tilde{t} = \mathrm{mod}(t,T)$, i.e., $\tilde{t} = t - nT$ where n is the greatest integer satisfying $t - nT \geq 0$. ϵ is the relative synaptic growth factor.

Equations (4) and (5) are a system of non-linear differential equations coupled to an oscillation given by (3). In simulations the oscillation is chosen to be slow compared to the dynamics of p_α. We study the system behavior by exposing it to different kinds of input. From a given database with different input patterns $\boldsymbol{P} \in [0,1]^N$ we present a randomly chosen pattern $\boldsymbol{P}^o$ during each ν-cycle, i.e., $\boldsymbol{P}^o$ defines the values of the input units for the duration of a ν-cycle, $\boldsymbol{p}^E = \boldsymbol{P}^o$. An input pattern $\boldsymbol{P}$ is, for visualization purposes, displayed as two-dimensional grey-level image.

Before we can start simulating the dynamics we have to choose a suitable set of parameters. To choose a consider an isolated minicolumn without self-excitation ($\frac{d}{dt}p = -a\,p^3$). In this case we expect that, e.g., an activity $p = 1.00$ rapidly decays to a value close to zero, e.g. $p = 0.05$, in about 1ms (the order of magnitude of action potentials and refraction times). For the activity levels $p(0\mathrm{ms}) = 1.00$ and $p(1\mathrm{ms}) = 0.05$ we get $a \approx 200\mathrm{ms}^{-1}$. Note that this is a very coarse estimate due to the arbitrariness in choosing the activities but one obtains the order of magnitude of a. The value of the coupling κ is taken to be only a small fraction of the value for a, $\kappa = 1.0\mathrm{ms}^{-1}$, and standard deviation σ of the Gaussian white noise is taken to be only a fraction of κ, $\sigma = 0.12\mathrm{ms}^{-1}$. For the oscillation of ν (3) we choose a period length of $T = 25$ms and a time of $T_{\mathrm{init}} = 2$ms with $\nu = 0$ and additional noise to reset the dynamics. After initialization ν is increased from $\nu_{\min} = 0.3$ to a value $\nu_{\max} = 0.55$ which is slightly greater than the critical value $\nu_c = 0.5$. For the dynamics of Hebbian plasticity (5) we choose $\epsilon = 0.2$ and a threshold of $\chi = 0.55$.

4 Simulations

Equations (3) to (5) can now be numerically simulated (e.g., using the Euler method for stochastic differential equations). In the first experiment we use the set of 42 input patterns displayed in Fig. 2A (compare [1]). By simulating dynamics (3) to (5) with $k = 6$ and parameters as given above we get RF self-organization as can be observed in Fig. 2B. After random initialization the RFs specialize to different classes of input patterns. If we have fewer minicolumns than major classes exist, we get coarser RFs (see Fig. 2C) and if we have more minicolumns we get RFs with higher specialization degrees (see Fig. 2D).

In the second experiment we use the bars test [7] in order to demonstrate the system's ability of learning a distributed code for input consisting of sub-pattern superpositions. We operate the system using the same parameters as in

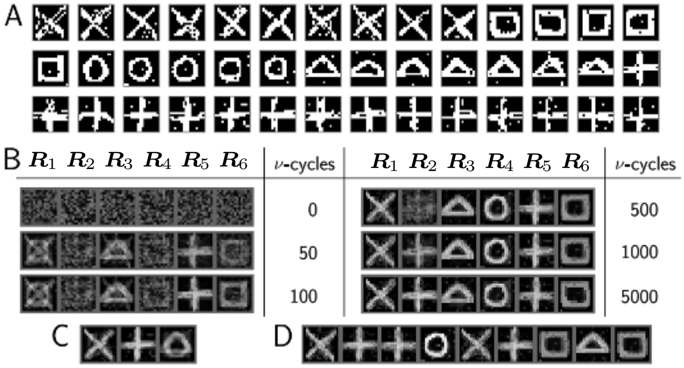

A
B
R_1 R_2 R_3 R_4 R_5 R_6
ν-cycles
0
50
100
R_1 R_2 R_3 R_4 R_5 R_6
ν-cycles
500
1000
5000
C
D

Fig. 2. In **A** the set of input patterns is displayed ($N = 16 \times 16$). During each ν-cycle one randomly chosen pattern of this set is presented. In **B** the modification of the RFs of a macrocolumn with $k = 6$ minicolumns is displayed. After 1000 ν-cycles six different pattern classes are represented. The RFs' degree of specialization further increases thereafter to a final degree. **C** RF specialization (after 250 ν-cycles) if an abstract macrocolumn with $k = 3$ minicolumns is used with the same input. **D** RF specialization (after 10000 ν-cycles) if a macrocolumn with $k = 9$ is used.

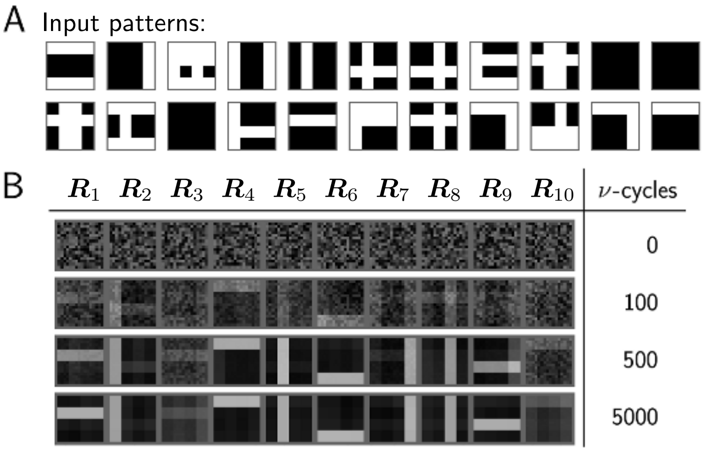

A Input patterns:
B
R_1 R_2 R_3 R_4 R_5 R_6 R_7 R_8 R_9 R_{10}
ν-cycles
0
100
500
5000

Fig. 3. **A** A selection of 22 typical input patterns ($N = 16 \times 16$) of a bars test with 8 different four pixel wide bars. **B** Typical example of the self-organization of the RFs of a macrocolumn with $k = 10$ minicolumns. During each ν-cycle a randomly generated input pattern of the upper type is presented.

the previous experiment and we use a bars test with $b = 8$ bars. Each bar occurs in an image with probability $\frac{1}{4}$ (see Fig. 3A). As can be seen in Fig. 3B, RF self-organization results in a representation of all bars. In 200 considered simulations with $k = 10$ minicolumns a bars test with above parameters required less than 600 ν-cycles in 50% of the cases to represent all bars (less than 410 in 20% and less than 950 ν-cycles in 80% of the simulations). The system found a correct representation for all bars in all simulations and is robust against various perturbations to the bars test. Note that the results for the bars test show an improvement compared to the explicit system presented in [1] which requires more ν-cycles for the same bars test. Thus dynamics (3) to (5) represent not only an abstraction but, at least in the here discussed bars test, also an improvement of the explicit dynamics in [1] (also compare [8]). Note that already the system presented in [1] has on the basis of extensive measurements shown to be highly competitive to all other systems suggested to solve the bars test.

5 Conclusion

On the basis of recurrent activity in cortical minicolumns, oscillatory inhibitory coupling between the minicolumns, and phase coupled Hebbian synaptic plasticity of afferents we derived a system of coupled differential equations which models self-organization of RFs of cortical minicolumns. Self-organization allows a macrocolumn to represent input using distributed activity of its minicolumns relative to an oscillation. The model combines most often independently discussed aspects of neural information processing and is functionally competitive in a standard benchmark test for feature extraction.

Acknowledgment

Funding by the Gatsby Charitable Foundation is gratefully acknowledged.

References

1. J. Lücke and C. von der Malsburg. Rapid processing and unsupervised learning in a model of the cortical macrocolumn. *Neural Computation*, 16:501 – 533, 2004.
2. V. B. Mountcastle. The columnar organization of the neocortex. *Brain*, 120:701 – 722, 1997.
3. D. P. Buxhoeveden and M. F. Casanova. The minicolumn and evolution of the brain. *Brain, Behavior and Evolution*, 60:125–151, 2002.
4. O. V. Favorov and M. Diamond. Demonstration of discrete place-defined columns, segregates, in cat SI. *Journal of Comparative Neurology*, 298:97 – 112, 1990.
5. K. E. Schmidt, D. S. Kim, W. Singer, T. Bonhoeffer, and S. Löwel. Functional specificity of long-range intrinsic and interhemispheric connections in the visual cortex of strabismic cats. *Journal of Neuroscience*, 17:5480 – 5492, 1997.

6. J. Lücke. Dynamics of cortical columns – sensitive decision making. ICANN 2005, accepted.
7. P. Földiák. Forming sparse representations by local anti-Hebbian learning. *Biological Cybernetics*, 64:165 – 170, 1990.
8. J. Lücke. Hierarchical self-organization of minicolumnar receptive fields. *Neural Networks, Special Issue 'New Developments in Self-Organizing Systems'*, 17/8–9:1377 – 1389, 2004.

Optimal Information Transmission Through Cortico-Cortical Synapses

Marcelo A. Montemurro and Stefano Panzeri

The University of Manchester, Faculty of Life Sciences,
Moffat Building, PO Box 88, M60 1QD, Manchester, UK
{m.montemurro, s.panzeri}@manchester.ac.uk

Abstract. Neurons in visual cortex receive a large fraction of their inputs from other cortical neurons with a similar stimulus preference. Here we use models of neuronal population activity and information theoretic tools to investigate whether this arrangement of synapses allows efficient information transmission. We find that efficient information transmission requires that the tuning curve of the afferent neurons is approximately as wide as the spread of stimulus preferences of the afferent neurons reaching a target neuron. This is compatible with present neurophysiological evidence from visual cortex. We thus suggest that the organization of V1 cortico-cortical synaptic inputs allows optimal information transmission.

1 Introduction

A typical neuron in visual cortex receives most of its inputs from other visual cortical neurons which have similar stimulus selectivity [1,2,3]. For instance, orientation selective neurons in superficial layers in ferret visual cortex receive more than 50% of their cortico-cortical excitatory inputs from neurons with orientation preference which is less than 30^o apart. However, this input structure is rather broad in terms of stimulus-specificity since connections between neurons with dissimilar stimulus orientation also exist [4]. The organisation of cortico-cortical connections has received a lot of attention because it may be involved in the generation of orientation tuning. However, little is yet known on whether this structure of inputs allows efficient transmission of sensory information across cortico-cortical synapses. We have previously addressed this issue by introducing a population coding model [5] that quantifies the information about the sensory stimuli that a typical cortical "target" neuron receives through its cortico-cortical synapses. In this paper we extend the previous analysis by studying numerically in more detail the properties of this model. We find that that, under a wide range of spread of stimulus preference values, efficient synaptic information transmission requires that the tuning curve of the afferent neurons is approximately as wide as the spread of stimulus preferences of the afferent fibres reaching the target neuron. By meta-analysis of anatomical and physiological data, we argue that this optimal trade-off is approximately reached in visual cortex. Thus, neurons in visual cortex may be wired to decode optimally information from their synaptic inputs.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 39–44, 2005.

2 Modeling the Activity of the Afferent Neural Population

We assume that the afferent population is made of a large number N of neurons ($N \approx 10000$ in real cortical afferent populations). The response of each neuron $r_k(k = 1, \cdots, N)$ is quantified as the number of spikes fired in a salient post-stimulus time window of a length τ. Thus, the overall population response is represented as a spike count vector $\mathbf{r} = (r_1, \cdots, r_N)$. We assume that the neurons are tuned to a small number D of stimulus parameters, such as *e.g.* orientation, speed or direction of motion. The stimulus will thus be described as a vector $\mathbf{s} = (s_1, \ldots, s_D)$ of dimension D. The number of stimulus features that are encoded by the neuron will be left as a free parameter. In fact, it has been shown [6] that, when considering large populations with a uniform distribution of stimulus preferences (such as an hypercolumn in V1 containing all stimulus orientations) the optimal tuning width of individual neurons depends crucially on the number of stimulus features being encoded. Thus, it is interesting to investigate how the optimal arrangement of cortico-cortical synapses depends on D.

The neuronal tuning function of the $k-th$ neuron ($k = 1, \cdots, N$), which quantifies the mean spike count of the $k-th$ neuron to the presented stimulus, is modelled as a Gaussian distribution, characterised by the following parameters: preferred stimulus $\mathbf{s}^{(k)}$, tuning width σ_f, and response modulation m:

$$f^{(k)}(\mathbf{s}) = m \exp\left(-\frac{(\mathbf{s} - \mathbf{s}^{(k)})^2}{2{\sigma_f}^2}\right) \tag{1}$$

The Gaussian tuning curve fits well the response of V1 or MT neurons to variables such as stimulus orientation motion direction [7], and is hence widely used in models of sensory coding [8,9]. Large values of σ_f indicate coarse coding, whereas small values of σ_f indicate sharp tuning.

Spike count responses of each neuron on each trial are assumed to follow a Poisson distribution whose mean is given by the above neuronal tuning function (Eq. 1). The Poisson model neglects all correlations between spikes. This assumption is certainly a simplification but it is sufficient to account for the majority of the information transmitted by real cortical neurons [10], and it makes our model mathematically tractable.

Neurons in sensory cortex receive a large number of inputs from other neurons with a variety of stimulus preferences. However, the majority of their inputs come from neurons with roughly similar stimulus preference [1,2,3]. To characterise mathematically this type of spread of stimulus preference, we assume that the probability distribution of the preferred stimulus among afferent neurons follows a Gaussian distribution:

$$P(\hat{\mathbf{s}}) = \frac{1}{(2\pi)^{D/2}\sigma_p^D} \exp\left(-\frac{(\hat{\mathbf{s}} - \hat{\mathbf{s}}_0)^2}{2\sigma_p^2}\right) \tag{2}$$

In Eq. (2) the parameter $\hat{\mathbf{s}}_0$ represents the the center of the distribution, thus being the most represented preferred stimulus in the population (we set, without loss of generality, $\hat{\mathrm{s}}_0 = 0$). The parameter σ_p controls the spread of stimulus preferences of the afferent neuronal population: a small value of σ_p indicates that a large fraction of the population have similar stimulus preferences, and a large value of σ_p indicates that all stimuli are represented similarly. A Gaussian distribution of stimulus preferences for the afferent population fits well empirical data on distribution of preferred orientations of synaptic inputs of neurons in both deep and superficial layers of ferret primary visual cortex [3].

3 Width of Tuning and Spread of Stimulus Preferences in Visual Cortex

In order to later compare our results with real parameters in cortex, we performed a meta-analysis by fitting our distributions, Eq. (3) and (4), to published data. For a target neuron in ferret primary visual cortex tuned to orientation, the spread of stimulus preferences σ_p of its inputs is $\approx 20^o$ for layer 5/6 neurons [3], and 16^o [3] for layer 2/3 neurons. The orientation tuning width σ_f of the cortical inputs to the V1 target neuron is that of other V1 neurons that project to it. This σ_f is 17^o for Layer 4 neurons [11], and it is similar for neurons in deep and superficial layers [3]. For Layer 4 neurons in cat visual cortex tuned to orientation, σ_p is 20^o [2] and σ_f is $\approx 17^o$. When considering a target neuron in ferret visual cortex and motion direction tuning, the spread of tuning of its inputs σ_p is $\approx 30\ ^o$ [1]. Motion direction tuning widths of macaque neurons is $\approx 28^o$, and this width is similar across species (see [7]).

The most notable finding of the above meta-analysis is that σ_p and σ_f appear to be approximately of the same size and their ratio σ_f/σ_p is distributed around 1, in the range 0.7 to 1.1 for the above data. We will use our model to understand whether this range of σ_f/σ_p corresponds to an optimal way to transmit information across a synaptic system. It is important to bear in mind that our model, Eq. (1), considers stimuli as non-periodic values. This is a good approximation because the experimental values of σ_p and σ_f are much smaller than the period of the stimulus, and therefore the periodic nature of the stimulus can be neglected to a good approximation. However, this simplification makes it difficult to compare *directly* the model results for optimal σ_p or σ_f values with the V1 experimental findings above. It is instead meaningful to compare the optimal ratio σ_f/σ_p (which is independent of the scale introduced by the period of the angular stimulus variable), and check that this optimal ratio is approximately constant across a wide range of parameter values.

4 Information Theoretic Quantification of Population Decoding

To characterise how a target neuronal system can decode the information about sensory stimuli contained in the activity of its afferent neuronal population, we use mutual information [12]. The mutual information between a set of stimuli and the neuronal responses quantifies how well any decoder can discriminate among stimuli by observing the neuronal responses, and is defined as follows:

$$I(\mathbf{S},\mathbf{R}) = \int d\mathbf{s}\, P(\mathbf{s}) \sum_{\mathbf{r}} P(\mathbf{r}|s) \log_2 \frac{P(\mathbf{r}|\mathbf{s})}{P(\mathbf{r})} \tag{3}$$

where $P(\mathbf{s})$ is the probability of stimulus occurrence (here taken for simplicity as a uniform distribution over the hypersphere of D dimensions and 'radius' s_ρ). $P(\mathbf{r}|\mathbf{s})$ is the probability of observing a neuronal population response $\mathbf{r}$ conditional to the occurrence of stimulus $\mathbf{s}$, and $P(\mathbf{r}) = \int d\mathbf{s} P(\mathbf{s}) P(\mathbf{r}|\mathbf{s})$. The probability $P(\mathbf{r}|\mathbf{s})$ is computed according to the Poisson distribution, which is entirely determined by the knowledge of the tuning curves [13]. The mutual information is difficult to compute for large populations because it requires the knowledge of the probability of the large-dimensional response vector $\mathbf{r}$ [14]. However, since in our model we assume that we have a very large number of independent neurons in the population and that the total activity of the system is of the order of its size, then we can use the following simpler (but still exact in the large N limit) expression[8]:

$$I(\mathbf{S},\mathbf{R}) = -\sum_{\mathbf{s}} P(\mathbf{s}) \log_2(P(\mathbf{s})) - \frac{D}{2} \log_2 (2\pi e) + \frac{1}{2} \int d\mathbf{s}\, P(\mathbf{s}) \log_2 (|\boldsymbol{\mathcal{J}}(\mathbf{s})|) \tag{4}$$

where $\boldsymbol{\mathcal{J}}(\mathbf{s})$ is the Fisher information matrix and $|\ldots|$ stands for the determinant. The Fisher information matrix is a $D \times D$ matrix whose elements i, j are defined as follows:

$$\mathcal{J}_{i,j}(\mathbf{s}) = -\sum_{\mathbf{r}} P(\mathbf{r}|\mathbf{s}) \left(\frac{\partial^2}{\partial s_i\, s_j} \ln P(\mathbf{r}|\mathbf{s}) \right), \tag{5}$$

The Fisher information matrix can be computed by taking into account that for a population of Poisson neurons is just the sum of the Fisher information for individual neurons, and the latter has a simple expression in terms of tuning curves [8]. Since the neuronal population size N is is large, the sum over Fisher information of individual neurons can be replaced by an integral over the stimulus preferences of the neurons in the population, weighted by their probability density $P(\hat{\mathbf{s}})$. After performing the integral over the distribution of preferred stimuli, we arrived at the following result for the elements of the Fisher information matrix:

$$\mathcal{J}_{i,j}(\mathbf{s}) = \frac{N\tau m}{\sigma_p^2} \frac{\sigma^{D-2}}{(1+\sigma^2)^{\frac{D}{2}+2}} \left(\delta_{i,j} + \sigma^2 \left(\delta_{i,j} + \xi_i \xi_j\right)\right) \exp\left(-\frac{\xi^2}{2\,(1+\sigma^2)}\right) \tag{6}$$

where we have introduced the following short-hand notation $\sigma_f/\sigma_p \rightarrow \sigma$ and $\mathbf{s}/\sigma_p \rightarrow \boldsymbol{\xi}$; $\delta_{i,j}$ stands for the Kroneker Delta. From Eq. (6) it is possible to compute explicitly the determinant $|\boldsymbol{\mathcal{J}}(\mathbf{s})|$, which has the following form:

$$|\boldsymbol{\mathcal{J}}(\mathbf{s})| = \prod_{i=1}^{D} \lambda_i = \alpha(\xi)^D (1+\sigma^2)^{D-1} \left(1 + \sigma^2(1+\xi^2)\right) \tag{7}$$

where $\alpha(\xi)$ is given by:

$$\alpha(\xi) = \frac{N\tau m}{\sigma_p^2} \frac{\sigma^{D-2}}{\left(1+\sigma^2\right)^{\frac{D}{2}+1}} \exp\left(-\frac{\xi^2}{2\left(1+\sigma^2\right)}\right) \tag{8}$$

Inserting Eq. (7) into Eq. (4), one obtains a tractable expression for $I(\mathbf{S},\mathbf{R})$, whose behaviour will be investigated next.

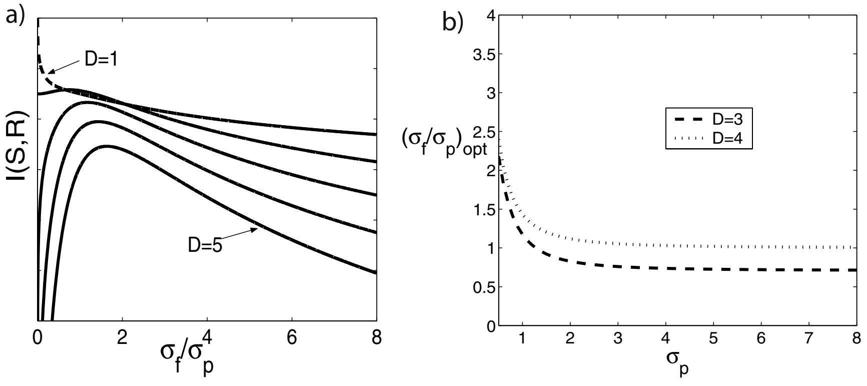


Fig. 1. a)Mutual information as a function of the ratio σ_f/σ_p. The curves for each stimulus dimensionality D were shifted by a constant factor to separate them for visual inspection (lower curves correspond to higher values of D). The y-axis is thus in arbitrary units. Parameters are as follows: $\mathbf{s}_\rho = 2$, $r_{max} = 50$Hz, $\tau = 10$ms. b)Plot of the optimal ratio σ_f/σ_p, for which there is maximal information transmission, as a function of σ_p and for $D = 3$, and 4. For a wide range of stimulus spread values, σ_f/σ_p is within the range found in cortex.

We have studied numerically the dependence of the mutual information on the parameters σ_f and σ_p as a function of the number of encoded stimulus features D [1]. We investigated this by fixing σ_p and then varying the ratio σ_f/σ_p over a wide range. As noted above, since we neglect the angular nature of the experimental visual stimuli, we can only meaningfully compare the ratio σ_f/σ_p (but not their individual values) to the real V1 data.

Results (obtained for $\sigma_p = 1$) are reported in Fig. 1.a. Unlike the case of a uniform distribution of stimulus preferences [6], there is a finite value of the width of tuning σ_f that maximizes the information for all $D \geq 2$. For $D \geq 2$ the range $0.7 \leq \sigma_f/\sigma_p \leq 1.1$ found in visual cortex either contains the maximum or corresponds to near optimal values of information transmission. For $D = 1$, information is maximal for very narrow tuning curves. However, also in this case the information values are still efficient in the cortical range $\sigma_f/\sigma_p \approx 1$,

[1] We found (data not shown) that other parameters such as m and τ, had a weak or null effect on the optimal configuration; see [9] for a $D = 1$ example in a different context.

in that the tail of the $D = 1$ information curve is avoided in that region. In Fig. 1b we show the optimal ratio σ_f/σ_p, that is the ratio for which there mutual information is maximal, for stimulus dimension $D = 3$, and 4 as a function of σ_p. Except when σ_p is very small the optimal ratio between σ_f and σ_p is within the range found in cortex, for a wide range of σ_p values. Thus, the range of values of σ_f and σ_p found in visual cortex allows efficient synaptic information transmission of a small number of stimulus features encoded by the neurons.

In conclusion, we have shown that the stimulus-specific structure of cortico-cortical connections may have also implications for understanding cortico-cortical information transmission. Our results suggest that, whatever the exact role of cortico-cortical synapses in generating orientation tuning, their wiring allows efficient transmission of sensory information.

Acknowledgments

Research supported the Human Frontier Science Program, Royal Society and Wellcome Trust 066372/Z/01/Z.

References

1. Roerig, B., Kao, J.P.Y.: Organization of intracortical circuits in relation to direction preference maps in ferret visual cortex. J. Neurosci. **19** (1999) RC44(105)
2. Yousef, T., Bonhoeffer, T., Kim, D.S., Eysel, U.T., Tóth, É., Kisvárday, Z.F.: Orientation topography of layer 4 lateral networks revealed by optical imaging in cat visual cortex (area 18). European J. Neurosci. **11** (1999) 4291–4308
3. Roerig, B., Chen, B.: Relations of local inhibitory and excitatory circuits to orientation preference maps in ferret visual cortex. Cerebral Cortex **12** (2002) 187–198
4. Martin, K.A.C.: Microcircuits in visual cortex. Current Opinion in Neurobiology **12** (2002) 418–425
5. Montemurro, M.A., Panzeri, S.: Optimal information decoding from neuronal populations with specific stimulus selectivity. In Saul, L.K., Weiss, Y., Bottou, L., eds.: Advances in Neural Information Processing Systems 17. MIT Press, Cambridge, MA (2005)
6. Zhang, K., Sejnowski, T.: Neuronal tuning: to sharpen or to broaden? Neural Computation **11** (1999) 75–84
7. Albright, T.D.: Direction and orientation selectivity of neurons in visual area MT of the macaque. J. Neurophysiol. **52** (1984) 1106–1130
8. Brunel, N., Nadal, J.P.: Mutual information, fisher information and population coding. Neural Computation **10** (1998) 1731–1757
9. Nevado, A., Young, M., Panzeri, S.: Functional imaging and neural information coding. Neuroimage **21** (2004) 1095–1095
10. Petersen, R.S., Panzeri, S., Diamond, M.: Population coding of stimulus location in rat somatosensory cortex. Neuron **32** (2001) 503–514
11. Usrey, W.M., Sceniak, M.P., Chapman, B.: Receptive fields and response properties of neurons in layer 4 of ferret visual cortex. J. Neurophysiol. **89** (2003) 1003–1015
12. Cover, T., Thomas, J.: Elements of information theory. John Wiley (1991)
13. Dayan, P., Abbott, L.F.: Theoretical Neuroscience. MIT Press (2001)
14. Panzeri, S., Treves, A.: Analytical estimates of limited sampling biases in different information measures. Network **7** (1996) 87–107

Ensemble of SVMs for Improving Brain Computer Interface P300 Speller Performances*

A. Rakotomamonjy, V. Guigue, G. Mallet, and V. Alvarado

P.S.I CNRS FRE 2645 INSA de Rouen,
Avenue de l'université,
76801 Saint Etienne du Rouvray, France
alain.rakotomamonjy@insa-rouen.fr

Abstract. This paper addresses the problem of signal responses variability within a single subject in P300 speller Brain-Computer Interfaces. We propose here a method to cope with these variabilities by considering a single learner for each acquisition session. Each learner consists of a channel selection procedure and a classifier. Our algorithm has been benchmarked with the data and the results of the BCI 2003 competition dataset and we clearly show that our approach yields to state-of-the art results.

1 Introduction

Some people who suffer some neurological diseases can be highly paralyzed due to the fact that they do not have anymore control on their muscles. Therefore, their only way to communicate is by using their electroencephalogram signals. Brain-Computer interfaces (BCI) research aim at developing systems that help those disabled people communicating with machines.

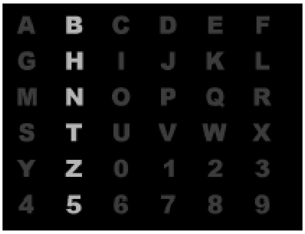
Research on BCI is a fast growing field and several EEG-based techniques have been proposed for realizing BCI. The BCI framework that is of interest for us is based on Event Related Potentials (ERP) which appear in response to some specific stimuli. Hence, this BCI produces an appropriate stimuli for which the patient is expected to respond. The core principle underlying such system is then based on the recognition of ERP which corresponds to the stimuli. In other words, this BCI is essentially based on classification of a EEG signals which is a difficult problem due to the low signal-to-noise ratio and the variability of the ERP responses within a single subject.

We propose some solutions for improving the performance of such BCI by addressing this variability problem through an ensemble approach based on linear SVMs [6].

The paper is structured as follows : section 2 describes the BCI classification problem with more details and the methodology that have been used for channel selection and for building the multiple classifier systems. Section 3 presents the

* This work was supported by grants from the IST programme of the European Community under the PASCAL Network of excellence, IST-2002-506778.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 45–50, 2005.



A	B	C	D	E	F
G	H	I	J	K	L
M	N	O	P	Q	R
S	T	U	V	W	X
Y	Z	0	1	2	3
4	5	6	7	8	9

Fig. 1. P300 speller matrix with an highlighted column

results that have been achieved whereas section 4 concludes the paper with comments and perspectives on the work.

2 Methods

The BCI problem we are addressing in this paper concerns the P300 speller. This speller has been introduced by Farwell and Donchin [3] who developed a protocol whereby a subject is presented a 6 × 6 characters matrix in which a row or column is randomly intensified. Then large P300 evoked potentials can be recorded in response to the intensification of a desired character. Hence the objective of the problem is to classify these potentials whether they correspond or not to the desired character . The data that we used in this study comes from the BCI 2003 competition dataset [1] and they have been provided by the Wadsworth institute [5]. In the following, we give a short description of the datasets. The data corresponds to the three separate spelling sessions of respectively 5,6, 8 words by a single subject. These recordings correspond to 64 channels which have been digitized at 240 Hz. However, in our case, we are only interested in the part of the signal that follows the intensification of a row or column. The experimental protocol, which have already been described in [3] is the following. For selecting a given character, each row and column of the matrix is highlighted in a random sequence (hence, the desired character appears on 2 out of 12 intensifications) and this procedure is repeated 15 times for the same character. Then after a short pause, the user has to focus on another character. In our case, the objective is to predict a word correctly by means of the fewer sequence repetitions as possible.

Brain Computer Interfaces classification problems are challenging essentially due to (i) the low signal-to-noise ratio of the signal, (ii) the variability of the EEG signal for a given user, and (iii) the variability between different users. Then, in order to achieve interesting results, it is expected that a classification strategy addresses all these points.

The problem we face is thus the following. We have $\{x_i, y_i\}_{i=1,\cdots,\ell}$ examples where each $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$. In our case, $d = 64 \times 240 \times t_d$ where t_d corresponds to the duration of the signal of interest after intensification. Since we have fixed t_d to $0.667s$, the number of features is equal to 10240 which suggests that a variable selection procedure would be helpful for a least, reducing the processing time.

According to the competition rules, the first two spelling sessions have been used for training our system while the last one has been kept for test. Hence, our training data is composed of 7560 stimuli responses which 1260 of them contain a true P300 ERP responses.

Signals from channels are. Here, we have filtered each signal using a 8-order bandpass filter with cut-off frequencies of 0.1 Hz and 20 Hz and then decimated each signal according to this latter frequency. Since we are interested only in a time window of 667 *ms* after intensification, at this stage the data dimension is 896. At this point, one can directly use the signals from preselected or user-defined channels as inputs of a classifier. This approach has been investigated by Meinicke et al. After appropriate scaling of the inputs, they trained a SVM with equal number of positive and negative examples. In this work, we propose an approach that tries to take into account the variability of the signals during different sessions or even during different words spelling in the same session.

The idea is to train a complete recognition stage for each word in the training set and then to combine the output of all the resulting classifiers for producing a single score for each signal.

Hence for training a single classifier, we have performed the following steps for each word spelling session signals:

- signals of each channel are normalized to zero mean and unit variance
- a channel selection is performed in order to retain only the most relevant channels. Our channel selection algorithm is based on a recursive channel elimination. The algorithm starts with all channels and then according to a user defined performance criterion C, the criterion $C^{(-j)}$ (the criterion when channel j is removed) is evaluated on all the remaining words of the training session. In our case, our criterion is defined as $C = \frac{tp}{tp+fp+fn}$ where tp, fp, fn are respectively the number of true positive, false positive and false negative. The channel that is suppressed is thus the one which removal maximizes the criterion C. Then we continue this procedure until all channels have been removed.

  Hence, this channel selection algorithm allows us to rank all the channels according to the criterion C and only the most relevant channels are subsequently used for classification.
- an linear SVM is then trained using all available examples described by the selected channels for this single word.

In this BCI problem, one should distinguish an evoked potential response classification (as described above) and a character recognition. Remember that the character that is spelled is characterized by a row and column of the matrix, and thus a character recognition is achieved by recognizing as positive a given row and column. Hence, for a given sequence of all rows and columns matrix illumination (which corresponds to 12 intensifications), if $f(x)$ is the score of a given row or columns rc given by our classifier then $S_{rc}(k) = S_{rc}(k-1) + f(x)$

where x is the input signal associated to the highlighted row or columns rc, $f(x)$ is the score (SVMs output) associated to x and $S_{rc}(k)$ is overall score of rc at sequence k. Then at a given sequence k, the character that should be recognized is the one with maximal row and column score. In our approach, we are using an ensemble approach since we have a classifier for each word of the training session, the score $f(x)$ is actually : $f(x) = \sum_{i=1}^{n} f_i(x)$ where $f_i(x)$ is the score given by each classifier to x.

3 Results

Several experiments have been carried out in order to analyze the benefits of our approach.

All SVM classifiers that we used are linear SVM classifier. We justify this choice by stating that what makes this problem (and many other biosignal classification problems) difficult is the variability of the datasets. Thus, we believe that dealing with a linear classifier will prevent from overfitting the training data.

Our results are compared to those obtained by BCI competition winner. We have reported in the table (1), the results obtained by Bostanov [2] and a result from a similar algorithm than the one described by Meinicke et al. [4]. In this experiment, we have used the 10 channel proposed by the authors and a linear SVM (instead of a gaussian kernel SVM) for which the C regularization parameter is 0.1. We have also evaluated the gain of using a multiple classifiers system in which each classifier is dedicated to a single word learning. As expected, taking into account the variability of the P300 within subject by multiplying the number of classifiers greatly enhances the recognition rate performances. From table (1), one can see that using all channels achieves a state of the art result since all words are correctly recognized with only four sequences. This is very interesting since it confirms our hypothesis that variability of responses play a very important role in the P300 recognition, and it is necessary to cope with this variability. Again, if we compare performance of a single SVMs and a mixture of SVMs approach with the 10 channels used by Meinicke et al., we can see that our approach only slightly improves performances. This latter point highlights the need of an appropriate channel selection which, again should take into account the problem variability.

We have analyzed the channel selection procedure. What we expect from the channel selection is two-fold : a reduced number of explicative channels and an increased processing speed, which will be useful in a real-time application context. First of all, we have analyzed the variability of the channel selection procedure for a single subject using the P300 speller within different sessions and runs. Table (2) shows the 10 top ranked channels for a given session according to the above described criterion $tpr/(tpr + fpr + fnr)$. The first remark that can be drawn from this table is that only few channels (55, 56, 58, 60 which are also known as P8 ,Po7 ,POz ,Po8) are considered as equally relevant for almost all the sessions. This point focuses again the variability of the data and justifies the

Table 1. Number of mispellings in the test words with respects to the number of sequences and the algorithm

	Nb. of sequences							
Algorithms	1	2	3	4	5	6	7	10
Bostanov [2]	11	5	2	1	1	0	0	0
10 preselected channels and single SVM	14	6	6	0	1	0	0	0
all channels and single SVM	14	10	9	5	5	5	1	0
10 preselected channels and 1 SVM per word	13	8	3	1	2	0	0	0
all channels and 1 SVM per word	7	4	3	0	0	0	0	0
4 relevant channels and 1 SVM per word	8	7	4	0	1	0	0	0
10 relevant channels and 1 SVM per word	8	5	5	1	0	1	0	0
26 relevant channels and 1 SVM per word	4	2	0	0	0	0	0	0
30 relevant channels and 1 SVM per word	5	3	0	0	0	0	0	0
optimal relevant channels and 1 SVM per word	4	2	1	0	0	0	0	0

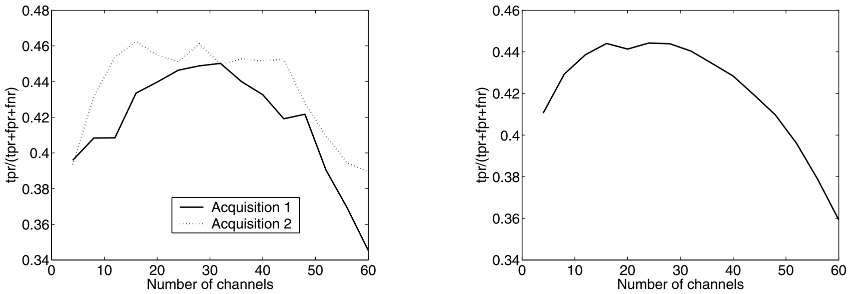


Fig. 2. Examples of channel selection criterion variation with respect to the number of channels. On the left, we have two examples for each training word and on the right we have plotted the averaged criterion value over all the training words.

need of taking into account separately the different word spelling acquisition. If we compare the channels top-ranked by our channel selection algorithm to those used by the other competitors of on this datasets [4], we can see that only few channels are in common. Figure (2) describes the variation of the channel selection criterion for different words spelling sessions. Again, we can see that the optimal number of channels differs considerably from a session to another and on average, the optimal number of channels is between 15 and 30.

Regarding the spelling performance given in table (1), two interesting points have to be highlighted. First of all, one can see that the if the number of relevant channels to be used is chosen appropriately, then all the words in the test set can be recognized correctly with only 3 sequences. However, our fully automated procedure (last line of the table) need 4 sequences for achieving equivalent performance. The second interesting point is that within 15 and 30 used channels, the overall performance is rather stable with a number of misspellings and number of needed sequences varying respectively from 6 to 10 and 2 to 3.

Table 2. 10 Top Ranked channels for the differents word spelling sessions

Sessions	10 Top Ranked Channels
1	9 15 18 36 40 55 56 59 63 64
2	18 39 53 55 56 58 59 60 61 64
3	9 18 40 48 53 55 56 58 61 64
4	10 18 33 42 46 55 56 58 60 64
5	16 22 39 40 50 56 57 60 61 62
6	2 10 36 42 48 50 55 56 58 60
7	10 17 21 25 31 43 46 51 55 56
8	10 32 41 44 49 52 55 56 60 61
9	10 23 42 48 55 56 58 60 62 63
10	4 10 17 41 42 49 55 56 58 64
11	13 34 41 48 55 56 58 60 62 64

4 Conclusions and Perspectives

We have described in this work a methodology for classifying event related potentials for a Brain-Computer Interface. The strength of our approach is based on an ensemble SVMs which allows us to deal with the variability of EEG responses. In this context, we have trained a SVM classifier and selected the most relevant channels associated to each word spelling sessions signal. This method yields to state-of-the-art results since with a fully automated procedure, we were able to correctly recognized all words with only 4 sessions. Our future works now will deal with the variability of EEG responses within different subjects.

References

1. B. Blankertz, K-R Mller, G. Curio, T. Vaughan, G. Schalk, J. Wolpaw, A. Schlgl, C. Neuper, G. Pfurtscheller, T. Hinterberger, M. Schrder, and N. Birbaumer. the BCI competition 2003: Progress and perspectives in detection and discrimination of eeg single trials. *IEEE Trans. Biomed. Eng*, 51(6):1044–1051, 2004.
2. V. Bostanov. BCI competition 2003-data sets Ib and IIb: feature extraction from event-related brain potentials with the continuous wavelet transform and the t-value scalogram. *IEEE Transactions on Biomedical Engineering*, 51(6):1057–1061, 2004.
3. L. Farwell and E. Donchin. Talking off the top of your head : toward a mental prosthesis utilizing event-related brain potentials. *Electroencephalography and Clinical Neurophysiology*, 70(6):510–523, 1988.
4. P. Meinicke, M. Kaper, F. Hoppe, M. Heumann, and H. Ritter. Improving transfer rates in brain computer interfacing : A case study. In S. Thrun In S. Becker and editors K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, volume 15, pages 1107–1114, 2003.
5. G. Schalk, D. McFarland, T. Hinterberger, N. Birbaumer, and J. Wolpaw. BCI2000 : a general-purpose brain-computer interface (BCI) system. *IEEE Transactions on Biomedical Engineering*, 51(6):1034–1043, 2004.
6. B. Scholkopf and A. Smola. *Leaning with Kernels*. MIT Press, 2001.

Modelling Path Integrator Recalibration Using Hippocampal Place Cells

T. Strösslin, R. Chavarriaga, D. Sheynikhovich, and W. Gerstner

École Polytechnique Fédérale de Lausanne (EPFL),
Brain-Mind Institute and School of Computer and Communication Sciences,
CH-1015 Lausanne, Switzerland
thomas.strosslin@a3.epfl.ch, {ricardo.chavarriaga,
denis.sheynikhovich, wulfram.gerstner}@epfl.ch

Abstract. The firing activities of place cells in the rat hippocampus exhibit strong correlations to the animal's location. External (e.g. visual) as well as internal (proprioceptive and vestibular) sensory information take part in controlling hippocampal place fields. Previously it has been observed that when rats shuttle between a movable origin and a fixed target the hippocampus encodes position in two different frames of reference. This paper presents a new model of hippocampal place cells that explains place coding in multiple reference frames by continuous interaction between visual and self-motion information. The model is tested using a simulated mobile robot in a real-world experimental paradigm.

1 Introduction

Place cells in the rat hippocampus are active only when the rat is in a specific region within an environment. This location-sensitive firing is influenced by both *internal* (e.g. self-motion) and *external* (e.g. visual) sensory inputs [1]. Electrophysiological studies reveal that the firing pattern of a place cell is sensitive to the position of visual landmarks placed around [2], but also within [3] the environment. However, using *path integration* (PI) [4], the animal is also capable of returning to the starting point of a journey based on internal cues only (i.e. homing). In this case no external cues are available and place cell activity depends only on PI [5]. Finally, behavioural experiments with rodents indicate that PI can be recalibrated using visual information [4].

In order to investigate how external sensory input and internal information control the location-specific activity of the hippocampal place cells, Gothard *et al.* [6] propose an experimental paradigm where rats alternate between a movable box at one end of a linear track and a fixed reward site at the other end. Depending on the type of information the animal uses to update its spatial representation, place cells activity can be aligned to the movable box or to the fixed visual cues. Their recordings show that in the initial part of the journey, place cells fire at fixed distances relative to the point of departure (box for outbound, fixed site for inbound), whereas towards the end of the journey, cells are aligned with the destination (fixed site for outbound, box for inbound). They conclude

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 51–56, 2005.

that the spatial representation is initially driven by PI, and, as the rat moves farther along the track, it becomes tied to the external cues.

This paper proposes a new neural model of the rat hippocampus. A representation of space is built by combining visual sensory input and self-motion information. The model resolves ambiguities in the visual data by means of path integration, whereas external input is used to prevent the accumulation of errors inherent to the PI. The interaction between the two sources of information is evaluated in the experimental paradigm described above using a simulated mobile robot. The firing profiles of modelled place cells exhibit properties similar to real hippocampal neurons.

2 Model Description

The model architecture (Fig. 1) is based on the anatomy of the rat hippocampal formation. It is consistent with fundamental electro-physiological properties of place cells [1]. This work extends previous models [7,8,9] by equipping them with a new visual system that can deal with realistic sensory input and an adaptive recalibration mechanism used to combine path integration and visual input.

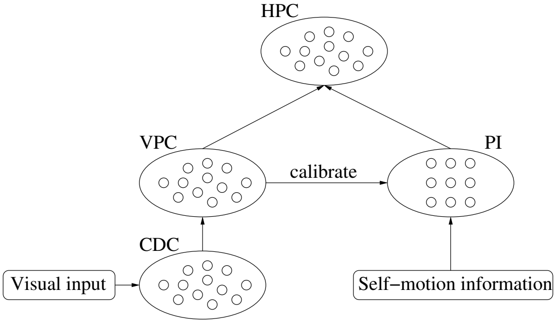


Fig. 1. Model architecture. It consists of four interconnected populations of rate-coded neurons. Column difference cells (CDC) store visual stimuli and drive visual place cells (VPC). Self-motion information drives the path integrator (PI). VPC calibrate PI and they both project to the combined hippocampal place cells (HPC).

2.1 Visual Place Code

The model's visual processing is based on low-level feature matching, rather than explicit object recognition. Complex Gabor wavelets with 8 different orientations serve as feature extractors. They are evaluated at all points of a rectangular grid. An example response of this "artificial retina" is shown in Fig. 2.

Each retinal response is translated into neural activity. During an experiment, cells are "recruited" as needed. Thus the number of cells grows with time. It is assumed that there are enough cells to represent the entire environment.

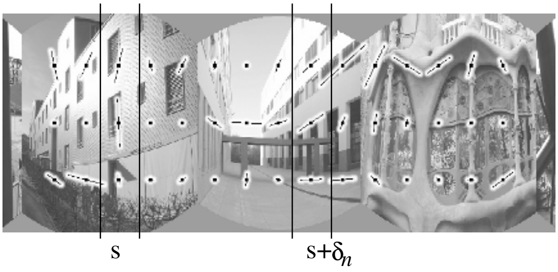


Fig. 2. Responses of an artificial 15×3 retina of Gabor filters to an input image of 280° horizontal view field. Each point of the grid contains 8 filters of different orientations. The thick lines indicate the direction and "strength" of edges near each retinal point. Two retinal columns at positions s and $s + \delta_n$ are highlighted.

During environment exploration, a set of "column difference cells" (CDCs) is recruited at each time step. CDC n stores the difference $\boldsymbol{d}_n = \boldsymbol{f}(s_n + \delta_n) - \boldsymbol{f}(s_n)$ between two retinal columns s and $s + \delta$, where $\boldsymbol{f}(s_n)$ is the vector of all filter activities at column s (Fig. 2). At a later time step, CDC n responds to the new input with a firing rate

$$r_n = k \cdot \exp\left\{-\min_s \left[(\boldsymbol{f}(s + \delta_n) - \boldsymbol{f}(s)) - \boldsymbol{d}_n\right]^2\right\} \quad , \tag{1}$$

where k is a normalisation constant. Spatial firing is obtained by combining the responses of several CDCs one synapse downstream in a population of visual place cells (VPCs). One-shot Hebbian learning is applied to tune the synaptic strengths w_{ij} between each active CDC j and a newly recruited VPC i to $w_{ij} = r_j \cdot r_i$. The new cell should be maximally active ($r_i = 1$) for the current afferent CDC projection. This is achieved by using a piecewise linear activation function:

$$r_i = \begin{cases} 0 & \text{if } \kappa_i h_i < \theta_{low} \\ 1 & \text{if } \kappa_i h_i > 1 \\ (\kappa_i h_i - \theta_{low})(1 - \theta_{low}) & \text{otherwise} \end{cases} \tag{2}$$

where $h_i = \sum_j w_{ij} r_j$ is the input potential of the VPC neuron i, $\kappa_i = 1/h_i^0$ determines the saturation potential of the neuron (with h_i^0 standing for the input potential at the time when neuron i was recruited) and $\theta_{low} = 0.2$ is the minimal input to activate the neuron.

The resulting place code represents the robot's position $\mathbf{P_v}$ within the environment, estimated by visual information only. The encoded location is extracted from the population activity using a population vector:

$$\mathbf{P_v} = \frac{\sum_i r_i \cdot \mathbf{x}_i}{\sum_i r_i} \quad , \tag{3}$$

where $\mathbf{x}_i$ is the position of the robot where VPC i was recruited.

2.2 Path Integration

The input to the path integrator are the rotation and displacement signals from the robot's odometers. After each movement a new estimated position of the robot in an abstract Cartesian coordinate frame is calculated using standard trigonometric formulas. In order to neurally represent the position we employ a population of "path integration cells" (PI) such that each cell j is assigned a preferred position $\boldsymbol{p}_j$ in the abstract frame. Firing rate of the cell is defined as:

$$r_j = e^{-\frac{(\|\mathbf{P_o} - \mathbf{p}_j\|_2)^2}{2\sigma_o^2}} \quad , \tag{4}$$

where $\boldsymbol{P_o}$ is the internal odometric position estimate.

In order to decrease the mismatch between the estimated positions $\mathbf{P_o}$ and $\mathbf{P_v}$, the path integrator is recalibrated using vision at each time step:

$$\mathbf{P} = \mathbf{P_o} - \beta \cdot (\mathbf{P_o} - \mathbf{P_v}) \ , \tag{5}$$

where $\beta = 0.1$ determines the influence of the visual cues.

2.3 Hippocampal Place Cells

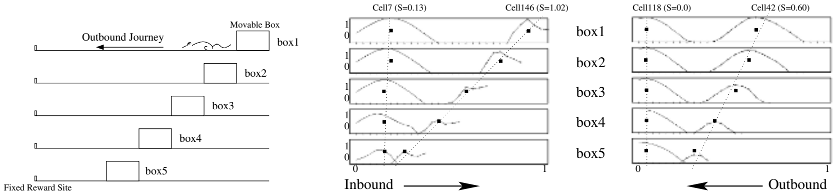
VPC and PI place cells project to a layer of hippocampal place cells (HPCs) (Fig.1). At each time step a place cell is recruited and its afferent connections from the VPC and PI are initialised using one-shot Hebbian rule. The firing rate of HPC neuron i is defined by (2) where the afferent cells are the PI and VPC.

3 Results and Conclusions

Gothard et al. [6] proposed an experimental paradigm to study how path integration and visual input contribute to the hippocampal representation of space. Rats were trained to shuttle back and forth on a linear track with a movable box located at one end of the track and a fixed reward site at the other (box1 configuration, Fig. 3(a)). During the journeys from the box to the fixed site (outbound journey), the box was moved randomly between five locations (box1 to box5). Once the animal reached the fixed site, it started the inbound journey to the box (now located at a new position). Cell recordings show that in the initial part of the journey place cells fired at fixed distances relative to the point of departure (box for outbound, fixed site for inbound), whereas towards the end of the journey cells were aligned with the destination (fixed site for outbound, box for inbound) [6].

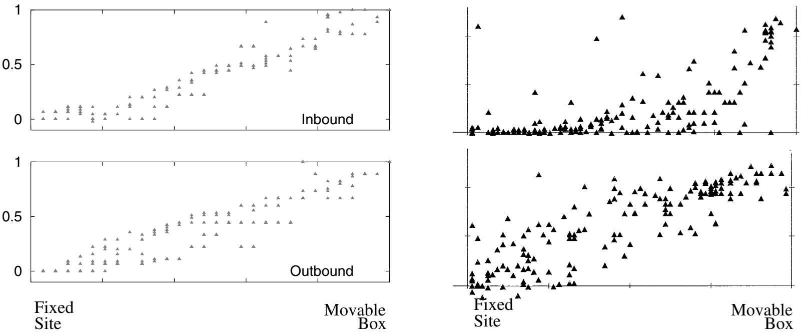
We apply the same experimental setup for our model. The place fields of four HPCs for the five box configurations are shown in Fig. 3(b). Consistent with observation in rats, HPCs initially fire with respect to the starting point, whereas towards the end of the journey, place fields align with the destination.

This change of reference frame in the model is explained by the interaction between internal and external information: in the inbound and outbound journeys visual information recalibrates the path integrator (PI) to either end of the



(a) (b)

Fig. 3. (a) Experimental setup of Gothard et al. [6]. The rat shuttles between a fixed site and a box, which is displaced during outbound journeys to five different locations (box1 to box5). (b) Firing profiles of four modelled place cells. Two cells are active during the inbound journey (left), and two during the outbound journey (right) for the five box configurations. Black dots show the place field displacements with respect to box1 condition, lines approximate the displacement slopes S (see text).



(a) (b)

Fig. 4. Displacement slopes of place cells versus the location of maximum firing in the box1 configuration. Slopes are normalised to be 0 for cells whose place fields do not shift following the box shift and 1 for cells whose place fields shift together with the box. (a) Our model. (b) Experimental results in rats (Redrawn from [6]).

well known box1 configuration. After leaving the box in the outbound journey, a mismatch occurs between vision and PI if the configuration differs from box1. This inconsistency is gradually reduced by recalibrating PI (5), until the representations are congruent near the end of the track. Later, during the inbound journey, another mismatch appears and again, PI recalibration by vision resolves the conflicting information. To quantify how the receptive fields of the place cells shift for the different configurations (box1 to box5) we calculate their *displacement slopes* [6]. This slope results from a linear fit of the place field shifts of a cell in box2 to box5 with respect to box1. Shifted positions are determined by the location of the maximum cross-correlation of the place field with respect to the box1 condition. The displacement slopes in the HPC population are shown in Fig. 4(a). Both for inbound and outbound directions, cells firing near the fixed end in the box1 condition do not exhibit a shift in their receptive fields whereas neurons which fire close to the box shift along with the box. These results are similar to animal experiments (Fig. 4(b)). However, in the outbound case, the distribution of displacement slopes differs from [6]. One possible explanation is that this distribution depends on the environment. In particular, the size of the box may influence its relevance when the rat visually localises itself.

This paper presents a model of hippocampal place cells based on interacting visual and self-motion sensory input. In contrast to previous models, this proposal is based on a visual system which uses low-level feature matching instead of abstract landmark detection. It is thus capable of working with realistic visual input. The model is able to build a stable place code. Moreover, it reproduces changes in this representation in a conflict situation as the one described above [6]. The receptive fields encode the agent's position with respect to two reference frames: Initially, the place code is aligned to internal coordinates given by path integration. After some time, the representation systematically shifts to an external reference frame given by visual cues. This supports the idea of a competition between the different sources of information in order to keep a consistent representation of space.

Acknowledgements

This work was supported by the Swiss National Science Foundation under grant no. 200020-100265/1.

References

1. Redish, A.D.: Beyond the Cognitive Map, From Place Cells to Episodic Memory. MIT Press-Bradford Books, London (1999)
2. Muller, R.U., Kubie, J.L.: The effects of changes in the environment on the spatial firing of hippocampal complex-spike cells. Journal of Neuroscience **7** (1987) 1951–1968
3. Gothard, K.M., Skaggs, W.E., Moore, K.M., McNaughton, B.L.: Binding of hippocampal CA1 neural activity to multiple reference frames in a landmark-based navigation task. Journal of Neuroscience **16(2)** (1996) 823–835
4. Etienne, A.S., Jeffery, K.J.: Path integration in mammals. Hippocampus (2004)
5. Save, E., Nerad, L., Poucet, B.: Contribution of multiple sensory information to place field stability in hippocampal place cells. Hippocampus **10** (2000) 64–76
6. Gothard, K.M., Skaggs, W.E., McNaughton, B.L.: Dynamics of mismatch correction in the hippocampal ensemble code for space: Interaction between path integration and environmental cues. Journal of Neuroscience **16(24)** (1996) 8027–8040
7. Arleo, A., Gerstner, W.: Spatial cognition and neuro-mimetic navigation: A model of hippocampal place cell activity. Biological Cybernetics, Special Issue on Navigation in Biological and Artificial Systems **83** (2000) 287–299
8. Strösslin, T., Krebser, C., Arleo, A., Gerstner, W.: Combining multimodal sensory input for spatial learning. In Dorronsoro, J.R., ed.: Artificial Neural Networks - ICANN'02, Springer LNCS (2002) 87–92
9. Chavarriaga, R., Gerstner, W.: Combining visual and proprioceptive information in a model of spatial learning and navigation. In: Proceedings 2004 International Joint Conference on Neural Networks (IJCNN'2004), IEEE Press (2004) 603–608

Coding of Objects in Low-Level Visual Cortical Areas

N.R. Taylor, M. Hartley, and J.G. Taylor

Department of Mathematics, King's College, Strand, London WC2R 2LS, UK
{ntaylor, mhartley}@mth.kcl.ac.uk, john.g.taylor@kcl.ac.uk

Abstract. We develop a neural network architecture to help model the creation of visual temporal object representations. We take visual input to be hard-wired up to and including V1 (as an orientation-filtering system). We then develop architectures for afferents to V2 and thence to V4, both of which are trained by a causal Hebbian law. We use an incremental approach, using sequences of increasingly complex stimuli at an increasing level of the hierarchy. The V2 representations are shown to encode angles, and V4 is found sensitive to shapes embedded in figures. These results are compared to recent experimental data, supporting the incremental training scheme and associated architecture.

1 Introduction

The problem of creating object representations by learning is basic in building a system able to learn new concepts in new environments. We present initial results on the coding we obtain using a mixture of hard-wired and trained hierarchically-connected modules, the former at the lowest level (V1), the latter at higher levels (V2 and above). Such a split is feasible since V1 appears to act as a template for low-level features in scenes. At higher levels in the processing hierarchy we must be prepared for ever more environmentally-sensitive templates, so learning must be present.

There have been many suggested models of V1, both hard-wired and learnt. A sequence of increasingly detailed hard-wired models has been proposed for V1, fitting experimental features of single cell responses on orientation and temporal sensitivity [1, 2, 3]. Not much modeling has been performed in relation to experimental data for V2 or beyond, in spite of the data now becoming available [4]. Those models available for higher level visual cortical areas are in the main hard-wired versions [5,6,7]. One counterexample is the Neocognitron model [8], built by incremental learning of a hierarchy of neural modules. However this has been used only for digit stimulus recognition and similar problems, not related to any specific brain areas at intermediate module levels. There are also a number of models of attention using a hierarchy of visual modules, with training of the feed-forward connections from one layer to the next [9, 10]. However a number of these use pre-set object codes at the temporal lobe level, nor do any relate to known data on V2 or V4 [4, 11, 12].

We present our basic architecture in section 2, specifying the parameters defining the feed-forward connections of the excitatory population of neurons in one module to the inhibitory and excitatory neurons of the next higher module. The results for the training from V1 to V2 are presented in section 3, and that of V4 training in section 4.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 57 – 63, 2005.

2 Processing Architecture

We use a feed-forward hierarchical architecture based on a simplified version of known anatomical knowledge [13]; its details are given in figure 1. The model consists of a ventral and dorsal visual stream. In the ventral stream V1 is responsible for orientation sensitivity whilst retaining topography. We take V1 to have hard-wired afferents, and act as a bank of orientation filters, using 4 sets of orientations (the connectivity is such that V1 nodes are preferentially activated by length 5 bars). The internal connectivity of the V1 layers along with LGN efferents serve to generate a reduced representation of the the original bar, to a strong central length 3 representation with some weak surrounding activation. Input to V2 is then made up of a composite V1 formed by combining the four ventral V1 regions such that there is a pin-wheel structure for the 4 orientations, which is repeated topographically. The connectivity from V1e ->V2e has a Gaussian spread that is thresholded and has a very small fall-off with distance this allows for highest activation in V2e neurons that receive input from both components of the stimuli, i.e. those neurons that are at the overlap of the V1e representations. V4 receives input from the ventral route V1 and V2, as well as the dorsal route from lateral intra-parietal areas (LIP). The major route for topographic information is via the fast dorsal route; this route is completely hard-wired such that topographic information which at V1 level is reasonably coarse is refined by V5 and finally by LIP. The dorsal route is used to provide low-level activation of V4 neurons, priming V4 for the orientation information via the slower ventral route.

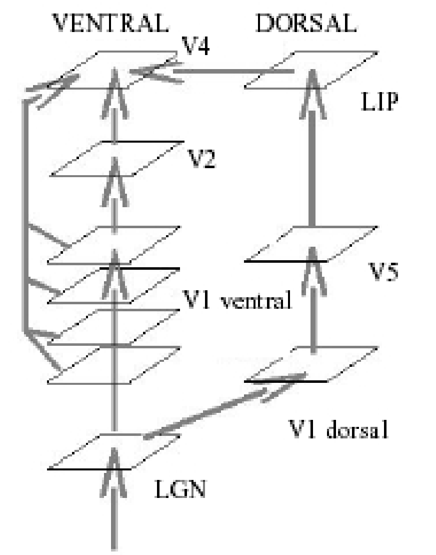


Fig. 1. The overall architecture of the model. The four ventral route V1 layers are shown, each preferentially active for one type of oriented bar.

The neurons form two populations, excitatory and inhibitory. Only the excitatory neurons send afferents to the next layer, and in the case of V1 afferents to next but one higher layers. The detailed nature of these afferents are shown in figure 2 for the particular case of V1 to V2. Lateral connections are modeled only in the excitatory population of each layer (Gaussian spread of 2 neurons with no self-excitation); from the excitatory population to the inhibitory population there is a Gaussian spread of 2 neurons; feedback from the inhibitory to excitatory neurons has a central strong spread with radius 8. Hence the inhibitory spread, being larger than the excitatory, prevents runaway activation.

The model has 14*14 neurons in each excitatory and inhibitory population for LGN, dorsal V1, V5, LIP and V4; ventral V1 populations are each composed of

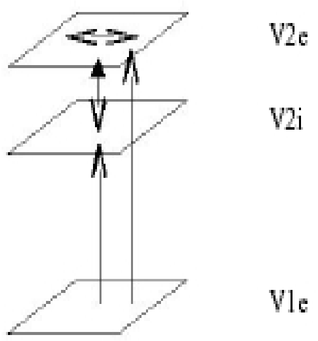


Fig. 2. The internal structure of V2, composed of an excitatory layer (V2e) and an inhibitory layer (V2i) and connectivity from V1, all regions except for LGN have the same 2-layered structure. Excitatory weights indicated by open arrow-heads, and inhibitory by closed arrow-heads.

14*14 nodes giving a total for the composite ventral V1 populations of 28*28 neurons, the same that compose V2.

The neurons used are leaky integrate and fire, with either only excitatory or inhibitory effects on other neurons according to the population of excitatory or inhibitory neurons to which they belong. All neurons have a spiking threshold of -52mV, a reset value of -59mV, a resting potential of -70mV, shunting inhibition value of -80mV.

The learning rule used is that of Spike Time Dependent Potentiation or STDP [14,15]. STDP is used to adapt V1 to V2e connections, V1 to V4e connections and V2 to V4e connections (these weights are initially 4 times greater than V1 to V4e connections). We use a time-window of 50ms for long term depression, with minimum value of -0.0005 reached at -10ms for the difference between post-synaptic spike and pre-synaptic spike; for long term potentiation the time-window is 25ms with maximum value of 0.001 for a 2ms difference.

3 Training on V2

The stimuli used to train V2 are based on those of [4], being composed of sets of articulating bars, as described by the lengths of the bars, and the angle between them. For our 4 orientations there is a total of 28 possible pairs of bars (each bar having length 5), indicated in fig. 3. We do not consider gratings or crossed bars.

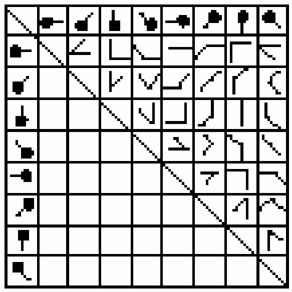


Fig. 3. The 28 pairs of bars used to train the network, the circles indicate the articulation point for each oriented bar. The 2 lines are interchangeable, hence the area under the diagonal line is just a mirror-image of that above.

Figure 4 shows the results of training on all 28 pairs for a single articulation point, only those neurons that show a highly selective response to a particular pair of bars are indicated (the non-preferred pairs have an activation level <65% that of maximal response, and most responses to non-preferred pairs is <50% maximal activation. Some V2 neurones have only spontaneous firing-levels as the stimuli do not impinge on their receptive fields, other neurons appear to represent a broader range angles, for instance 45° and 90° with little difference in firing rates. Generally there are more neurons that are preferentially active for the acute angles (45° and 90°, 36.5% and 34.4%, respectively) than for the obtuse angles

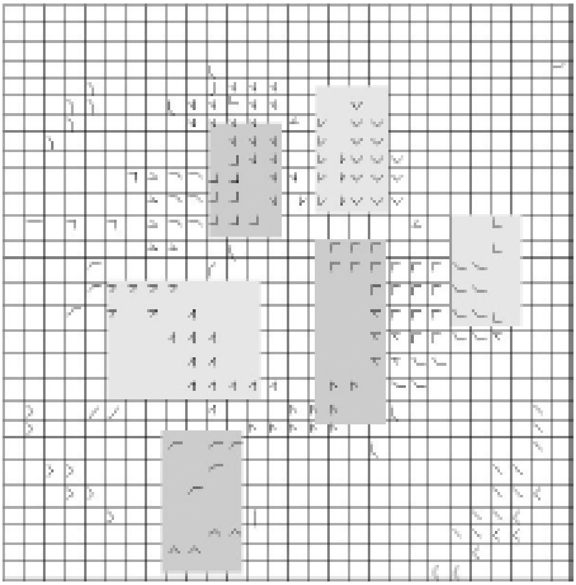


Fig. 4. The resulting V2 map after being trained on all 28 possible angles articulated at the same point in LGN

(135° and 180°, 21.2% and 7.9%, respectively). These results are slightly different to [4] which had: 39.5% of recorded neurons responsive to sharp angles (30°), these are comparable to our 45° angles (36.5%); 36.0% responsive to wide angles (60-150°) which are our 90° and 135° angles (55.6%); and 16.7% to 180° angles for which we had 7.9%. These differences are reasonably easy to explain since our results relate to those neurons which are highly responsive to specific pairs of bars, whilst the results from [4] relate to neurons that are selective to some extent for angles. It can also been seen that there are the movements of one bar or the other, indicated by the grey areas in fig. 4, showing that in some cases similar pairs of bars are represented by nearby neurons.

The response of a particular neurone to all 28 angles is shown in figure 5. This neuron is maximally responsive to an angle formed by the 180° and 315° bars with a firing-rate of 40Hz, this is a similar angle chosen in [4] and indicated in their figure 2. For non-preferred pairs the highest activation is for pair composed of 180° and 270° bars (25Hz, 62.5% firing-rate of the maximal response), and also for the angle formed by 45° and 225° lines (this is the long straight line that lies closest to the centre line of the maximal pair, an angle of 247.5°). All other responses of this neuron require that the pair of bars must include one of the 180° and 315° bars (the preferred pair) or the 225° bar which lies near the centre of the maximal pair.

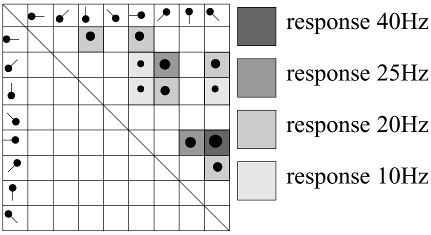


Fig. 5. The responses of a particular neuron to the 28 different angles, all other pairs are silent or have a firing-rate below spontaneous

To see if the network preserved some topography at the V2 level, the next training set was composed of the 4 right-angles required to define a square, with articulation points over a central 9*9 region of LGN. The results are shown in figure 6: each square represents a V2 neuron, the contents of each squareindicate the particular angle (grey-scale) and the centre of the angle input (small black squares) in LGN. Only the V2 nodes with maximal activation to a given input are indicated, where more than one centre of input is indicated all responses will be more than 10Hz greater than the largest response by that node to another angle. Some topographic invariance exists and some particular angles fail to generate a significant response in any V2 nodes (from a total of 324 angles 23 fail to produce a significant response).

4 Training on V4

The stimuli used to test V4 in [11] were of two sorts: complex curves and gratings of various forms. To simplify we do not consider the gratings (so do not discuss theinfluence of shading on object representations). For the complex shapes we take especially the case of arcs of a circle, in other words ‘C’ shapes. In order to develop V4 we train it on a set of rectangles, and test on both C’s (seen as part of the rectangle), single angles and whole rectangles. V2 is trained on the set of right-angles as in fig. 6 plus the horizontal and vertical long lines. The rectangles used have dimension 9*5 oriented horizontally or vertically, leading to C's with sides 5*5*5, 5*9*5 and 9*5*9. Figure 7 indicates the activity patterns in V4 for a rectangle, the small C, and a single right-angle. Greater maximal activity is seen for the C shape (~100Hz), a similar maximal firing rate occurs for the single angle, with the rectangle having maximal activity ~60Hz. The larger C's (not shown) have a firing rate lower than the rectangle.

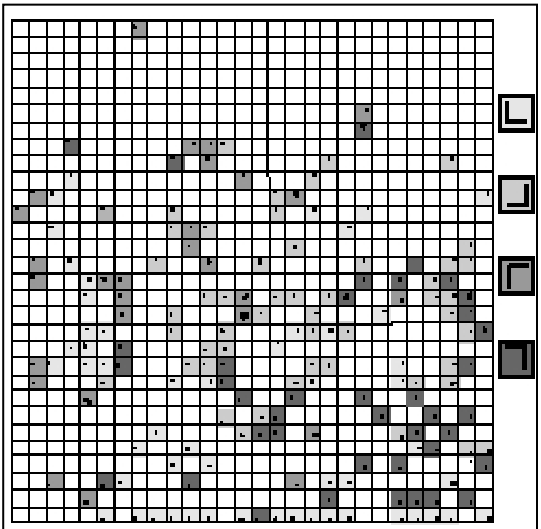


Fig. 6. V2 map resulting after training on 4 right-angles

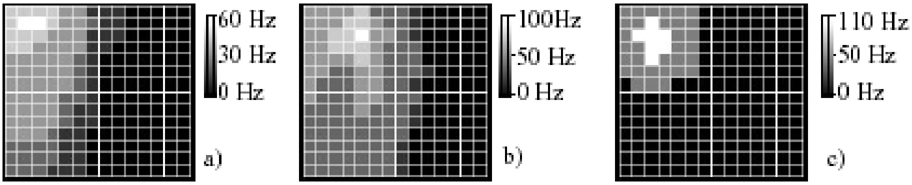


Fig. 7. V4 firing-rate patterns for: a) rectangle; b) small C-shape (5*5*5); c) right-angle

5 Discussion

We have presented results of the coding of various shapes arising in a model V2 and V4, embedded in a hierarchical architecture LGN->V1->V2->V4. The lowest module V1 is taken as a set of orientation templates, while the afferents to V2 and V4 are trained by STDP, using a suitable connectivity between the excitatory and inhibitory neurons present. These results compare very well with experimental data [4, 11].

Future work will compare representations in higher modules (TE, TEO and prefrontal cortical sites) developed by this approach with experimental data, as well as explore further properties of the lower-level results (such as their temporal dynamics). This will allow improvement of the architecture (such as a multi-layered cortex). The question of the importance of feedback for the representations will also be explored, in particular by comparison of non-attentive or attentive form, with attention as in [16]. Dynamical properties of representations and the role of feedback (clearly important, as providing enlargement to cell responses beyond the classical receptive-field properties) cannot be treated within the space of the paper and will be considered elsewhere, as will the mathematical aspects of the resulting coupled system which is complex.

Acknowledgements

We would like to thank the EPSRC of the UK and the Cognitive Systems Dept of the EC (under the GNOSYS grant) for financial support while this work was being performed, and our colleagues K Chokshi & C Panchev for useful discussions.

References

1. Somers DC, Nelson SB & Sur M (2005) An Emergent Model of Orientation Selectivity in Cat Visual Cortical Simple Cells. J Neurosci 15:5448-5465
2. Troyer TW, Krukowski AE, Priebe NJ & Miller KD (1998) Contrast-Invariant Orientation Tuning in Cat Visual Cortex: Thalamocortical Input Tuning and Correlation-Based Intracortical Activity. J Neurosci 18:5908-5927
3. Lauritsen TZ & Miller KD (2003) Different Roles for Simple-Cell and Complex-Cell Inhibition. J Neurosci 23:10201-10213
4. Ito M & Komatsu H (2004) Representation of Angles Embedded within Contour Stimuli in Area V2 of Macaque Monkeys. J Neurosci 24:3313-33249.
5. Riesenhuber M & Poggio T (1999) Hierarchical models of object recognition in cortex. Nature Neurosci. 2:1019-1025
6. Riesenhuber M & Poggio T (2000) Models of object recognition. Nature Neurosci. Supplement 3:1199-1204
7. Giese M & Leopold DA (2004) Physiologically inspired neural model for the encoding of face spaces. (preprint)
8. Fukushima K (2004) Neocognitron capable of incremental learning. Neural Networks 17:37-46
9. Deco G & Rolls E (2002) Object-based visual neglect: a computational hypothesis. Eur J Neurosci 16:1994-200013.

10. Van der Velde F & de Kamps M (2001) From Knowing What to Knowing Where: Modeling Object-based Attention with Feedback Disinhibition of Activation. J Cog Neursoci 13:479-491
11. Pasupathy A & Connor CE (2001) Shape Representation in Area V4: Position-Specific Tuning for Boundary Configuration. J Neurophysiol 86:2505-2519
12. Hegde J & Van Essen DC (2000) Selection for Complex Shapes in Primate Visual Area V2. J Neurosci 20:RC61:1-6
13. Felleman DJ & van Essen DC (1991) Distributed hierarchical processing in the primate cerebral cortex. Cerebral Cortex 1:1-47
14. Bi G-Q, Poo M-M (1998) Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. J Neurosci 18:10464-10472
15. Dan Y & Poo M-M (2004) Spike Timing-Dependent Plasticity of Neural Circuits.Neuron 44:23-30
16. Taylor J, Hartley M & Taylor N (2005) Attention as Sigma-Pi Controlled Ach-Based Feedback. IJCNN05 (submitted)

A Gradient Rule for the Plasticity of a Neuron's Intrinsic Excitability

Jochen Triesch[1,2]

[1] Department of Cognitive Science, UC San Diego,
9500 Gilman Drive, La Jolla, CA 92093-0515, USA
[2] Frankfurt Institute for Advanced Studies, Johann Wolfgang Goethe University,
Max-von-Laue-Str. 1, 60438 Frankfurt am Main, Germany
triesch@ucsd.edu
http://csclab.ucsd.edu

Abstract. While synaptic learning mechanisms have always been a core topic of neural computation research, there has been relatively little work on intrinsic learning processes, which change a neuron's excitability. Here, we study a single, continuous activation model neuron and derive a gradient rule for the intrinsic plasticity based on information theory that allows the neuron to bring its firing rate distribution into an approximately exponential regime, as observed in visual cortical neurons. In simulations, we show that the rule works efficiently.

1 Introduction

Individual biological neurons can change their intrinsic excitability through the modification of voltage gated channels [1,2,3,4,5]. Such plasticity of the intrinsic excitability, or *intrinsic plasticity*, has been observed across many species and brain areas — it is a ubiquitous phenomenon that may play an important role in shaping the dynamics of neural circuits [6]. Although our understanding of these processes and their underlying mechanisms remains quite unclear, it has been hypothesized that this form of plasticity contributes to a neuron's *homeostasis* of its firing rate level. More specifically the goal of intrinsic plasticity may be to obtain an approximately exponential distribution of the firing rate level. Such exponential distributions have been observed in visual cortical neurons and are thought to allow the neuron to transmit the maximum amount of information given a fixed level of metabolic costs [7]. This is because the exponential distribution has the highest entropy among all distributions of a non-negative random variable with a fixed mean. The idea that intrinsic plasticity may contribute to this goal was first explored in [8], where a Hodgkin-Huxley style model with a number of voltage gated conductances was considered. A learning rule was derived that adapts the properties of voltage gated channels to match the firing rate distribution of the unit to a desired distribution. We have recently proposed an intrinsic plasticity rule for a continuous activation model neuron, which is based on low order moments of a neuron's firing rate, and showed that it is effective in producing approximately exponential firing rate distributions [9]. We also showed that there may be a synergistic relation between intrinsic and

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 65–70, 2005.

synaptic learning mechanisms and that the two may work together to discover sparse directions in the input. The intrinsic plasticity rule proposed in [9] was formulated as a set of simple proportional control laws. The goal of this paper is to derive a gradient rule for intrinsic plasticity, and to validate its performance through simulations.

2 Gradient Rule for Intrinsic Plasticity

Let us assume the input to the neuron's firing mechanism (total synaptic current arriving at the soma), denoted as x has the distribution $f_x(x)$. In our model, x is passed through the nonlinearity g to obtain the neuron's output $y = g(x)$ (firing rate). We assume that g, being a firing rate, is non-negative. Further assuming that g is strictly monotonically increasing, the distribution of y is given by:

$$f_y(y) = \frac{f_x(x)}{\frac{\partial y}{\partial x}} . \tag{1}$$

As motivated above, we are looking for ways to adjust g so that $f_y(y)$ is approximately exponential, i.e. we would like $f_y(y)$ to be "close" to $f_{\exp}(y) = \frac{1}{\mu}\exp(\frac{-y}{\mu})$. A natural metric for measuring the distance is to consider the Kullback Leibler divergence (KL-divergence) between f_y and $f_{\exp}$:

$$D \equiv d(f_y \mid\mid f_{\exp}) = \int f_y(y) \log \left(\frac{f_y(y)}{\frac{1}{\mu}\exp\left(\frac{-y}{\mu}\right)} \right) dy \tag{2}$$

$$= \int f_y(y) \log(f_y(y)) dy - \int f_y(y) \left(-\frac{y}{\mu} - \log \mu \right) dy \tag{3}$$

$$= -H(y) + \frac{1}{\mu} E(y) + \log \mu , \tag{4}$$

where $H(y)$ is the entropy of y and $E(y)$ is its expected value. Note that this relation directly shows that the exponential distribution is the maximum entropy distribution among all distributions with a fixed mean: for fixed $E(y)$, the right hand side of (4) is minimized when H is maximized. The left hand side, however, because it is a KL-divergence, is non-negative, and is zero if and only if $f_y(y) = f_{\exp}(y)$. Another way to look at (4) is that in order to minimize $d(f_y \mid\mid f_{\exp})$ we need to maximize the entropy $H(y)$ while minimizing the expected value $E(y)$. The factor $1/\mu$ can be seen as scaling the relative importance of maximizing entropy vs. minimizing the mean. The constant $-\log \mu$ does not depend on g and is thus irrelevant for this minimization.

Now, we would like to derive a stochastic gradient descent rule for intrinsic plasticity that strives to minimize (4). To this end, we consider a parameterized non-linearity g and ask how D changes if g changes. Let us consider the sigmoid nonlinearity given by:

$$y = g(x) = \frac{1}{1 + \exp(-(ax + b))} . \tag{5}$$

This nonlinearity depends on two parameters a and b. To construct the gradient, we must consider the partial derivatives of D with respect to a and b:

$$\frac{\partial D}{\partial a} = \frac{\partial d(f_y \mid\mid f\text{exp})}{\partial a} = -\frac{\partial H}{\partial a} + \frac{1}{\mu}\frac{\partial E(y)}{\partial a} \tag{6}$$

$$= E\left(-\frac{\partial}{\partial a}\log\left(\frac{\partial y}{\partial x}\right) + \frac{1}{\mu}\frac{\partial y}{\partial a}\right) \tag{7}$$

$$= -\frac{1}{a} + E\left(-x + \left(2 + \frac{1}{\mu}\right)xy - \frac{1}{\mu}xy^2\right) \ , \tag{8}$$

where we have used (1) and moved the differentiation inside the expected value operation to obtain (7), and exploited:

$$\log\left(\frac{\partial y}{\partial x}\right) = \log a + \log y + \log(1-y) \tag{9}$$

in the last step. Similarly, for the partial derivative with respect to b we find:

$$\frac{\partial D}{\partial b} = \frac{\partial d(f_y \mid\mid f\text{exp})}{\partial b} = -\frac{\partial H}{\partial b} + \frac{1}{\mu}\frac{\partial E(y)}{\partial b} \tag{10}$$

$$= -1 + E\left(\left(2 + \frac{1}{\mu}\right)y - \frac{1}{\mu}y^2\right) \ . \tag{11}$$

The resulting stochastic gradient descent rule is given by $a := a + \Delta a$ and $b := b + \Delta b$, with:

$$\Delta a = \eta\left(\frac{1}{a} + x - \left(2 + \frac{1}{\mu}\right)xy + \frac{1}{\mu}xy^2\right) \tag{12}$$

$$\Delta b = \eta\left(1 - \left(2 + \frac{1}{\mu}\right)y + \frac{1}{\mu}y^2\right) \ , \tag{13}$$

where η is a small learning rate. This rule is very similar to the one derived by Bell and Sejnowski for a single sigmoid neuron maximizing its entropy [10]. However, our rule has additional terms stemming from the objective of keeping the mean firing rate low. Note that this rule is strictly local. The only quantities used to update the neuron's nonlinear transfer function are x, the total synaptic current arriving at the soma, and the firing rate y.

The above derivation can be generalized to other non-linearities with different adjustable parameters. The requirements are that g should be strictly monotonically increasing and differentiable with respect to y, and the partial derivatives of $\log \partial y/\partial x$ with respect to the parameters of g (in the above case a and b) must exist.

3 Experiments

3.1 Behavior for Different Distributions of Total Synaptic Current

To illustrate the behavior of the learning rule, we consider a single model neuron with a fixed distribution of synaptic current $f_x(x)$. Figure 1 illustrates the result of adapting the neuron's nonlinear transfer function to three different input distributions — gaussian, uniform, and exponential. We set the unit's desired mean activity to $\mu = 0.1$, i.e., a tenth of its maximum activity, to reflect the low firing rates observed in cortex. The following results do not critically depend on the precise value of μ. In each case, the sigmoid nonlinearity moves close to the optimal nonlinearity that would result in an exactly exponential distribution of the firing rate, resulting in a sparse distribution of firing rates. Since the sigmoid has only two degrees of freedom, the match is not perfect, however. As can be seen in the figure, large deviations from the optimal transfer function can sometimes be observed where the probability density of the input distribution is low. A closer fit could be obtained with a different nonlinearity g with more free parameters. It is presently unclear, however, what additional degrees of freedom biological neurons have in this respect.

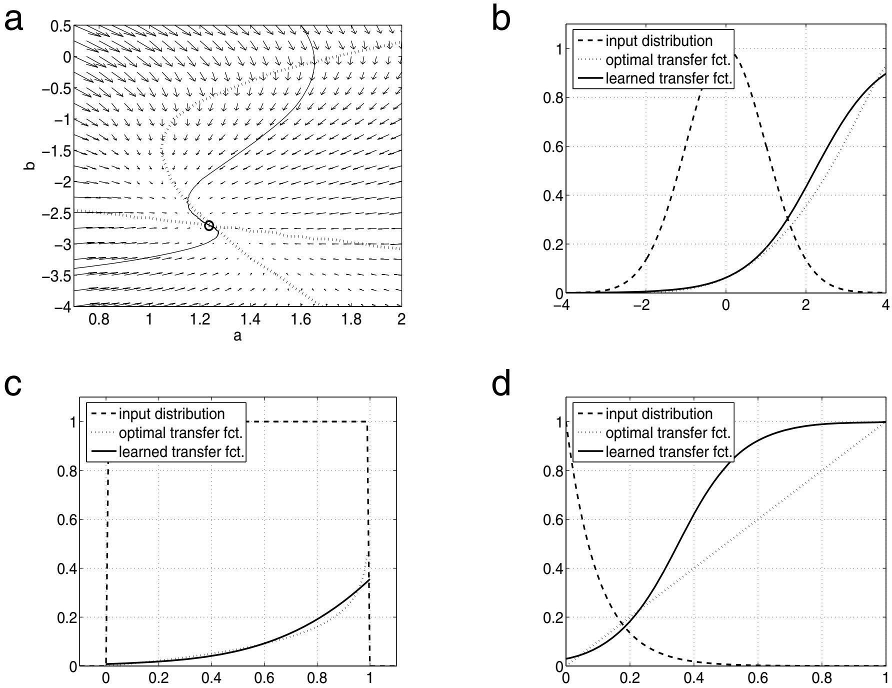


Fig. 1. Dynamics of the intrinsic plasticity rule for various input distributions. **a,b:** Gaussian input distribution. Panel **a** shows the phase plane diagram. Arrows indicate the flow field of the system. Dotted lines indicate approximate locations of the null-clines (found numerically). Two example trajectories are exhibited (solid lines) which converge to the stationary point (marked with a circle). Panel **b** shows the theoretically optimal (dotted) and learned sigmoidal transfer function (solid). The Gaussian input distribution (dashed, not drawn to scale) is also shown. **c,d:** same as **b** but for uniform and exponential input distribution. Parameters were $\mu = 0.1$, $\eta = 0.001$.

The results obtained are very similar to the ones from our previous study [9], which used a set of two simple control laws that updated the sigmoid nonlinearity based on estimates of the first and second moment of the neuron's firing rate distribution. The current rule leads to faster convergence, however.

3.2 Response to Sudden Sensory Deprivation

The proposed form of plasticity of a neuron's intrinsic excitability may help neurons to maintain stable firing rate distributions in the presence of systematic changes of their afferent input. An interesting special case of such a change is that of sensory deprivation. Here, we model sensory deprivation as a change to the variance σ^2 of the total synaptic current arriving at the soma. Figure 2 shows how the neuron learns to adjust to the changed distribution of total synaptic current. After a transient phase of low variability in its activity, the neuron gradually re-establishes its desired firing rate distribution.

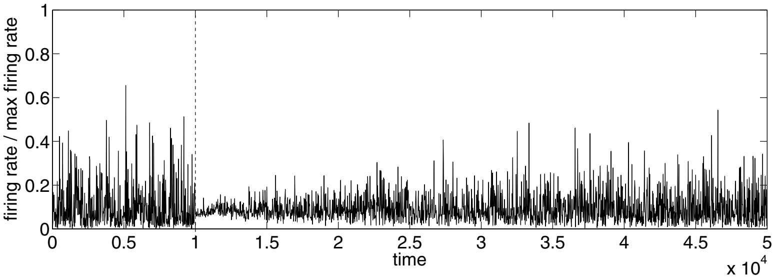


Fig. 2. Response to sudden sensory deprivation. The graph shows the unit's firing rate y as a function of the number of presented inputs (time t). We only plot the response to every 20th input. At time step 10000 the standard deviation of the input signal is reduced by a factor of 10. In response, the neuron adjusts its intrinsic excitability to restore its desired exponential firing rate distribution.

4 Discussion

Intrinsic plasticity mechanisms that change the membrane properties of individual neurons represent a fundamental aspect of cortical plasticity. While these mechanisms are still poorly understood, there is a shared belief that they may contribute to a neuron's homeostasis by helping it to keep its firing rate in a desired regime. More specifically, they may allow the neuron to transmit a maximum amount of information to downstream targets while minimizing the metabolic costs to do so. In this paper, we derived a gradient rule for intrinsic plasticity that adjusts the parameters of a sigmoid nonlinearity to drive the neuron's firing distribution towards an exponential distribution — as frequently observed in visual cortical neurons [7]. From an information theoretic point of view, this approach is more principled than the proportional-control-law rules we have proposed previously [9], because these control-laws only take the first and second moments of the distribution into account. In a set of simulations, we showed that the rule works effectively.

While we have only considered an individual model neuron in this paper and in our previous work, we think it is very important to explore the effects of intrinsic plasticity at the network level [6]. What is its role in the learning of sparse, map-like sensory representations? How will it influence the dynamics of recurrent networks? At the same time, it will be interesting to develop less abstract, biophysically plausible models of intrinsic plasticity mechanisms for spiking neurons [8], and their interaction with spike-timing dependent plasticity. We feel that in any theory of cortical computation and plasticity, mechanisms that change a neuron's intrinsic excitability must play a central role.

Acknowledgments

This work is supported in part by the National Science Foundation under grant NSF IIS-0208451. I thank Nicholas Butko for comments on the manuscript.

References

1. Desai, N.S., Rutherford, L.C., Turrigiano, G.G.: Plasticity in the intrinsic excitability of cortical pyramidal neurons. Nature Neuroscience **2** (1999) 515–520
2. Zhang, W., Linden, D.J.: The other side of the engram: Experience-driven changes in neuronal intrinsic excitability. Nature Reviews Neuroscience **4** (2003) 885–900
3. Daoudal, G., Debanne, D.: Long-term plasticity of intrinsic excitability: Learning rules and mechanisms. Learning and Memory **10** (2003) 456–465
4. Zhang, M., Hung, F., Zhu, Y., Xie, Z., Wang, J.H.: Calcium signal-dependent plasticity of neuronal excitability developed postnatally. J. Neurobiol. **61** (2004) 277–287
5. Cudmore, R., Turrigiano, G.: Long-term potentiation of intrinisic excitability in lv visual cortical neurons. J. Neurophysiol. **92** (2004) 341–348
6. Marder, E., Abbott, L.F., Turrigiano, G.G., Liu, Z., Golowasch, J.: Memory from the dynamics of intrinsic membrane currents. Proc. Natl. Acad. Sci. USA **93** (1996) 13481–13486
7. Baddeley, R., Abbott, L.F., Booth, M., Sengpiel, F., Freeman, T.: Responses of neurons in primary and inferior temporal visual cortices to natural scenes. Proc. R. Soc. London, Ser. B **264** (1998) 1775–1783
8. Stemmler, M., Koch, C.: How voltage-dependent conductances can adapt to maximize the information encoded by neuronal firing rate. Nature Neuroscience **2** (1999) 521–527
9. Triesch, J.: Synergies between intrinsic and synaptic plasticity in individual model neurons. In Saul, L.K., Weiss, Y., Bottou, L., eds.: Advances in Neural Information Processing Systems 17. MIT Press, Cambridge, MA (2005)
10. Bell, A.J., Sejnowski, T.J.: An information-maximization approach to blind separation and blind deconvolution. Neural Computation **7** (1995) 1129–1159

Building the Cerebellum in a Computer

Tadashi Yamazaki and Shigeru Tanaka

Laboratory for Visual Neurocomputing, RIKEN Brain Science Institute,
2-1 Hirosawa, Wako, Saitama 351-0198, Japan
tyam@brain.riken.jp, shigeru@riken.jp

Abstract. We have built a realistic computational model of the cerebellum. This model simulates the cerebellar cortex of the size 0.5mm×1mm consisting of several types of neurons, which are modeled as conductance-based leaky integrate-and-fire units with realistic values of parameters adopted from known anatomical and physiological data. We demonstrate that the recurrent inhibitory circuit composed of granule and Golgi cells can represent a time passage by population of active granule cells, which we call "the cerebellar internal clock". We also demonstrate that our model can explain Pavlovian eyelid conditioning, in which the cerebellar internal clock plays an important role.

1 Introduction

It is known that the cerebellum plays a critical role in procedural learning and memory. An example is Pavlovian eyelid conditioning [1]. This paradigm involves repeated presentations of a conditional stimulus (CS; e.g., a tone) paired with an unconditional stimulus (US; e.g., an airpuff) delayed for a certain time after the CS onset, which elicits an automatic conditioned response (CR; e.g., an eyeblink). The subject is tested for learning an association between the CS and the US, as evidenced by the CR in anticipation of the US. The cerebellum is known to play a necessary role in learning a well-timed CR in anticipation of the US. Thus, the cerebellum must have a kind of "memory trace" of the CS that bridges the interstimulus interval (ISI) to form a CS-US association, but how?

We have theoretically studied a simple random recurrent inhibitory network, which we call an internal clock [2]. In this model, activity patterns of neurons evolved with time without recurrence due to random recurrent connections among neurons. The sequence of activity patterns is generated by the trigger of an external signal and the generation is robust against noise. Therefore, a time passage from the trigger of an external signal can be represented by the sequence of activity patterns.

We have previously shown that our internal clock model is derived from the recurrent inhibitory circuit composed of granule and Golgi cells in the cerebellum. In the present study, we build a realistic computational model of the cerebellum. We demonstrate that the realistic model also works as an internal clock. We then examine if the present model can explain Pavlovian eyelid conditioning. We confirm that the anticipatory CR is induced by the combination of the

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 71–77, 2005.

internal clock and the long-term depression (LTD) [3] at Purkinje cells. Furthermore, we study the functional role of the cerebellar-precerebellar feedback.

2 Model Description

Figure 1 represents the schematic of known cell types and synaptic connections in the cerebellum [4].

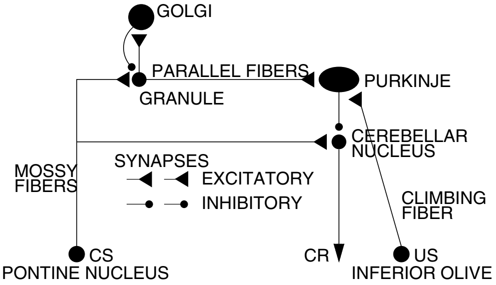


Fig. 1. Schematic of known cell types and synaptic connections in the cerebellum

The neural signal of the CS comes from the pontine nucleus through mossy fibers to the cerebellar nucleus, which attempts to elicit spike activity. On the other hand, the CS signal is also sent to Purkinje cells relayed by granule cells, and then Purkinje cells inhibit the cerebellar nucleus. Thus, the cerebellar nucleus receives both excitatory inputs directly through mossy fibers and inhibitory inputs from Purkinje cells. A Golgi cell receives excitatory inputs from granule cells through parallel fibers and inhibits a set of nearby granule cells. Here we assume that the connections from granule to Golgi cells are random while these from Golgi to granule cells are uniform. The neural signal of the US comes from the inferior olive through climbing fibers to Purkinje cells.

Neurons are modeled as conductance-based leaky integrate-and-fire units.

$$C\frac{dV}{dt} = -g_{\mathrm{leak}}(V - E_{\mathrm{leak}}) - g_{\mathrm{ex}}(V - E_{\mathrm{ex}}) - g_{\mathrm{inh}}(V - E_{\mathrm{inh}}) - g_{\mathrm{ahp}}(V - E_{\mathrm{ahp}}), \quad (1)$$

where V and C are the membrane potential and the capacitance, respectively. The membrane potential is calculated by four types of currents specified by the right-hand side in Eq. (1), they are, the leak, excitatory and inhibitory currents, and the current representing the afterhyperpolarization which simulates a refractory period. For each type $c \in \{\mathrm{leak}, \mathrm{ex}, \mathrm{inh}, \mathrm{ahp}\}$, the current is calculated by the conductance g_c and the reversal potential E_c. A conductance is calculated by the convolution of the alpha function representing the shape of postsynaptic potentials and spike trains of presynaptic neurons. Neurons elicit a spike when the membrane potential (V) exceeds the threshold (θ). After the spiking, V is reset to the resting potential which is set at the leak potential and a refractory period follows. For further details, see [5]. Parameter values are adopted from known anatomical and physiological data, though they are not shown here due to the page limit.

The model simulates the cerebellar cortex of the size 0.5mm×1mm for 1sec, which consists of 512 Golgi cells, 512 granule cells and 1 Purkinje cell, and 1 excitatory neuron in the cerebellar nucleus. The 4th order runge-kutta method is used to calculate Eq. (1), where $\Delta t = 1$msec. CS signals are represented by Poisson spikes consisting of a transient component of 200Hz for the first 20msec and a sustained component of 50Hz for 1sec, whereas US signals are assumed to arrive at the Purkinje cell at 500msec. LTD is simulated by decreasing synaptic weights of parallel fiber synapses of granule cells which elicit spikes for 20msec after the US onset, specifically, the synaptic weights are multiplied by 0.8. If the neuron in the cerebellar nucleus elicits spikes at 500msec, and the time to first spike is slightly earlier than 500msec, we regard it as the success of learning.

We demonstrate that the activity patterns of granule cells can represent a time passage, that is, the population of spiking granule cells at one time step is dissimilar to the population at a different time step when the interval between the two steps are large. Therefore, we use the following correlation function as the similarity index.

$$C(t_1, t_2) = \frac{\sum_i z_i(t_1) z_i(t_2)}{\sqrt{\sum_i z_i^2(t_1)}\sqrt{\sum_i z_i^2(t_2)}}, \tag{2}$$

where $z_i(t) = 1$ if granule cell i generates a spike at time t, and 0 otherwise.

3 Results

The top left panel in Fig. 2 plots spikes of the first 100 cells out of 512 granule cells for 1sec. All granule cells were activated for 20msec by the transient component of the input signal, which was followed by a quiescent period in which few granule cells became active. Then, granule cells showed random repetition of spiking/nonspiking states because of the random recurrent inhibition via Golgi cells. Thus, the population of spiking granule cells changed with time. The top middle panel shows the similarity index calculated using Eq. (2). Since Eq. (2) takes two arguments of t_1 and t_2, and the simulation was conducted for 1sec with $\Delta t = 1$msec, we obtained a 1000×1000 matrix, where the row and columns were specified by t_1 and t_2, respectively. Similarity indices were plotted in a gray scale in which black indicated 0 and white 1. A white band appeared diagonally. Since the similarity index at the identical step ($t_2 = t_1$) takes 1, the diagonal elements of the similarity index appeared white. The similarity index decreased monotonically as the interval between t_1 and t_2 became longer. This was confirmed as shown in the top right panel, which plots $C(200, t)$, $C(500, t)$ and $C(800, t)$. This result indicates that the population of spiking granule cells changed gradually with time without recurrence, and that a time passage from the stimulus onset can be represented. The bottom panels in Fig. 2 plot the temporal change of membrane potentials of some granule cells. Different fluctuations of membrane potentials for different granule cells are evident.

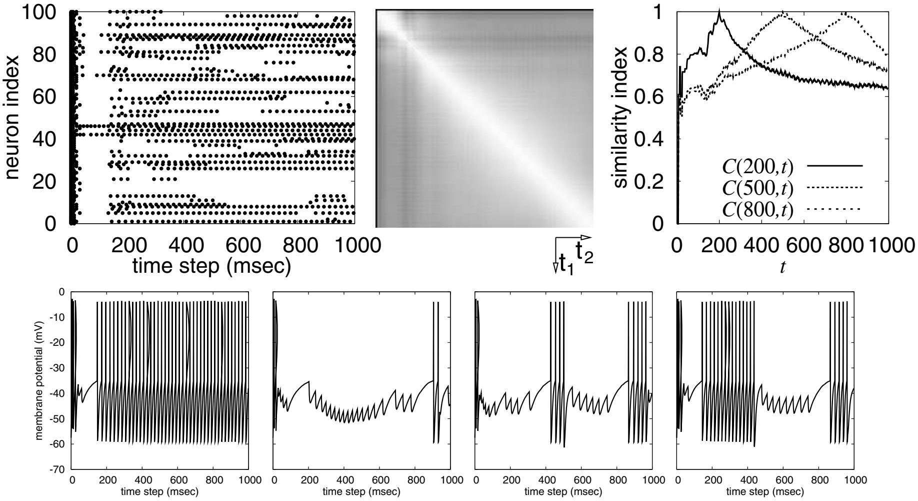


Fig. 2. Top panels: Raster plot of spiking granule cells (left), similarity index ($C(t_1, t_2)$) (middle), plots of $C(200, t)$, $C(500, t)$ and C(800,t) (right). Bottom panels: Plots of membrane potentials of some granule cells.

We examined how our model learns the CS-US interval and elicits the CR at the correct timing slightly before the onset of US signals. Top panels in Fig. 3 plot the activity of the Purkinje cell at the 1st, 4th, 7th, and 10th trials, whereas bottom panels the activity of the cerebellar nucleus neuron at the same trials. At the 1st trial, the Purkinje cell regularly elicited spikes at about 80Hz, and hence the nucleus neuron were tonically inhibited for 1sec. Purkinje cell activity decreased as the trial number increased because of LTD at parallel fiber synapses, and at the 4th trial the firing rate was as low as 40Hz. Moreover, the Purkinje cell tended to reduce the spike rate around 500msec. Accordingly, the nucleus neuron increased activity around 500msec though the neuron was still tonically inhibited. At the 7th trial, the Purkinje cell completely stopped spiking from 400 to 600 msec and the nucleus neuron elicited a few spikes expressing the CR. The first spike, however, appeared after 500msec, suggesting that the spike activity could not be the anticipatory CR against the US. At the 10th trial, the time window of nonspiking period of the Purkinje cell enlarged and the total spike activity further decreased. The nucleus neuron elicited more spikes and the time to first spike shifted as early as 420msec, which is sufficient to anticipate the US. Therefore the anticipatory CR was expressed.

The bottom rightmost panel in Fig. 3 shows that the nucleus neuron elicited the first spike at 420msec and the spiking period was sustained up to 650msec. The early part of the spikes is necessary to induce the anticipatory CR, whereas the late part may not be necessary and may cause over-expression of the CR. How do we suppress the occurrence of these unnecessary spikes?

It is known that there is a feedback connection from the cerebellar nucleus to the precerebellar nucleus which sends inputs to the cerebellum [6]. The feedback

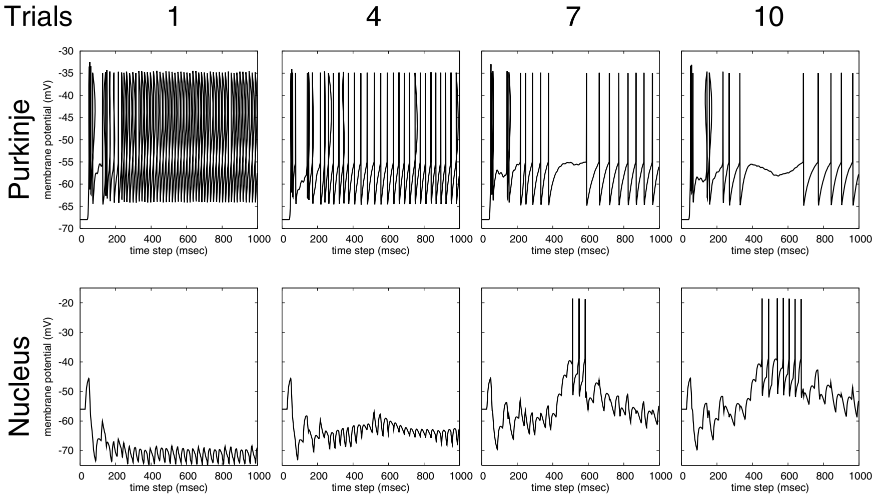


Fig. 3. Plots of membrane potentials of the Purkinje cell (top panels) and the neuron in the cerebellar nucleus (bottom panels) at the 1st, 4th, 7th and 10th trials (from left to right).

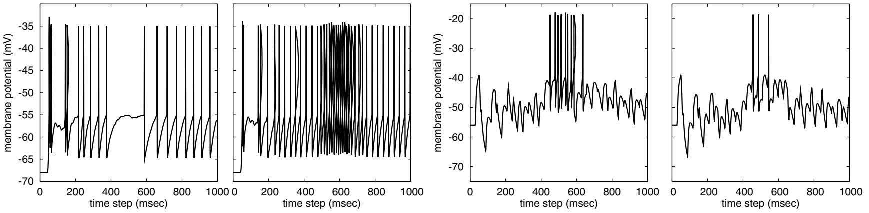


Fig. 4. (From left to right) Plots of membrane potentials of the Purkinje cell which receive the US signals, the Purkinje cell which does not receive the US signals, the nucleus neuron in the model without feedback, and the nucleus neuron in the model with feedback.

connection transmits the activity of the nucleus neuron to Purkinje cells after the CR onset. Purkinje cells which receive the US signals are not involved because they are depressed when the feedback signals arrive at. On the other hand, Purkinje cells which do not receive the US signals would show increasing activity after the CR onset. Therefore, if we incorporate one more Purkinje cell which do not receive the US signals into the model and the cell also inhibits the nucleus neuron, then the nucleus neuron would elicit spikes for the anticipatory CR and be suppressed after that.

We examined whether the above scenario was possible. We added the feedback connection by which the activity of the nucleus neuron was transmitted to granule cells through mossy fibers. We also incorporated one more Purkinje cell in the model cerebellum, which was assumed to receive signals from all granule cells through their parallel fibers. The parallel fiber synapses were not depressed: the synaptic weights were unchanged. To compensate the strong inhibition from the new Purkinje cell to the nucleus, the connection strength from mossy fibers to the nucleus was increased. The leftmost panel in Fig. 4 plots the activity of the Purkinje cell receiving the US signals at which LTD was induced. The cell showed decreasing activity starting at 400msec. The next right panel plots the activity of the Purkinje cell which was added newly. The cell showed increasing activity after 500msec. The 3rd panel from the left shows the activity of the nucleus neuron in the model without feedback, in which, the late spikes after 500msec were evident. The rightmost panel plots the activity of the nucleus neuron in the model with the feedback connection. The late spikes were suppressed while the time to first spike was still at about 420msec. This result suggests that the feedback connection plays an essential role for accurate expression of the CR.

4 Discussion

We built a realistic model of the cerebellum and examined whether the model accounts for the mechanism of Pavlovian eyelid conditioning. We have theoretically studied the dynamics of the recurrent inhibitory circuit composed of granule and Golgi cells in the rate-coding scheme and showed that the population of active granule cells could represent a time passage, which we call the cerebellar internal clock [2]. In the present study we confirmed that the realistically implemented spiking neural network also works as the internal clock. We then demonstrated that the Purkinje cell stopped inhibition to the neuron in the cerebellar nucleus at the timing of the US onset by LTD at parallel fiber synapses at the Purkinje cell and hence the nucleus neuron could elicit spikes expressing the CR. Since the spikes were elicited slightly earlier than the US onset, the CR anticipated the US. We also studied the functional role of the feedback connection from the cerebellar nucleus to the precerebellar nucleus. We hypothesized that the feedback can stop the unnecessary activity of the nucleus neuron after the US onset by increasing the activity of Purkinje cells which do not receive the US signals, and confirmed that this scenario was possible. Furthermore, it is evident that Purkinje cells which receive US signals show decreasing activity at the US onset, whereas cells which do not receive US signals show increasing activity [7]. Our hypothesis implies that the decreasing activity is due to LTD while the increasing activity is due to the feedback.

Although we did not show the results, our model is not robust against Poisson spikes in inputs as it is. We generated two different Poisson spikes of 50Hz using different random seeds and presented to the model. The obtained two activity patterns of granule cells were completely different, suggesting that the time coding is not robust. This problem, however, was resolved by increasing the number of granule cells. As the number increased two activity patterns became similar, and when the number was 1024 times larger, the two activity patterns were almost identical for 800msec. The reason is as follows. A Golgi cell receives input signals from granule cells. If the number of presynaptic cells is small, the activity of a Golgi cell strongly depends on the fluctuation of input signals. On the other hand, if the number is huge, the net input to a Golgi cell is somehow averaged and the effect of fluctuation is diminished by the law of large numbers. Since a Golgi cell inhibits a set of nearby granule cells, these granule cells show a similar activity pattern. Thus, the activity patterns become similar regardless of randomness in Poisson spikes. This indicates that the model can become robust when much more granule cells are incorporated. Hence, in this study we considered the noiseless case by using one Poisson spikes and simplified the simulation setting.

Mauk and his colleagues have built a similar cerebellar model as ours [8]. They have demonstrated a nice matching in results between experiments and simulations on Pavlovian eyelid conditioning [9]. They, however, did not clarify the mechanism of why/how their model works. On the other hand, we elucidated the mechanism by focusing on the functional roles of circuits.

References

1. Mauk, M.D., Donegan, N.H.: A model of Pavlovian eyelid conditioning based on the synaptic organization of the cerebellum. Learning & Memory **3** (1997) 130–158
2. Yamazaki, T., Tanaka, S.: Neural modeling of an internal clock. Neural Computation **17** (2005) 1032–1058
3. Ito, M.: Long-term depression. Ann. Rev. Neurosci. **12** (1989) 85–102
4. Ito, M.: The Cerebellum and Neuronal Control. Raven Press, New York (1984)
5. Gerstner, W., Kistler, W.M.: Spiking Neuron Models. Cambridge University Press (2002)
6. Tsukahara, N., et al.: Properties of cerebello-precerebellar reverberating circuits. Brain Res. **274** (1983) 249–259
7. Berthier, N., Moore, J.: Cerebellar purkinje cell activity related to the classically conditioned nictitating membrane response. Exp. Brain Res. **63** (1986) 341–350
8. Medina, J.F., Mauk, M.D.: Computer simulation of cerebellar information processing. Nature Neurosci. **3** (2000) 1205–1211
9. Medina, J.F., et al.: Timing mechanisms in the cerebellum: Testing predictions of a large-scale computer simulation. J. Neurosci. **20** (2000) 5516–5525

Combining Attention and Value Maps

Stathis Kasderidis[1,*] and John G. Taylor[2]

[1] Foundation for Research and Technology – Hellas,
Institute of Computer Science, Vassilika Vouton,
P.O. Box 1385, 711 10 Heraklion, Greece
stathis@ics.forth.gr
[2] King's College, Dept. of Mathematics,
Strand, London WC2R 2LS, UK
john.g.taylor@kcl.ac.uk

Abstract. We present an approach where we combine attention with value maps for the purpose of acquiring a decision-making policy for multiple concurrent goals. The former component is essential for dealing with an uncertain and open environment while the latter offers a general model for building decision-making systems based on reward information. We discuss the multiple goals policy acquisition problem and justify our approach. We provide simulation results that support our solution.

1 Introduction

There is currently much research interest in developing autonomous agents. One of the primary problems in the field is that of multiple goal satisfaction. Approaches such as reinforcement learning have provided a general method for modelling the goal satisfaction problem for the case of a single goal at a given time [1]. Finding a suitable policy for multiple concurrent goals is a generalisation of the previous problem. Again reinforcement learning methods can be applied directly. However, the approach lacks the ability to deal with an ever-changing environment in an immediate way. This can be handled by attention. There is much work in recent years that has been devoted in understanding attention [2]. It has been modelled in a control theory framework by the second author [3]. Inspired by the above developments we have developed recently the Attentional Agent architecture [4] which combines a goals-based computational model with an attention mechanism for selecting priority of goals dynamically in run time, for handling novelty and unexpected situations as well as learning of forward models based on the level of attention [5]. We now extend this model further to allow the attention mechanism to act as an alarm system when we approach limiting conditions. This model can be combined with a reinforcement learning approach for single goals to provide the solution for multi-goal policy acquisition. The structure of the paper is as follows: In section 2 we present a concrete problem statement and we describe the environment used for testing a robotic agent. In section 3 we review the Attentional Agent Architecture and its extension to multi-goal policy acquisition. In section 4 we present supporting simulations.

[*] Corresponding author.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 79 – 84, 2005.

2 Problem Specification

To test the proposed architecture we select a robot navigation task. The concrete setting is as follows: We assume the existence of a suitable space where the robotic agent moves from a point A to a point B transporting some item of interest. The high-level decomposition of the task is shown in Fig. 1. Transport is the primary agent goal. At the same time there is the goal of maintaining itself in a working condition, which appears in Fig. 1, as the Power Monitor goal. Inside the space there are a number of stationary and moving objects. The overall task is to provide the Transport service of moving items from point A to B while avoiding collisions with other moving and stationary objects and also by making sure that the robot always has enough power. If the power level drops low the agent should recharge itself and return to the previous task. Recharging can take place in the recharge station (Terminal C) or by collecting small charges from objects of class Obj+ (by moving to the same grid cell). Correspondingly, objects of class Obj- should be avoided as they reduce the power level if touched. We assume that objects Obj+ carry a reward of +1, while objects of Obj- carry a reward of -1. Moving to goal position B achieves a reward signal of +20. All other states are assumed as having zero reward initially. A possible configuration of the space is shown in Fig. 2.

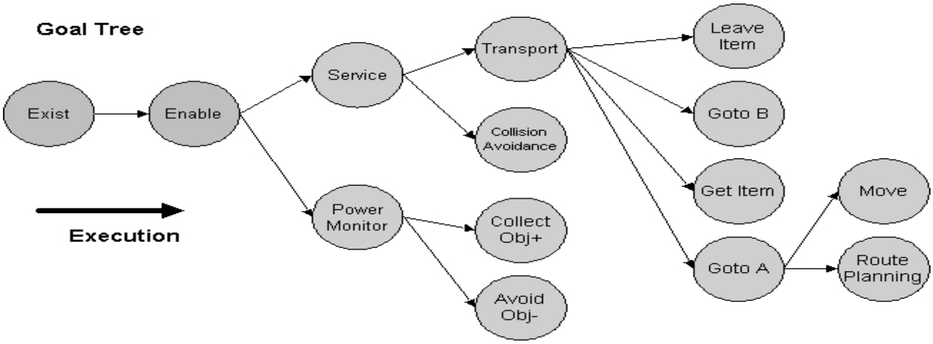


Fig. 1. Goals Tree of agent

In Fig. 1 there are three high-level goals: Transport, Collision-Avoidance and Power Monitor. The Transport goal executes a sequential program of four sub-goals {Goto A, Get Item, Goto B, Leave Item}. The Goto A goal is further decomposed to Route-Planning and Move goals. All goals are executed ultimately by calling primitives which are not shown in Fig. 1. The Route-Planning goal is responsible for collecting the current sensory state and for calculating a new position for moving the agent closer to the goal position, based on the value map of the corresponding goal. Then it passes this new location to the Move goal to execute it. Internally it uses predictive models (Forward Models) for forecasting the possible position of the other moving agents.

Given that a location is “closer” to the target position and it will not be occupied by other agents in the next time instance, the location is selected. During the execution of

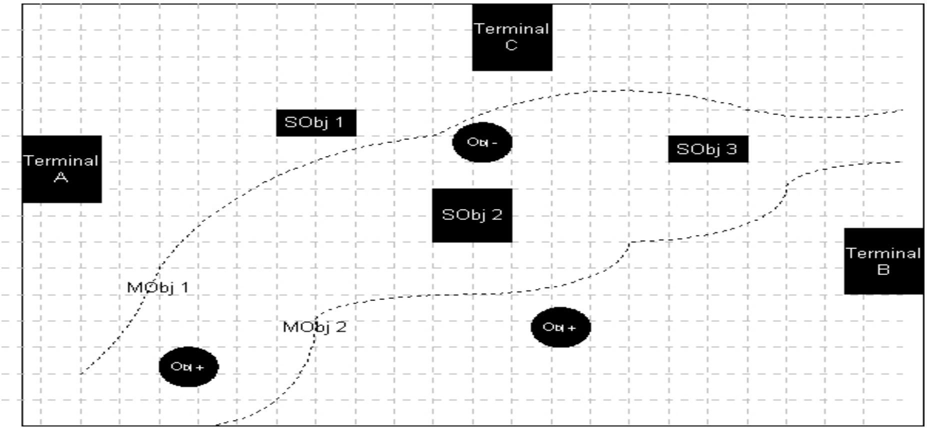


Fig. 2. A possible configuration of GridWorld. MObj x represent a moving object x. SObj x represent static objects. Obj+/- represent objects that are assumed static but having a positive/negative influence on the power level of the agent if touched. Terminal C is the recharging station. Terminals A and B is the start and finish position of the Transport goal.

the actual move, it might occur that some other agent moved to the calculated location, because our movement prediction was wrong. We guard against such a case using the Collision-Avoidance goal, which implements a motor attention scheme. This goal is normally suppressed by the Transport goal; if however a collision is imminent, an attention event is created, which in turn raises the overall Action-Index of the goal. The final result is that the Transport goal is suppressed due to losing the global attentional competition against the Collision-Avoidance goal. The Power Monitor goal monitors the current power level and if low it will re-direct the robot to the recharging terminal or to a nearby Obj+ object. The policy is not hard-wired but learnt as described in section 3.2. Care is given to avoid objects Obj-. Collision with a static or moving object corresponds to a reward of -5 and -10 respectively. We also assume that the maximum power level is 1000 power units, motion expends 1 power unit per cell, and touching Obj+ and Obj- increases / reduces the power level by +10/-10 units with a reward of +1/-1 respectively.

3 Multiple Goal Policy Acquisition for Attentional Agents

3.1 Attentional Agent Architecture

The Attentional Agent architecture was thoroughly discussed in [4]. An Attentional Agent is a system which has the following major components: 1. A goal set, organised in a tree (GoalsTree), see Fig 1; 2. A complex execution unit, called a goal, with an internal structure; 3. A global attention-based competition mechanism, which influences the priority of the goals; 4. A local attention-based mechanism, in the scope of a goal, which detects novel states and initiates learning of new models or adaptation of existing ones. The local attention process works in dual modes: sensory and motor ones; 5. Each goal contains the following major modules: *State Evaluation*, *Rules*, *Action Generation*, *Forward Models*, *Observer*, *Attention* (local) *Controller*, *Monitor*, *Goals.* The first five modules are implemented by models which can be adapted if erroneous performance is realised. We consider here that the Rules module is a Value Map that is created through reinforcement learning; 6. Partitioning the input and output spaces into suitable sub-sets. The input space is the sensory space, while the output space is the action space. We extend this model to include in the local scope an additional attention process that of the *Boundary Attention*. This process is responsible for raising an attention event when we approach a limiting condition in the scope of a goal. For example when the power level is low this can be considered as a boundary condition that must capture attention and thus increase the priority of the goal. This relates to general *homeostasis* mechanisms of biological agents. When a homeostatic variable moves near (or out of) the boundary of its preferred range then attention is raised so that appropriate corrective action will be taken.

3.2 Policy Acquisition for Multiple Goals

Our proposed solution for multiple goals policy acquisition is based on the following ideas: i. Use of attention allows the agent to have fast reaction and deal with novel and unexpected situations that develop in a time scale faster than that used for single-goal policy learning; ii. Instead of learning an overall (joint) policy for the current set of goals directly it is simpler to combine individual goal policies to an overall strategy. Learning an overall policy seems unlikely in biological agents as one has to store a value map (of the policy) for each combination of goals ever encountered; iii. We effectively acquire an overall policy by selecting at each time instance only one active goal and the action output of the agent is the action selected in the scope of the active goal. The learning of the goal's policy takes place in an individual basis using a standard RL method; iv. The scheme for selecting priorities for a set of competing goals is based on the following formula:

$$\text{ActInd}=(\text{W}+\text{S-AI}+\text{M-AI}+\text{B-AI}+\sum\text{ActInd}) \,/\, (4 + \#\ \text{Contributing Children}) \qquad (1)$$

Formula (1) is an extension of the corresponding formula in [4]. ActInd is the action index of goal, which effectively controls the priority of the goal in global competition. W is the "intrinsic" weight. S-AI is the sensory attention index (to capture novel and unexpected situations), M-AI is the motor attention index (to capture impending dangers), B-AI is the boundary attention index discussed in 3.1. All attention indices and the intrinsic weight are bounded in [0,1]. The sum of Action Indices in the numerator is over all contributing children in the sub-tree of a goal. A child is contributing if any of its corresponding attention indices is non-zero. This allows for the propagation of attention events in the goal hierarchy; v. Non-competing goals (due to referring to a different sub-region of the action space) are processed in parallel.

4 Simulation Results

We use the setting described in section 2, the set of goals in Fig. 1, formula (1) for selecting goal priorities and we define the sensory and motor attention indices as in [4,5]. The boundary attention index is defined as a sigmoid function over the value map of the energy states of the agent, and it is given by (2):

$$\text{B-AI}=1/(1+\exp[\ \mathrm{Val}(E(t)-E_{max}/2)\ ]\ ) \qquad (2)$$

where Val(·) is the value for the corresponding energy state (as the sum of all future discounted rewards) – it takes negative values for negative energies -, E_{max} is the maximum energy level (1000 units) and the energy value map has been acquired as all other value maps using the Q-learning algorithm [6]. The general methodology for training was as follows: We first trained the agent in each sub-problem separately using the Q-learning method with parameters of a=0.1 and γ=0.9 for learning and discount rate respectively. Then we allowed the existence of multiple goals concurrently. The selected action at each time step was determined by the currently active goal by using its own value map acquired during the separate training. The active goal is the goal which wins the attentional competition. The maximum number of training sessions for learning a value map was a million iterations. The intrinsic weights in (1), which code the relative importance of goals are given as relative ratios: $|\mathrm{Val}_G|/\sum |\mathrm{Val}_G|$ of the values for each goal, in our case: +20 for Transport (point B), +10 for Power Monitor (point C), -10 for MObj collision, -5 for SObj collision, +1 for Obj+ collision, -1 for Obj- collision and 5 for Collision Avoidance respectively. When approaching one of SObj, MObj, Obj+, Obj- closer than a threshold range R=3 cells, we calculate the S-AI and M-AI indices as described in [4,5]. Adding their contribution to (1) allows the alternation of priorities of goal execution and thus the determination of the currently active goal. The size of the GridWorld was 50x30 cells. We assume that when the agent reaches point B it then returns to A for starting another Transport action. It continually expends energy. We run 50 simulations to check the probability of

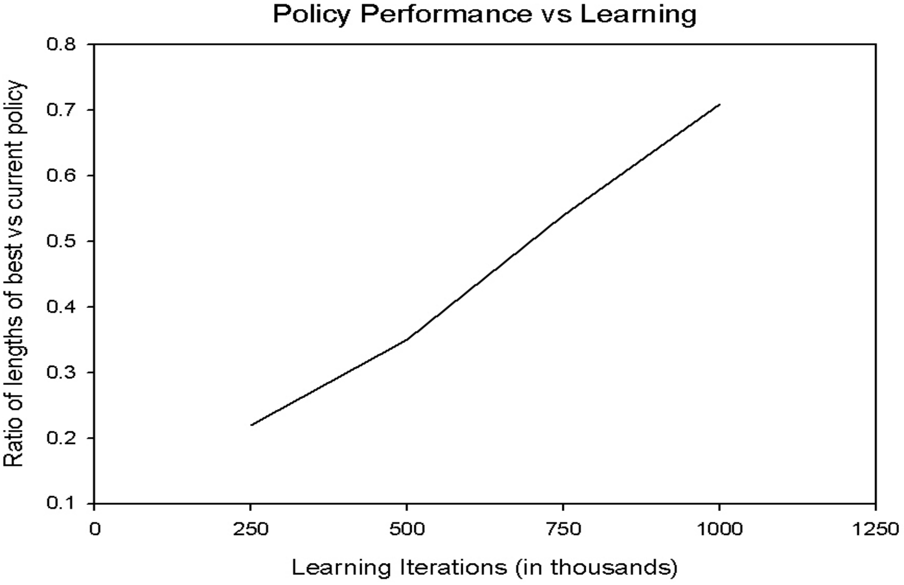


Fig. 3. Ratio of path lengths (A-B-C-A) of "best" vs current overall policy against the number of iterations for learning a policy. The path includes a *recharge* event and the current policy lengths are averaged over 50 simulations.

collisions and switching from the Transport goal to the Power Monitor goal. Each simulation session included the execution of 10 Transport commands (so as to deplete the energy and force a recharge). The results show that the agent successfully switched to the appropriate goal's value map for action selection in all cases. In some cases collisions with moving objects took place due to false predictions regarding the future position of the objects. With further training of the predictive models for the motion of moving objects, the collisions are decreased, as it was also described in [5]. We used from 5-15 moving objects having different paths per simulation as in [5]. The overall performance for the agent is shown in Fig. 3. We show a curve that depicts the ratio of lengths of path A-B-C-A for the best/current policy against the number of training iterations for acquiring the policy (250K, 500K, 750K and 1M). The "best" policy was determined by us empirically using the optimal action at each time step. The curve is drawn by using the total length of the overall curve from point A to B and back. We used only the curves during which we had a recharge event and we averaged the lengths over the 50 simulation sessions. It is clear that as the learnt individual goal policies approach their optimal targets the actual length comes closer to the shortest length. In Fig. 3 all policies corresponding to goals were trained to the same level.

Acknowledgments

We would like to acknowledge the support of the European Union through the FP6 IST GNOSYS project (FP6-003835) of the Cognitive Systems Initiative to our work.

References

1. Sutton R. & Barto, A. (2002). Reinforcement Learning: An Introduction, MIT Press 2002, 4th Ed.
2. Rushworth M. F. S. et al (1997). The left parietal cortex and motor attention. Neuropsychologia 33, 1261-1273.
3. Taylor J. G. (2000). Attentional movement: the control basis for Consciousness. Soc. Neuroscience Abstracts 26, 2231, #893.3
4. Kasderidis, S., & Taylor, J.G. (2004). Attentional Agents and Robot Control, International Journal of Knowledge-based and Intelligent Systems 8, 69-89.
5. Kasderidis, S. & Taylor J.G. (2004). Attention-based Learning, International Joint Conference on Neural Networks (IJCNN 2004), Budapest 25-29 July 2004, p.525-531.
6. Mitchell T. (1997). Machine Learning, McGraw Hill.

Neural Network with Memory and Cognitive Functions

Janusz A. Starzyk[1], Yue Li[1], and David D. Vogel[2]

[1] School of Electrical Engineering and Computer Science,
Ohio University, Athens, OH 45701, U.S.A
starzyk@bobcat.ent.ohiou.edu
[2] Ross University School of Medicine, Commonwealth of Dominica
Dvogel@rossmed.edu.dm

Abstract. This paper provides an analysis of a new class of distributed memories known as R-nets. These networks are similar to Hebbian networks, but are relatively sparsly connected. R-nets use simple binary neurons and trained links between excitatory and inhibitory neurons. They use inhibition to prevent neurons not associated with a recalled pattern from firing. They are shown to implement associative learning and have the ability to store sequential patterns, used in networks with higher cognitive functions. This work explores the statistical properties of such networks in terms of storage capacity as a function of R-net topology and employed learning and recall mechanisms.

1 R-Nets Neural Network Organization

1.1 Main Concept of R-Nets

R-nets have been used as components in the modular construction of larger networks capable of computations analogous to serial memory, classical and operant conditioning, secondary reinforcement, refabrication of memory, and fabrication of possible future events. R-net components of these larger networks appear to be appropriate objects of more detailed analysis than has been performed [Vogel, 2005].

R-nets stress biological plausibility and have demonstrated large storage capacities with the sparse connectivity of mammalian cortex. The number of synapses of principal cells on interneurons is at least 320 [Sik, Tamamaki, & Freund 1993]; the number of synapses of interneurons on principal cells is 1000 to 3000 [Freund & Buzsáki, 1996] and the ratio of interneurons to principal cells is roughly 0.2. The R-net modeled for this paper has 40% of excitatory neuron pairs linked though at least 1 inhibitory neuron [Vogel, 2005]. The detailed network structure is described in previous studies [Vogel, 2005]. The biological plausibility of this arrangement is discussed by Vogel [2001, 2005] and also by Fujii, Aihara, and Tsuda [2004].

Mathematically, R-nets are defined as randomly connected artificial neural networks with primary and secondary neurons. The network structure and connections between primary and secondary neurons are discussed by Vogel [2001, 2005].

R-nets implement distributed memories able to recall input patterns. During training, an input pattern is presented to the R-net by activating a selected cluster C of primary neurons. All links between active primary neurons are trained. During recall a subset of one of the stored patterns is presented to the input, activating corresponding primary neurons (initial recall set). The initial recall set is expected to activate all

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 85 – 90, 2005.

neurons of one of the stored patterns that include the activated neurons as a subset. Each primary neuron is activated if it is not inhibited.

During recall, inhibitory neurons linearly sum the weighted projections.

$$a_{i,x} = \sum W_{i,e} a_{e,x} \tag{1}$$

where $a_{i,x}$ represents the activity of the ith inhibitory neuron on the xth cycle, $a_{e,x}$ is the current activity of the eth excitatory neuron with possible values 0 or 1, and $W_{i,e}$ is the weight of the projection of the eth excitatory neuron onto the ith inhibitory neuron with possible values of 1(untrained) or 10(trained). The excitatory neurons are then synchronously updated according to the rules [Vogel, 2005].

2 Statistical Model of R-Nets

A series of papers [Vogel and Boos, 1997; Vogel, 1998; Vogel 2001] demonstrated the substantial storage capacities of networks progressively approximating the R-nets. An R-net with 10^6 excitatory neurons and brain-like connectivity will store at least $2x10^8$ bits of information [Vogel, 2005]. In this section, a statistical model of R-nets is presented and is compared with simulated R-nets.

Let us assume that an R-net is characterized by the set of primary neurons ***P***, the set of secondary neurons ***S***, the primary neurons' outgoing sets, $\boldsymbol{k_p}$, and the secondary neurons' outgoing sets, $\boldsymbol{k_s}$. These numbers are related through $P * k_p = S * k_s$.

Let us define P_{ci} as a set of primary neurons reachable from a primary neuron C_i through the secondary neurons, and α as probability that $c_j \in P_{ci}$ for a selected primary neuron c_j and a given P_{ci}, so that $\alpha = \frac{P_{ci}}{P}$. The expected value of the number of different primary and secondary neurons reaching to (or reached from) a secondary and a primary neuron, are respectively

$$k_{sp} = P * \left(1 - \left(1 - \frac{1}{P}\right)^{k_s}\right) < k_s, k_{ps} = S * \left(1 - \left(1 - \frac{1}{S}\right)^{k_p}\right) < k_p \tag{2}$$

If the links through other secondary neurons reached from a primary neuron are considered, the number of primary neurons linked to a given primary neuron is

$$P_{ci} = P * \left(1 - \left(1 - \frac{1}{P}\right)^{k_s k_p}\right) < k_s k_p . \tag{3}$$

2.1 Eliminating Spurious Neurons

Spurious neurons are defined as neurons that are not a part of the original pattern and that are activated during the recall process. The probability that a potential spurious neuron, $\boldsymbol{c_j}$, will be inhibited depends on the probability of the existence of an inhibitory link from an activated primary neuron.

The probability that a projection out of a primary neuron in a trained set is trained with T patterns stored in the R-net is estimated as:

$$P_{t1} = 1 - \left(1 - \frac{C}{P}\left(1 - \left(1 - \frac{\alpha}{k_{ps}}\right)^{C}\right)\right)^{T} \tag{4}$$

P_{t1} also equals $P_{t2,}$ the probability of a projection out of a secondary neuron being trained with T patterns stored.

As shown in Fig. 1, a neuron is spurious if it meets both of the following conditions: a) It has no projection from S_{wa}; and b) All its projections from S_{sa} are trained, where S_{sa} is the strongly activated set of secondary neurons and S_{wa} is the weakly activated set of secondary neurons.

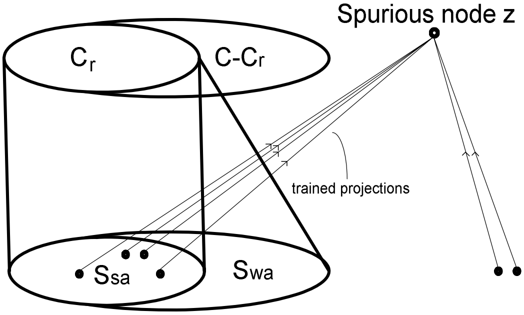


Fig. 1. Spurious neurons and activated secondary

The probability that a secondary node, y, belongs to S_a is the same as the probability that a selected secondary neuron is active during recall.

$$p_{asr} = 1 - \left(1 - \frac{C_r}{P}\right)^{k_{sp}} \tag{5}$$

Thus the probability that a node in S_a is strongly activated is approximately

$$P_{ssa} = 1 - (1 - P_{t1})^{(k_{sp}-1)P_{sr}+1} \tag{6}$$

and consequently the probability that a node in S_a is weakly activated is $1 - P_{ssa}$.

The probability that a potentially spurious node, z, is not linked to any node in S_{wa} is

$$P_{nwa} = (1 - P_{asr}(1 - P_{ssa}))^{k_{ps}} \tag{7}$$

Assume that a node z is not linked to S_{wa}. The probability that such primary node z is connected to a node in S_{sa} is

$$P_{pssa} = \frac{P_{asr}P_{ssa}}{1 - P_{asr}(1 - P_{ssa})} \tag{8}$$

Using this result, we can obtain the probability that a primary node z has k projections to S_{sa} and no projections to S_{wa}, and all of the links to S_{sa} are trained, thus obtaining the probability that z is a spurious node as

$$P_z = P_{nwa} \sum_{k=0}^{k_{ps}} \binom{k_{ps}}{k} P_{pssa}^k (1 - P_{pssa})^{k_{ps}-k} P_{t1}^k \tag{9}$$

The increasing probability of spurious neurons with increasing numbers of patterns stored limits the maximum number of patterns that can be stored in the R-net memory. Since, in the recall process, we can tolerate no more than S_{max} spurious neurons, and each neuron in the P-C set has probability of being spurious equal to P_z, then the recall set has no more than S_{max} spurious neurons with the probability

$$P_{NO\_spurious} = \sum_{i=0}^{S_{max}} \binom{P-C}{i} P_z^i (1 - P_z)^{P-C-i} \tag{10}$$

The analysis is in reasonable agreement with actual simulations of modestly large networks, and anticipate that increasing the size of both the networks and the subsets used for recall will only make the network stochastically smoother and the analysis more accurate. This anticipation does not replicate the error found in Marr [1971] (discussed in [Vogel, 2005]).

2.2 Eliminating Missing Neurons

A missing neuron is a neuron from C-C_r which is suppressed by an inhibitory projection to an activated primary neuron. The following lemma can be established.

Lemma: Each missing neuron is suppressed by an inhibitory link to a spurious neuron connected through a secondary node **w**, where **w** is different from all nodes in S_a, as shown in the Fig. 2.

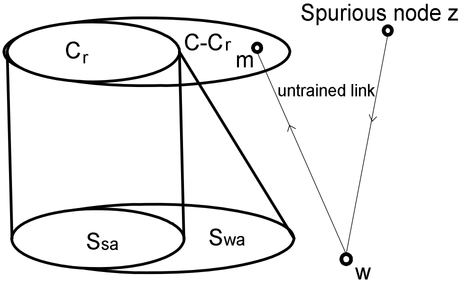


Fig. 2. Creation of missing neurons

Proof: If **m** connects to S_a, then there is a node **x** in C_r such that **x** and **m** are linked. Since m and **x** are a part of the same pattern, both parts of their link (projections from the primary node **x** to the secondary node in S_a and from the secondary node in S_a to the primary node **m**) are trained. Obviously the inhibition cannot result from such link. Therefore, the inhibitory link to **m** must pass through a secondary node outside of S_a.

To prove the argument that inhibition must come from a spurious neuron we may notice that no neuron in C_r can be connected to **w** and that if a node **x** in C is connected to **w** then m and x are connected through a trained link, since they are in the same pattern C. Therefore no other node in C can inhibit m. This leaves, as the only option that an inhibitory link to m comes from a spurious node outside of C.

The probability that a given primary node will be missing due to a single spurious node can be estimated to be less than

$$P_{mis\sin g\_one} = \frac{1}{P}(1 - P_{t1}P_{t2})k_{ps}k_{sp}(1 - P_{asr})P_z \tag{11}$$

By connecting all possible locations of missing neurons, the probability that a single primary neuron is missing is

$$P_{mis\sin g} = 1 - (1 - P_{mis\sin g\_one})^{P-C} \tag{12}$$

So the probability that the recall set has no more than S_{max} missing neurons is

$$P_{NO\_mis\sin g} = \sum_{i=0}^{S_{max}} \binom{C}{i} P^{i}_{mis\sin g}(1 - P_{mis\sin g})^{C-i} \tag{13}$$

2.3 Results of the Statistical Model

The statistical model is in a good agreement with simulated R-nets and can be applied to estimate the computational performance of very large R-nets. We simulated the storage capacity for 20 to 100 neuron patterns of networks up to 10^7 primary neurons with 1000 projections from each primary neuron to $2x10^6$ secondary neurons with 5000 projections from each back to primary neurons. Result is shown in Fig. 3.

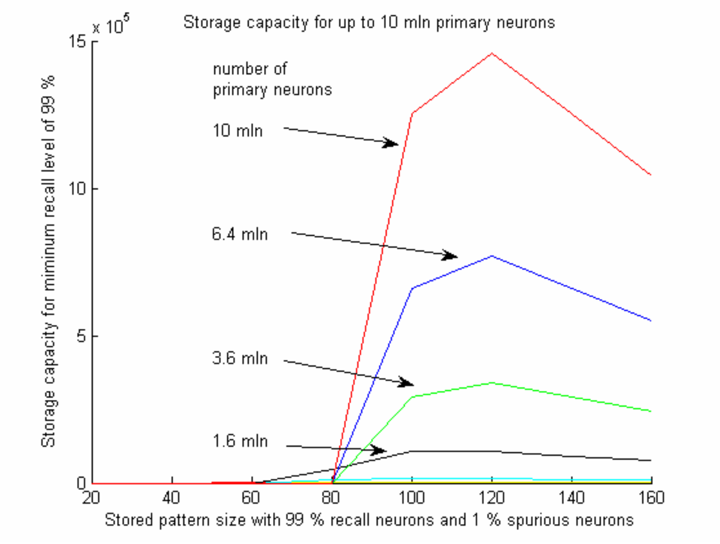


Fig. 3. Storage capacity of R-net with up to 10^7 primary neurons

In addition, from the conducted analysis of R-net properties based on the presented model, we can conclude that their storage capacity grows faster than the number of primary neurons and that a slope of growth is close to 10/7 on the logarithmic scale which agrees with experimental results reported in [Vogel, 2005]. When the network size reaches 10^9 primary neurons (with average number of projections 10^4 that is similar to interconnection density of human brain), the network can store over 10^9 patterns and the optimum storage for these very large memories is achieved with a pattern size of about 150 neurons.

3 Conclusion

In this paper a statistical model of R-nets was presented and results were compared to results observed in simulated R-nets. This model has already demonstrated that work on the role of disinhibition in paired-pulse induced long-term potentiation may be of fundamental importance to understanding memory and higher cognitive functions. It suggests an entirely new understanding of the role of massive projections of excitatory neurons onto neurons of distant regions [Vogel, 2005]. These models have produced computations analogous to serial memory, context dependent classical and operant conditioning, secondary reinforcement, refabrication of memory, and planning. They distinguish between perceived and recalled events, and predicate responses on the absence as well as presence of particular stimuli. Analysis suggests that the models may be expected to scale up to brain-sized networks efficiently.

References

Freund, T.F., and Buzsáki, G. (1996). Interneurons of the Hippocampus. *Hippocampus.* 6, 347-470.

Marr, D. (1971). Simple memory: a theory for archicortex. *Philosophical Transaction of the Royal Society of London, B.* 262, 23-81.

Sik, A., Tamamaki, N., and Freund, T.F. (1993). Complete axon arborization of a single CA3 pyramidal cell in the rat hippocampus, and its relationship with postsynaptic parvalbumin-containing interneurons. European Journal of Neuroscience. 5: 1719-1728.

Vogel, D., and Boos, W. (1997). Sparsely connected, Hebbian networks with strikingly large storage capacities. *Neural Networks* 10, 671-682.

Vogel, D. (1998) Auto-associative memory produced by disinhibition in a sparsely connected network. *Neural Networks.* 11: 897-908.

Vogel, D. (2001). A biologically plausible model of associative memory which uses disinhibition rather than long term potentiation, Brain cogn. 45, 212-228.

Vogel, D. (2005). A neural network model of memory and higher cognitive functions. International Journal of Psychophysiology. 55 (1), 3-21.

Associative Learning in Hierarchical Self Organizing Learning Arrays

Janusz A. Starzyk, Zhen Zhu, and Yue Li

School of Electrical Engineering and Computer Science,
Ohio University, Athens, OH 45701, U.S.A
starzyk@bobcat.ent.ohiou.edu

Abstract. In this paper we introduce feedback based associative learning in self-organized learning arrays (SOLAR). SOLAR structures are hierarchically organized and have the ability to classify patterns in a network of sparsely connected neurons. These neurons may define their own functions and select their interconnections locally, thus satisfying some of the requirements for biologically plausible intelligent structures. Feed-forward processing is used to make necessary correlations and learn the input patterns. Associations between neuron inputs are used to generate feedback signals. These feedback signals, when propagated to the associated inputs, can establish the expected input values. This can be used for hetero and auto associative learning and pattern recognition.

1 Introduction

Associative learning has been long recognized as one of the necessary elements of intelligence, thus it is desirable that an artificial system that mimics biological intelligence be able to perform both spatial and temporal associations. Associative networks were developed as a special class of artificial neural networks to handle associative learning and retrieval of information. There are two types of associative networks, hetero-associative (HA) and auto-associative (AA). Hetero-associative networks are capable of making associations between two or more different types of input signals. Auto-associative networks learn associations between elements of the same input vector. Such networks can learn various patterns, and then recall the pattern based on a fractional part of a pattern. Examples of HA networks include multilayer perceptron [1], the counter-propagation network [2], the bidirectional associative memory [3] and multi-associative spatio-temporal network [4], while the Hopfield network [5] and the Vogel associative memories [6,7] are AA. In this paper we present a model of the self-organizing learning array that implements both the hetero and the auto-associative learning.

Spatio-temporal associations are particularly important in both biological and electro-mechanical systems. For instance, a spatio-temporal association may trigger a reactive response in an animal or guide the robot to its target. Time delays have been used in Hopfield networks [5] to generate spatio-temporal sequences which are time dependent sequences of spatial patterns. Storage and retrieval of spatio-temporal sequences was studied in many papers ([8 - 10]). While the proposed approaches achieved reasonable storage and retrieval of input sequences, they have some serious drawbacks if one wants to implement them in biologically plausible structures. In this

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 91 – 96, 2005.

paper we take on a different approach to pattern storage and associations. A hierarchical, multilayer structure based on our self-organizing learning architecture [11] is used, and we demonstrate that such structure can make the necessary associations between patterns using sparsely connected neurons.

SOLAR (Self-Organized Learning Array) is a regular, two or three-dimensional array of identical processing cells, connected to programmable routing channels. Each cell in the array has ability to self-organize by adapting its functionality in response to information contained in its input signals. Cells choose their input signals from the adjacent routing channels and send their output signals to the routing channels. Like artificial neural networks (ANNs), SOLAR is inspired by the structure of biological neural networks and shares their robust, distributed and parallel signal processing, yet it differs from existing realizations of ANNs. It has a deep multi-layer hierarchical structure, which helps to handle complexity of target problems, it uses online learning with dynamically set neuron functions and dynamically learned sparse connections, efficient in hardware realization. Prior study of SOLAR structures reported in [11] concentrated on demonstrating its pattern recognition and classification abilities. In this paper we introduce a feedback mechanism with inhibitory connections and associative learning to SOLAR.

This paper has been organized in 4 sections. The second section discusses the structure and behavior of the proposed network. Section 3 presents testing results on several bench-mark machine learning problems. Section 4 contains conclusions.

2 Network Structure and Operations

In this work, the network has been formed as a two-dimensional structure, which is pseudorandomly constructed with interconnection structure of small world networks [12]. For a recognition task, it is trained with the input features that represent the patterns, and the corresponding codes that represent the classification. The input span, defined by the number of rows, is set equal to or greater than the dimensionality of inputs. The depth of the network (the number of hierarchical levels) is set according to the input span. In a hierarchical structure, each neuron connects only to the neurons of the previous layer. Once the learning is completed, a network is capable to make necessary associations, such that when presented with the pattern only, it drives feedback to the associated inputs to assert the unknown code values. Similar to pattern recognition, missing data can be found from feedback traced to the unknown portion of the input.

The outside input should be presented to the network in a binary form ranging from 0 to 1. The signal strength is measured as the distance between the signal level and 0.5. A signal is determinate if it is 0 or 1. It is a low (or high) if it is below (or above) 0.5, and is unknown or inactive if it is 0.5. The probabilities of I_1 and I_2 being low or high and their joint probabilities can be recorded in each neuron. The conditional probabilities $P(I_2 | I_1)$ and $P(I_1 | I_2)$ can then be computed.

A simplified confidence interval measure is used for each of the probabilities: $\mathrm{CI} = \frac{2(1 - P(I_2 | I_1))}{\sqrt{N}}$, where N stands for the number of training inputs. The value of $P(I_2 | I_1)$ - CI is then compared against a threshold τ. If larger, we can say that I_1 can be **implied** from I_2. Likewise, $P(I_1 | I_2)$ decides whether I_1 can be **implied** from I_2.

Definition: Inputs I_1 and I_2 of a neuron are **associated** if and only if I_2 can be implied from I_1 and I_1 can be implied from I_2 simultaneously. Such a neuron is then an **associative neuron**. Otherwise it is a **transmitting neuron**.

Fig. 1 illustrates six different situations of I_1 and I_2 inputs that an associative neuron may receive in training.

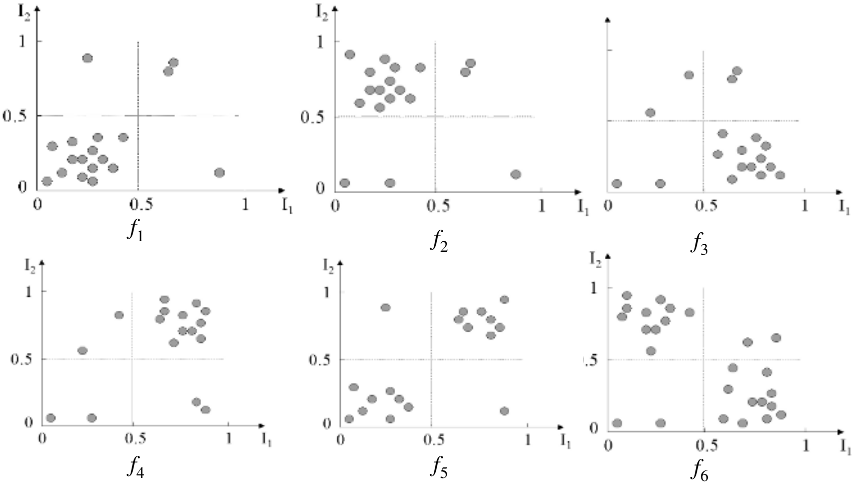


Fig. 1. Input Distribution to an Associative Neuron

2.1 Feed Forward Scheme

For the simplicity of discussion, we assume a fixed interconnection structure where each neuron has two inputs I_1 and I_2 and a single output O. The task of a neuron during training is to discover the potential relationship between the two inputs and to remember it. The neuron needs to select a proper transfer function f from a predefined set F that can best describe the relationship between I_1 and I_2. It can then generate output O using f. Six functions, f_1 to f_6, are designed to include all the logic relationships between I_1 and I_2 in an associative neuron, as shown in Fig. 1. In an associative neuron, the majority of the training data is either distributed in one **dominant quadrant**, or two diagonal quadrants. f_1 to f_4 are designed for the four possible locations of the dominant quadrant. Their output is always 1 for the dominant quadrant, and 0 for all the others. When data points are mostly distributed in two diagonal quadrants, f_5 and f_6 are used as shown in Fig. 1. To accommodate noise f_5 and f_6 are defined only based on I_1 to include all the data points. For example,

$$f_5(I_1, I_2) = \begin{cases} 0, & \text{if} \quad I_1 \text{ is low} \\ 1, & \text{if} \quad I_1 \text{ is high} \end{cases} \tag{1}$$

The neuron output is set “inactive” or 0.5, whenever either one of the inputs is 0.5.

$$O = \begin{cases} 0.5, & \text{if} \quad I_1 = 0.5 \text{ or } I_2 = 0.5 \\ f(I_1, I_2), & \text{otherwise} \end{cases} \tag{2}$$

If a neuron observes any distribution other than what is included in Fig. 1, it is a transmitting neuron. It simply transmits the input with higher entropy, called the **dominant input,** to O, with the other ignored. An input I_1 is the dominant input if

$$\text{abs}(P(I_1 \text{ is low}) - P(I_1 \text{ is high})) < \text{abs}(P(I_2 \text{ is low}) - P(I_2 \text{ is high})) \quad (3)$$

2.2 Feedback Scheme

During testing, missing parts of the data need to be recovered from existing data through association. For example, in a pattern recognition problem, the neurons that are physically connected to the unknown code inputs are responsible for providing feedback from the associative neurons and define the values of the code. This, in turn, can be used either to classify the input pattern or to recover the uncertain inputs.

The feedback scheme is an important part of associative learning. Fig. 3(a) shows a conceptual view of the network with separated known and unknown inputs. The white circles are the neurons that do not participate in signal processing. The black circles are **actively associating neurons** defined below. The gray circles are the remaining neurons involved in signal processing. The neurons on the known side generalize the information that activates associative neurons, which generate feedback to the unknown side. In order to explain the working mechanism in single neurons, Fig. 2(b) shows a snapshot of the communication among four interconnected neurons.

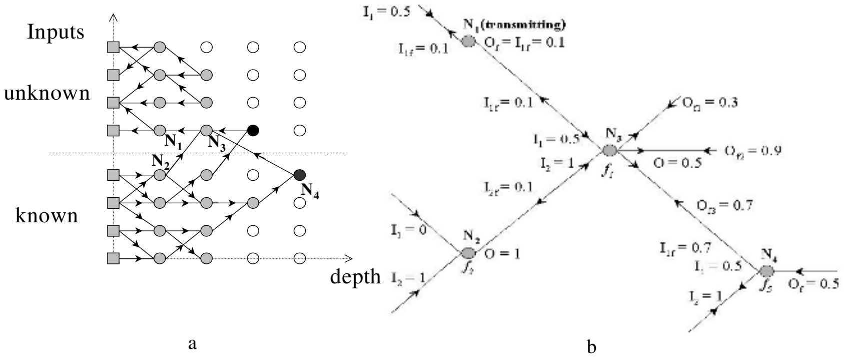


Fig. 2. Neuron Feedback Scheme

When a neuron receives at least one active output feedback, the strongest feedback O_f triggers the feedback to the neuron's inputs. The input/output relationship in the feedback scheme for each neuron can be described by one of the following types.

1. Transmitting neurons. A transmitting neuron (e.g. N_1 in Fig. 2,) simply passes O_f back into its dominant input. When the feedback I_{1f} is stronger than the dominant input I_1, I_1 will be overwritten by I_{1f}.

2. Associative neurons with determined inputs. If I_1 and I_2 of an associative neuron (e.g. N_2 in Fig. 2,) are either 0 or 1, O will consequently be at full strength. O_f won't be able to change O. Feedback takes no effect and information passes forward.

3. Associative neurons with active feedbacks and inactive input(s). For an associative neuron that doesn't have determinate signals on both inputs (e.g. N_3 in Fig. 2,) O_f creates feedbacks I_{1f} and I_{2f} through the function *f*. If the feedback signals are stronger than the original inputs, these inputs will be overwritten. Consequently, overwritten inputs become feedback signals to the neurons N_1 and N_2, to which N_3 inputs are linked. These neurons pass information backwards and they are not allowed to propagate forward to higher hierarchical layers.

4. Associative neurons with inactive feedbacks. Some neurons located deeply in the network may not receive active feedback at all, (e.g. N_4 in Fig. 2). If one of their inputs is inactive, it will be overwritten based on its association with the other input and the neuron function *f*. These type of neurons are called **actively associating** and are the backbone of the associative processing in SOLAR. For instance, since I_1 is 0.5 for N_4, the feedback to I_1 is determined based on the known input I_2 and the function f_5. For neurons that fit scenarios 3 and 4, the input feedback is calculated differently for each function, based on the strength of O_f and the quality of each neuron's learning, which is not described in full details in this paper.

3 Simulation Results

Several benchmark classification and missing data recovery tasks have been used to test the performance of the proposed network.

Teaching Assistant Evaluation database [13] consists of 151 instances, 5 features and 3 equally sized classes. After a 15-cross validation, the overall correct classification rate of SOLAR is 68.33% compared to 67% in [13].

SOLAR was also tested with the Iris database [14], which has 3 classes, 4 numeric attributes and 150 instances. The hierarchical SOLAR network gets an average classification rate of 75.33% from a 15-cross validation. An optimal input arrangement (using straight sliding bars and merging features and class id code) could further improve the performance to 86%. For comparison, results reported in literature [15] give correct classification rate for the Iris database between 91.33% and 97.33%.

The Glass Identification Database [16] was used to study the impact of the target problem's complexity on the depth of the network. The network was first trained and tested with the whole database, which contains 6 classes and 9 features, and then with half of the database that only has 3 classes. It was found that the more classes were used, the more layers SOLAR needs in its hierarchical structure.

In addition, the network has successfully accomplished binary image recovery tasks. Although the current setup uses a two dimensional architecture, it is believed that a three dimensional network would handle image related problems better.

4 Conclusions

This paper presents an associative learning network based on a hierarchical SOLAR structure. SOLAR is a biologically inspired machine-learning concept. It is a sparsely connected network organized as a fixed lattice of distributed, parallel processing units (neurons). The associative learning SOLAR network described in this paper is constructed as a fixed connection network with feedback and inhibitory links. Similar to Vogel's distributed auto-associative memories [7], SOLAR discovers the correlation between inputs and establishes associations inside the neurons, without a need to differentiate between the associated classification code and data patterns. It is capable of handling a wide variety of machine learning tasks including classification and data recovery, and is suitable for online learning. The SOLAR organization will be further modified towards an advanced machine intelligence system capable of associative learning, adaptations, and value driven interaction with environment.

References

[1] D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, *Parallel Distributed Processing*, Cambridge, MA: MIT Press, 1986.

[2] R. Hecht-Nielsen, *Neurocomputing*, Reading, MA: Addison-Wesley, 1990.

[3] B. Kosko, "Bidirectional associative memories," IEEE Trans. Syst., Man, Cybern., vol. 18, pp. 49-60, Jan./Feb. 1988.

[4] L. Wang, "Multi-associative Neural Networks and Their Applications to Learning and Retrieving Complex Spatio-Temporal Sequences," IEEE Trans. Syst., Man, Cybern., vol. 29, no. 1, Feb. 1999.

[5] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," in Proc. Nat. Acad. Sci. USA, Apr. 1982, vol. 79, pp. 2554-2558.

[6] D. Vogel and W. Boos, "Sparsely connected, Hebbian networks with strikingly large storage capacities," Neural Networks 1997; 4(10): 671.682.

[7] D. Vogel, "Auto-associative memory produced by disinhibition in a sparsely connected network," Neural Networks 1998; 5(11): 897-908.

[8] I. Guyon, L. Personnaz, J. P. Nadal, and G. Dreyfus, "Storage and retrieval of complex sequences in neural networks," Phys. Rev. A, vol. 38, pp. 6365-6372, 1988.

[9] D. L. Wang and B. Yuwono, "Anticipation-based temporal pattern generation," IEEE Trans. Syst., Man, Cybern., vol. 25, pp. 615-628, Apr. 1995.

[10] L. Wang, "Multi-associative neural networks and their applications to learning and retrieving complex spatio-temporal sequences," IEEE Trans. Syst., Man, Cybern., Part B, vol. 29 , no. 1 , pp. 73 - 82, Feb. 1999.

[11] J. A. Starzyk, Z. Zhu and T.-H. Liu, "Self-Organizing Learning Array," IEEE Trans. on Neural Networks, vol. 16, no. 2, pp. 355-363, March 2005.

[12] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," Nature, vol. 393, pp. 440-442, June 4, 1998.

[13] Wei-Yin Loh, Teaching Assistant Evaluation, June 7, 1997. Available:
http://www.stat.wisc.edu/~limt/mach1317.pdf

[14] R.A. Fisher, Iris Plants Database, July, 1988. Available:
http://faculty.cs.byu.edu/~cgc/Teaching/CS_478/iris.arff

[15] Razvan Andonie, Angel Cat¸aron, "An Informational Energy LVQ Approach for Feature Ranking", The European Symposium on Artificial Neural Networks (ESANN 2004), Bruges, Belgium, April 28-30, M. Verleysen, Ed., D-side publications, 2004, pp. 471-476.

[16] B. German, Glass Identification Database, September, 1987. Available:
http://www.ailab.si/orange/doc/datasets/glass.htm

A Review of Cognitive Processing in the Brain

John G. Taylor

Department of Mathematics, King's College, Strand, London WC2R 2LS, UK
john.g.taylor@kcl.ac.uk

Abstract. A review of cognitive processing in the brain is presented, using insights from brain science to extract general principles. After discussing the nature of cognition, the triumvirate of basic brain functions: attention control, reinforcement learning and memory, are explored as to how they could support cognitive processes, from which certain cognitive principles are developed. Several specific cognitive tasks are discussed and their simulations considered which support the effectiveness of the approach.

1 Introduction

A global control system is undoubtedly needed to organize overall coherence between parts of the brain performing different functions in cognition. There is now good evidence that attention is such a system, acting so as to filter out distracters from necessary targets in the cluttered world, or emphasize certain actions out of many possible. The distinguishing features of attention have become clear through many brain imaging and single cell experiments using a range of experimental paradigms. So attention is an important component to be included in cognitive models.

Attention needs to have a value system to guide what it is worth attending to and process to a higher level. Memory of reward and penalty is also needed. The nature of the storage of reward and penalty in the brain is being systematically uncovered through brain science, with the role of dopamine and of various limbic sites becoming clarified. A powerful link-up with machine learning in the guise of temporal difference and adaptive critic (ACE/ASE) learning has developed apace, so that the TD/ACE/ASE system has good evidence for being used in the brain.

Beside reward memory there is also need for episodic memory in cognitive processing. Again considerable progress has been made in understanding a number of the features of this, associated especially with the hippocampus, so both reward and memory can be used as corner-stones for cognitive modeling.

In the next section are given a brief discussion of cognition. Extraction of a general architecture and principles for a cognitive agent is given in section 3, after a brief discussion, and on the basis of, attention reward and memory. It is then shown in section 4 how this framework can be used in specific cognitive tasks; the paper finishes with conclusions.

2 The Nature of Cognition

The basic components of cognition will be taken here to be: awareness, thinking, knowing, reasoning and executive functions. Awareness presents an enormous

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 97 – 102, 2005.

difficulty, since there is no universally accepted model of its nature, nor we do even understand it. A model of awareness, developed from a control model of attention using engineering control ideas, will be adopted here [1, 6]. The claim that awareness is properly included in such a model has been argued in detail elsewhere [6, 7], and will not be considered further here; some such indication is needed for any self-respecting model of cognition to be worth its salt.

An important aspect of engineering control is a forward model involving the process of prediction:

$$x(t+1)=F[x(t),u(t) \qquad (1)$$

where x(t) is the estimated state of the controlled system at time t, and u(t) is a control signal at that time, which through the forward model function F[,] leads to an estimate of the state at the next time step. Purely sensory 'thinking' can be envisaged as arising from the recurrent running of the above forward model, using control signals u(t) from, say, the attention control system. The sequence would be concluded when a suitable goal state had been reached (as set up initially as part of a reasoning problem). More general thinking and reasoning can be achieved using a sensory-motor coupled system, with state estimate x being the attended sensory and motor state and the forward model run by joint sensory-motor attention control signals (which arise from separate parts of the brain, sensory attention in the right hemisphere, motor attention from the left). The forward and related models will involve memory codes, so as to be more efficient by using previous information of the world.

3 Attention/Reward/Memory for Cognition

Attention: This arises from a control system in higher order cortex (parietal and prefrontal) generating a signal to amplify a specific target representation in posterior cortex. The higher cortical sites generating the control signal (inverse model for attention movement) use a bias from prefrontal goals (held in working memory) to amplify (by contrast gain) posterior activity in semantic memory sites (early occipital, temporal and parietal cortices). This leads to the following ballistic model of attention control:

Goal bias (PFC)->Inverse model controller (Parietal lobe)->Lower Posterior CX (in various modalities) (2)

The amplified target activity is then able to access buffer working memory sites in posterior cortices (temporal and parietal) which act as attended state estimators. The access to this buffer has been modeled in the more extended CODAM model [1, 2] as a threshold process, arising from two-state neurons being sent from the down to the up-state (more specifically by two reciprocally coupled neurons almost in bifurcation). The access to the sensory buffer is aided by an efference copy of the attention movement control signal generated by the inverse attention model sited in the brain in the SPL. The existence of an efference copy of attention was predicted as being observable by its effect on the sensory buffer signal (as represented by its P3) [3]; this has been observed in an experiment on the Attentional Blink, where the N2 of the second target is observed to inhibit the P3 of the first when T2 is detected [4].

Reward/Value: An organism tries to optimise its experience in the world by taking actions on stimuli which maximise its future rewards. The signal for reward, and in particular reward prediction error, has been observed as carried in the limbic brain by dopamine, with predicted future reward values of stimuli encoded in the OBFC. A machine learning model of this has been developed under the guise of TD learning and the ACE/ASE system [5]. It allows a further bias to be exerted on the attention control system and resulting responses; it also provides a method of enhanced learning of stimulus representations as valued goals in prefrontal cortex.

Memory: There are a range of forms of memory: short-term, long-term, procedural, declarative, and so on. We have already considered short-term (working) memory as part of CODAM (in both parietal lobes as a buffer system and in prefrontal cortex as part of the executive system). Thus we consider only long-term memory, which is based on the hippocampus (HC). Numerous models of the HC cell fields exist, as well as models with more extensive connections to prefrontal cortex. HC stores memories in at least two ways: in afferents to CA1, and in recurrent connections in CA3. These are both stored during theta wave activity in HC, and then played back in the HC in SWS; finally it appears that playback to cortex, possibly in REM sleep, leads to a more permanent code, both of episodic and semantic memories. These form the basis of knowledge codes.

Cognitive Sub-tasks: In cognition a variety of subtasks are carried out, including: 1) Storage and retrieval of memories in hippocampus (HC) and related areas; 2) Rehearsal of content working memory; 3) Transformation of buffered material into a new, goal-directed form (such as spatial rotation of an image held in the mind); 4) Inhibition of pre-potent responses [8]; 5) The learning of forward maps of attention in both sensory and motor modalities, so that consequences of attended actions on the world can be imagined; 6) Determination of the reward value of sequences of sensory-motor states as they are being activated in forward model recurrence; 7) Learning automatic sequences (chunks) to speed up cognitive processing.

The rehearsal, transformation, inhibition and retrieval processes are those that can be carried out already by a CODAM model [1, 2, 3] (with additional HC for encoding & retrieval). CODAM can be used to set up a goal, such as the transformed state of the buffered image, or its preserved level of activity on the buffer, and transform what is presently on the buffer, by the inverse attention controller, into the desired goal. Such transformations arise in CODAM by monitoring if the original image has been correctly transformed or preserved under an attention feedback signal, generated by a suitably created error signal. Longer term storage of material for much later use would be encoded in the HC, under attention control. The comparison process involves yet again monitoring in CODAM. The use of forward models like (1) allows for careful planning of actions and realization and valuation of consequences. Multiple recurrence through the forward models of the form (1) allow further look-ahead, and prediction of consequences of several action steps. Automatic process are created by sequence learning in the frontal cortex, using basal ganglia as well as cerebellum involvement, to learn chunks. Attention agents have been constructed {13], recently combined with reward learning [14] involving look-ahead through forward models.

Cognitive Principles: We can deduce some principles of cognitive processing:

P1: There is overall control by attention of the cognitive process, using attention-based control signals to achieve suitable transformations to solve cognitive tasks;

P2: Fusion of attention control (in parietal lobe and PFC) and long-term learning in HC achieves an expanding set of stimuli and actions, and their attention control;

P3: The creation of a valuation of goals occurs in PFC to handle reward prediction biasing of the processes 1) to 6) above;

P4: Transformations on buffered stimuli is by suitable PFC goals for required transformations being carried out on buffered activities, under attention control;

P5: Forward models (as equation (1)) are created under attention control so, by recurrence, passage through sequences of attended sensory-motor states is achieved (as in thinking), to reach a desired goal (as in reasoning) or valued states that may correspond to new ways of looking at a problem (as in creativity);

P6: There is creation of, and ability to access, automated 'chunks' of knowledge, so they can be inserted into forward model sequences under P5. They are initially created by effortful attention at an earlier time (using error-based learning in cerebellum) but then gradually transferred to automatic mode by suitable rehearsal;

P7: Attention is employed as a gateway to consciousness, which provides an overarching control function of speed-up of attention, with important survival value.

Cognitive Architecture: A possible architecture is a) CODAM as an attention controller (with sensory and motor forms) b) Extension of CODAM by inclusion of value maps and the reward error prediction delta, as begun in [11]; c) Extension of CODAM to include an HC able to be attended to and to learn short sequences d) Further extension of CODAM by addition of cerebellum to act as an error learner for 'glueing' chunked sequences together, with further extension to addition of basal ganglia to have requisite automated chunks embedded in attended control of sequential progression. The goal systems in PFC are composed of basal ganglia/thalamus architecture, in addition to prefrontal cortex, as in [12].

4 Modelling Cognitive Tasks

The tasks I will consider are the Wisconsin Card Sorting (WCST) and the Tower of Hanoi. These are used to investigate prefrontal deficits in cognitive processing.

WCST: The WCST task has a pack of cards with 1, 2, 3 or 4 shapes of 4 kinds, with each card being in one of 4 colours. Four test cards lie face up on the table. The subject's task is to take a card from the pack and place it on one of the four test cards according to a rule (following the matching of colour, shape or number). If the choice fits a rule chosen by the experimenter, unbeknownst to the subject, then the experimenter says 'yes', otherwise 'no' . Either way the subject has to try again. After (usually) 10 correct choices by the subject, the experimenter changes the rule, although without telling the subject except inthe 'yes' or 'no' response they give to the subject's choices.

Numerous neural models of this task exist, such as [9]. This uses a set of recurrent loops (CX ? BG ? Thalamus ? CX) so as to store in working memory the rule presently in use by the subject. Receipt of a 'yes' continues the rule; a 'no' causes the WM activation to be destroyed by the amygdala, acting as a punishment device, and a new rule is then formulated and tried until the correct new rule is discovered. This model captures the essential components of the process, and fits reasonably well with brain imaging data ([9]. However the results in [10] indicate the need for further development of the recurrent prefrontal lobe loop structure to be more completely implemented.

Tower of Hanoi: This task involves a set of vertical rods arranged, for example, in a line, with rings of increasing radius that can fit over each rod. It is not allowed to have a larger ring above a smaller one on any rod, and only one ring can be moved at any time. The purpose is for the subject to work out the smallest number of moves to take the rings from one allowed configurations of rings to another only making allowed moves. A standard initial goal state is all the rings on one rod (in a legal configuration), with the final goal state they being all on another rod (again in a legal configuration); the task usually has only three rods and 3-5 rings.

There are several executive processes involved in solving the problem: WM encoding of the present position and of the ultimate goal state; transformation by a legal move to a further arrangement of rings from the initial one; continued transformation from one configuration to the next by a legal move: Config(n)?Config(n+1); achievement of the goal state by a final configuration reached, Config(N); if not achieved start a new set of legal moves from the initial state; repeat until successful. There are more abstract approaches to this problem but we are considering here how the task would be solved by a person on first acquaintance with it. Thus the most crucial step is the manipulation of Config(n)?Config(n+1) in WM, using a suitable legal move as represented in PFC. This corresponds to taking a mental image (in WM) of Config(n), and making a mental legal move of one ring to the top of the set of rings on another rod, in which the final state Config(n+1) is also legal. Thus a check has to be made that such a move is legal by performing the move and then checking that there are no smaller ring below the moved ring on the new rod. If not, the move is legal and is made (and also stored in memory); if there is a smaller ring then the move is not taken and a new move is tried for its legality. We note that in the above processes, attention and memory play a crucial role. Reward is involved in setting up the goal, but is not necessarily needed unless complete autonomy is needed for the agent to be suitable motivated.

5 Conclusions

After a discussion of the nature of cognition in section 2, we considered in the following section some details of the brain basis of the three basic functions involved in cognition: attention, memory and reward. This led to elaboration of a set of detailed component functions of cognition, and thence to principles for cognitive processing in the brain and an associated cognitive architecture. In section 4 we analyzed two basic cognitive tasks, the WCST and the Tower of Hanoi. There is much more to be done, especially by the implementation of language, for developing such high-level cognitive processing, beyond the simple cognitive processing discussed here.

Acknowledgement

The author would like to acknowledge the support of the European Union through the FP6 IST GNOSYS project (FP6-003835) of the Cognitive Systems Initiative.

References

[1] Taylor JG (2000) Attentional Movement: the control basis for Consciousness. Soc Neurosci Abstr 26:2231 #839.3

[2] Fragopanagos N, Kockelkoren S & Taylor JG (2005) Modelling the Attentional Blink Cogn Brain Research (in press)

[3] Taylor JG (2003) Paying Attention to Consciousness. Progress in Neurobiology 71:305-335

[4] Sergent C & Dehaene S (2005) private communication

[5] Barto A (1995) Adaptive Critics and the Basal Ganglia. In: Models of Information Processing in the Basal Ganglia, JC Houk, J Davis & DC Beiser (Eds). Cambridge MA: MIT Press. pp 215-232

[6] Taylor JG (2005) From Matter to Consciousness: Towards a Final Solution? Physics of Life Reviews 2:1-44

[7] Taylor JG (2004) A Review of brain-based neuro-cognitive models. Cognitive processing 5(4):19-217

[8] Houde O & Tzourio-Mazayer N (2003) Neural foundations of logical and mathematical cognition. Nat Rev Neuroscience 4:507-514

[9] Monchi O, Taylor JG & Dagher A (2000) A neural model of working memory processes in normal subjects, Parkinson's disease and schizophrenia for fMRI design and predictions. Neural Networks 13(8-9):963-973

[10] Monchi O, Petrides M, Doyon J, Postuma RB, Worsley K & Dagher A (2004) Neural Bases of Set Shifting Deficits in Parkinson's Disease. Journal of Neuroscience 24:702-710

[11] Taylor JG & Fragopanagos N (2005) The interaction of attention and emotion. Neural Networks 18(4) (in press)

[12] Taylor N & Taylor JG (2000) Analysis of Recurrent Cortico-Basal-Ganglia-Thalamic Loops for Working Memory. Biological Cybernetics 82:415-432

[13] Kasderidis S & Taylor JG (2004) Attentional Agents and Robot Control. International Journal of Knowledge-based & Intelligent Systems 8:69-89

[14] Kasderidis S & Taylor JG (2005) Combining Attention and Value Maps. Proc ICANN05 to appear)

Neuronal Behavior with Sub-threshold Oscillations and Spiking/Bursting Activity Using a Piecewise Linear Two-Dimensional Map

Carlos Aguirre, Doris Campos, Pedro Pascual, and Eduardo Serrano

GNB, Escuela Politécnica Superior, Universidad Autonoma de Madrid,
28049 Madrid, Spain
{Carlos.Aguirre, Doris.Campos, Pedro.Pascual, Eduardo.Serrano}@uam.es

Abstract. A phenomenological neuronal model based on a coupled piecewise linear two–dimensional map is presented. The model mimics many of the neuronal features such as spiking, bursting and subthreshold activity. The model requires a computational effort lower than most of the phenomenological or differential neuronal models and its behavior is coherent with the one present in the other models. The regimes of synchronization of a pair of coupled maps is also explored.

1 Introduction

In the last few years phenomenological neuron models that allow the replication of spiking/bursting neural behavior have been proposed [1][2][3]. These models present a low computational effort in difference with well known differential models such as the Hodgkin-Huxley model [4], or the Hindmarsh-Rose model [5] that require a high number of floating point operations to simulate a single neuron in the network for 1 ms and, therefore, are not suitable for the simulation of large scale neural networks even for short periods of time.

These new phenomenological models allow the study of biological neural networks composed of a large number of neurons. Most of the these phenomenological models are based on low–dimensional iterative maps [1][3] or are numerical approximations to differential maps [2] and most of them lack important biological features such as subthreshold oscillations. Subthreshold activity is known to play an important role in some specific of rhythmic activity vulnerable to noise [6]. The lack of subthreshold activity in phenomenological models does not allow the realistic simulation of several kinds of neurons that present such behavior and recent modifications have been proposed to some of these models in order solve that lack [7].

In this work we extend the functionality of the model presented at [3] in order to obtain stable subthreshold neuronal activity without losing previous neuronal features of the model such as spiking/bursting behavior, spontaneous activity or response to external neuronal input. The model here presented maintains a computational effort of two floating points operations in each iteration for the simulation of neuronal activity and its behavior is coherent with the one found

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 103–108, 2005.

in other phenomenological or differential models. We also study the behavior of coupled neurons. We present a very simplified synaptic model that again maintains a low computational effort presenting biological features such as several regimes of synchronization between neurons for different coupling strengths.

2 A Neuron Model with Sub-threshold Oscillations

Models of neuronal behavior based on bifurcation theory [1][9] allow the mimic of several types of behavior that occur in neural activity. These models are mainly based on two–dimensional maps composed of both a slow and a fast variable. In these models the transition from silence to spiking/bursting activity is provided by a noninvertible transition as a control parameter (usually an external input) crosses certain threshold. Even when this situation is typical for some biological neurons, some other neurons present a soft transition from the regime of silence to the spiking/bursting regime. These neurons must present oscillations bellow the spike threshold.

In [3] a neuronal model based on a coupled two–dimensional map with one fast and one slow variable is presented. This model mimics spiking/bursting activity and presents a behavior that is consistent with other phenomenological and differential models that require a higher computational effort. The model can be modified in order to obtain subthreshold activity, the new map can be written in the form:

$$y_{n+1} = \begin{cases} \frac{H(s_n)}{B} * y_n & \text{if } 0 \leq y_n < B \\ (y_n - B) * \frac{K(s_n)-H(s_n)}{C-B} + H(s_n) & \text{if } B \leq y_n < C \\ (y_n - C) * \frac{T(s_n)-K(s_n)}{D-C} + K(s_n) & \text{Otherwise} \end{cases} \tag{1}$$

$$s_{n+1} = \begin{cases} 0 \quad \text{if } s_n = 1 \quad and \quad (y_n > D \ or \ (y_n > C - S \quad and \quad y_n < C)) \\ 1 \quad \text{if } s_n = 0 \quad and \quad (y_n < L \ or \ (y_n >= C \quad and \quad y_n < C + E)) \\ s_n \ \text{Otherwise} \end{cases} \tag{2}$$

where $H(s) = H_0+s*(H_1+\sigma)$, $K(s) = K_0+s*(K_1+\sigma)$ and $T(s) = T_0+s*(T_1+\sigma)$ and B, C, D, S, E, L, H_1, H_0, K_1, K_0, T_1, T_0 are non–negative parameters verifying the following conditions: $L < B < C < D$, $H_0 \leq B$, $H_1 + H_0 \geq B$, $K_0 \leq C$, $K_1 + K_0 \geq C$, $T_0 \leq D$ and $T_1 + T_0 \geq D$, σ represents an external total input to the neuron. The parameter C is the spike threshold. The value of the slow variable s is modified as the fast variable y, that represents the membrane potential, crosses the threshold values L, E, S and D. The variable s represents the re–polarization ($s = 0$) or de–polarization ($s = 1$) status of the neuron. The change in the value of the slow variable s, and therefore in the polarization status of the neuron, is governed by the crossing of the fast variable y through the threshold values L, E, S and D. A plot of the map is depicted in Figure 1.

In difference with the model presented in [3] this modified model allows the presence of subthreshold activity. In Figure 2 the behavior of a bursting wave

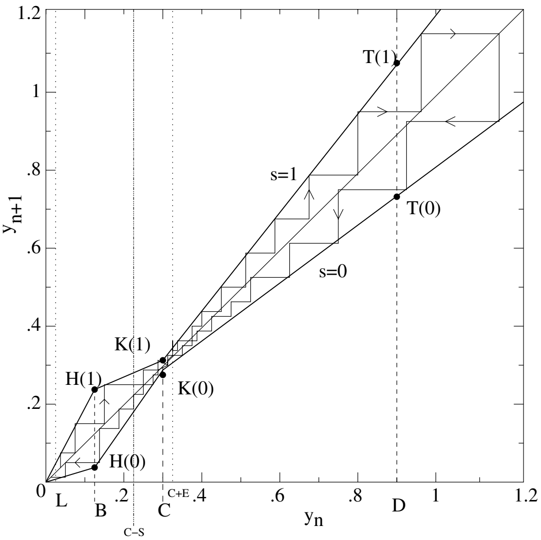


Fig. 1. Graph of the piecewise linear function y

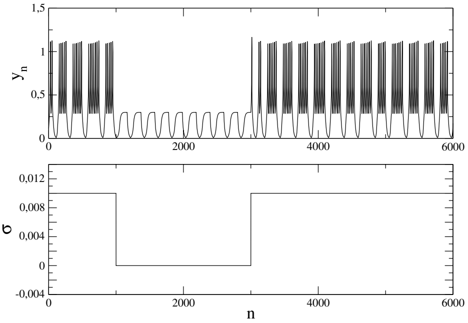


Fig. 2. Behavior under different injection of external input. Subthreshold oscillations are clearly observed when $\sigma = 0$. Parameters are described in the text.

with sub-threshold oscillations under a non-constant injection of external input is depicted. Observe that in the regime of no external input, the model presents oscillations below the spike threshold. Once the external input is reestablished, the model continues its bursting activity. The parameters for the map are: $L = .01$, $B = .15$, $C = .3$, $D = .9$, $H_0 = .14$, $K_0 = .28$, $T_0 = .75$, $H_1 = .02$, $K_1 = .0199$, $T_1 = .3$, $E = 0.0085$ and $S = 0.0001$.

Furthermore the model can present a soft transition from the silent mode to the spiking/bursting mode governed by the external input parameter σ in a similar way than other phenomenological models do [7]. Observe in Figure 3 the transition from the silent mode to an spiking mode as the external input σ is increased.

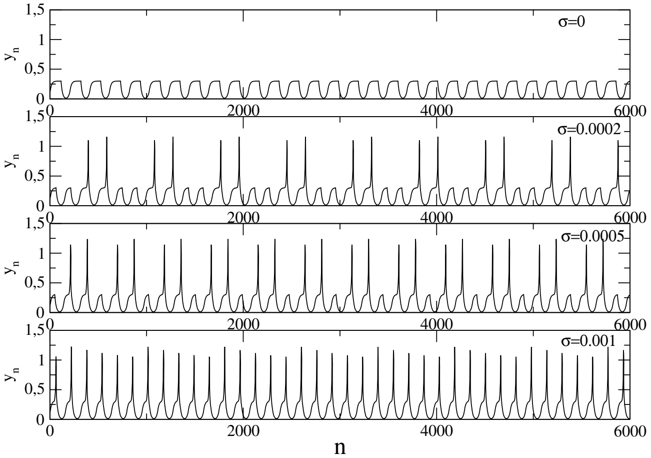


Fig. 3. Soft transition from silent to spiking regime. Parameters are described in the text.

3 Regimes of Synchronization in Coupled Maps

The previous map can be generalized in order to receive synaptic inputs from other neurons in the network. In the generalized mode, a chemical synaptic transmission can be modeled by substituting a constant input σ to neuron i by the total input at time n:

$$\sigma_{n,i} = \sigma_i^e + \frac{1}{\Gamma_i} \sum_{j=1}^{N} g_{ij} s_{n-1,j} H(y_{n-1,j} - \Theta) \qquad (3)$$

where σ_i^e represents both the external input to neuron i and the action of any current not explicitly captured by the model. Γ_i is the number of neighbors of neuron i, N is the number of neurons in the network, $y_{n,j}$ and $s_{n,i}$ represent respectively the value of y and s for the neuron j at time n. The parameter g_{ij} is the synaptic coupling coefficient between neuron i and neuron j and $H(x)$ is the usual Heaviside function. The threshold Θ has been chosen such that every spike in the single neuron can reach the threshold ($\Theta = C$).

Synchronized neuronal firing has been suggested as particularly relevant for neuronal transmission and coding. The presence of synchronization has been demonstrated in special areas such as the olfactory system [11] or the hipocampal region [12]. Real neurophysiology experiments [8,10] show that ensembles of coupled neurons can present different regimes of synchronization. These regimes are reproduced both by differential or iterative models [9]. The synchronization phenomena in map (1) can be observed in figure 4 for identical interacting bursting neurons with a symmetric coupling value of $g_{1,2} = g_{2,1} = .05$. As can be observed in the lower panel of figure 4 the synchronization of the individual spikes is achieved after a initial transient period. Note that with a low value

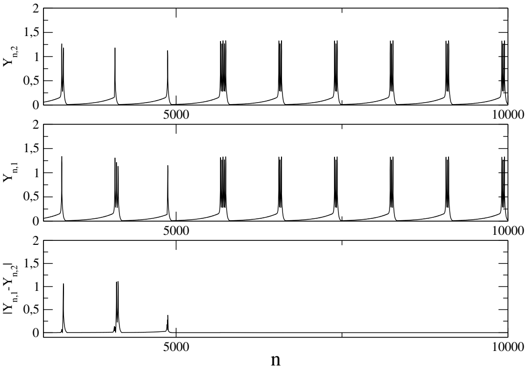


Fig. 4. Synchronization in coupled maps. Parameters are described in the text, $g_{1,2} = g_{2,1} = .05$ and $\sigma_1^e = \sigma_2^e = .001$.

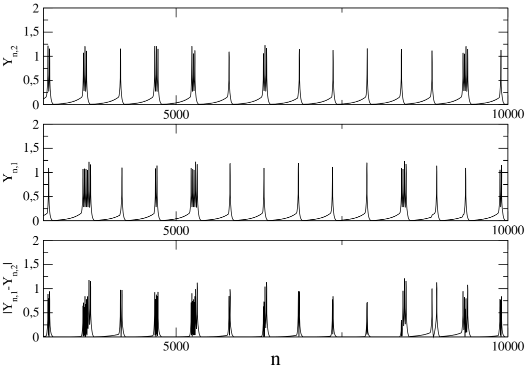


Fig. 5. Synchronization in coupled maps. Parameters are described in the text. $g_{1,2} = g_{2,1} = .005$ and $\sigma_1^e = \sigma_2^e = .001$.

of the coupling coefficient $g_{1,2} = g_{2,1} = .005$ the synchronization effect is not clearly obtained as can be seen on figure 5.

4 Results and Conclusions

The following results and conclusions can be established.

- A piecewise linear two–dimensional map that, in difference with most phenomenological models, can present sub-threshold oscillations and spiking/bursting behavior with a low computational cost is presented.
- The map can present a soft transition from silent mode to a spiking mode similar to real neurons or differential models.
- The map can be generalized in order to accept input from other neurons in the network.

– Coupled maps present a regime of synchronization when enough coupling strength is considered. Low coupling strength yields no (or slow) synchronization.

We thank the Ministerio de Ciencia y Tecnología (BFI-2003-07276). (PP) and (CA) are partially supported by BFM2002-02359. (ES) is partially supported by TIN2004-04363-C03-03.

References

1. Rulkov, N. F., Modeling of spiking-bursting neural behavior using two–dimensional map, Physical Review E **65** 041922 (2002).
2. Izhikevich, E. M. Simple Model of Spiking Neurons IEEE Trans. on Neural Networks **68** 052901 (2003).
3. Aguirre, C., Campos, D., Pascual P. and Serrano E., A model of spiking-bursting neuronal behavior using a piecewise linear two-dimensional map Lecture Notes in Computer Science **3512** (2005) 130–135.
4. Hodgkin, A. L. and Huxley, A. F., A quantitative description of membrane current and application to conduction and excitation in nerve, Journal of Physiology **117** (1954) 165–181.
5. Rose, R. M. and Hindmarsh, J. L., The assembly of ionic currents in a thalamic neuron, I The three dimensional model. Proceedings of The Royal Society of London B, **237** (1989) 267–288.
6. Makarov, V.A., Nekorkin, V.I. and Velarde, M.G., Spiking behavior in a noise-driven system combining oscillatory and excitatory properties, Physical Review Letters **86** 3431 (2001)
7. Shilnikov, A.L. and Rulkov, N. F., Subthreshold oscillations in a map-based neuron model, Physics Letters A **328** (2004) 177–184.
8. Elson, R. C. et al, Synchronous Behavior of Two Coupled biological Neurons. Physical Review Letters , 81 (25), 5692 (1998)
9. Izhikevich, E. M. Neural Excitability, Spiking and Bursting, International Journal of Bifurcation and Chaos **10** (2000) 1171–1266.
10. Rulkov, N. F., Timofeev I, and Bazhenov M, Oscillations in large-scale cortical networks: map-based model Modeling of spiking-bursting neural behavior using two–dimensional map, J. Computational Neuroscience **17** (2004) 203-223.
11. M. Bazhenov, M. Stopfer, M. Rabinovich, R, Huerta, H.D.I. Abarbanel,T.J. Sejnowski and G. Laurent, Model of transient oscillatory synchronization in the locust antennal lobe, Neuron **30** (2001) 553–567.
12. C.M. Gray and W. Singer, Stimulus-specific neuronal oscillations in orientation columns of cat visual cortex Proceedings of the National Academy of Sciences. U.S.A. **86** (1989) 1698–1702

On-Line Real-Time Oriented Application for Neuronal Spike Sorting with Unsupervised Learning

Yoshiyuki Asai[1,2,3], Tetyana I. Aksenova[4,5], and Alessandro E.P. Villa[1,2,4,6]

[1] Institute for Scientific Interchange Foundation, Torino, Italy
[2] Neuroheuristic research group, INFORGE-CP1, University of Lausanne, Switzerland
{oyasai, avilla}@neuroheuristic.org
[3] National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan
yoshiyuki.asai@aist.go.jp
[4] Laboratory of Preclinical Neuroscience, INSERM U318, Grenoble, France
tatyana.aksyonova@ujf-grenoble.fr
[5] Institute of Applied System Analysis, Ukrainian Academy of Sciences, Ukraine
[6] Laboratoire de Neurobiophysique, Université Joseph Fourier Grenoble, France
alessandro.villa@ujf-grenoble.fr

Abstract. Multisite electrophysiological recordings have become a standard tool for exploring brain functions. These techniques point out the necessity of fast and reliable unsupervised spike sorting. We present an algorithm that performs on-line real-time spike sorting for data streaming from a data acquisition board or in off-line mode from a WAV formatted file. Spike shapes are represented in a phase space according to the first and second derivatives of the signal trace. The output of the application is spike data format file in which the timing of spike occurrences are recorded by their inter-spike-intervals. It allows its application to the study of neuronal activity patterns in clinically recorded data.

1 Introduction

Extracellularly recorded action potentials–the spikes–may appear under a broad range of waveforms which depend on the type of the neuron that is generating the discharge and on a number of experimental variables such as the microelectrode electrical properties, microelectrode geometry, neuropil insulation properties and, last but not least, the distance and location of the microelectrode tip with respect to the neuronal cell body. It is generally assumed that a neuron that is generating spikes given a similar membrane state during stationary recording conditions will produce signals that will appear nearly identical in shape. Several methods of spike sorting were developed during the past decades. The main steps of such methods can be summarized as follows: (1) spike detection, which is a task to cut out a certain time window of raw signal which are assumed to correspond to the spikes; (2) extraction of the characteristics of the

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 109–114, 2005.

detected waveforms; (3) clusterization of the waveforms into several groups, such that each group should ideally correspond to the spikes generated by one neuron. Step (2) has been extensively studied using principal component analysis [7], independent component analysis [6], wavelet transform [4] and probabilistic model [5]. The classification at step (3) directly corresponds to the spike sorting in most of those algorithm, implying that these algorithms are designed for off-line data processing. Functional human neurosurgery such as embedding a chronic electrode for deep brain stimulation [2] requires quick characterisation of the neuronal activity in order to select the optimal target. This need has put high pressure for the development of on-line real-time spike sorting methods. One such algorithm requires to add two more steps to the previous methods, *i.e.* step (4) finding representative signatures in each cluster, and step (5) sorting spikes according to the signatures. Then, spikes clustering and spikes sorting should be separate tasks.

In this report we present a new application for on-line unsupervised spike sorting based on the method developed by [1]. We use the representative signatures as templates for the template-based spike sorting which can be operated in either on-line or off-line mode.

2 Methods

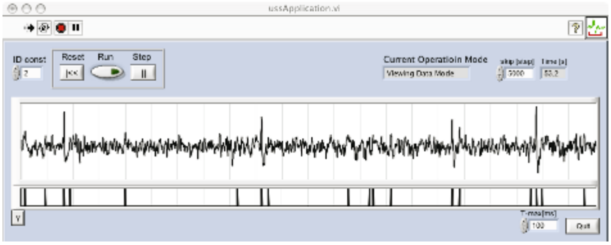
2.1 Application Architecture

The proposed application is composed of two parts, i.e., the computation engine and the user interface. The computation engine was written in ANSI C language, which was clearly independent from the user interface. The graphical user interface (GUI) was built on Labview 7.1 (National Instrument). Since Labview is available on several OS platforms, the application developed on Labview can be multi-platform compatible. We also developed command-line user interface (CUI), which can work as a batch process.

2.2 Unsupervised Leaning Algorithm

The first and second derivatives were used to detect events instead of the raw signal. The derivatives were calculated by a convolution of the raw signal and a kernel function [1]. Since this numerical method has a bandpass filtering effect, the noise in the derivatives was reduced and long-term trend of the raw signal was also removed. The event detection threshold was defined as $m \pm k\sigma$ where m and σ represent the mean and deviation of the first derivatives, respectively, and k is a coefficient set by users.

A trace which represents a typical shape of a extracellularly recorded neuronal spike is referred to as "template" in this manuscript. Spikes generated by the same neuron are supposed to be similar in shape to the template of the neuron. Hence it is assumed that all spikes generated by the same neuron will form a cluster around the template in a phase space spanned by time, the first and second derivatives of the raw signal [1], in which the measure of the dissimilarity



ID const 2

Current Operatioin Mode Viewing Data Mode

skip [ssap] 3000

time [s] 53.2

T-max[ms] 100

Fig. 1. Main window of the application with graphical user inter face. There are several buttons to control the application at the top of the window. The waveform shows the raw signal and a small chart just below it indicates occurrences of detected events.

between spikes are defined. The "learning step" is a procedure to form clusters of detected events and to find an event nearest to the center of mass of each cluster, which is referred to as a template. This procedure is done automatically without any intervention by users, hence this is termed as "unsupervised learning".

Sorting spikes is the process aimed to associate a newly detected event to one of the "learned" templates. The dissimilarity between the new event to all templates were evaluated. The event was sorted into the cluster that gave the minimum dissimilarity to the event, if the minimum dissimilarity was smaller than a threshold associated to each template. Otherwise the event was not sorted. Notice that one event can be sorted only into one cluster.

2.3 Performance Test

The performance of the spike sorting can be evaluated by the number of sorting errors, which include three types of errors, *i.e.*, (I) to miss events that should be sorted; (II) to sort events which should not be sorted; (III) to sort events which should be sorted in another template. The performance of our algorithm was tested with three data sets. The first test set, noiseless data, included three types of template spikes distributed randomly in time with the firing rate at 7.5 spikes/s for each template. The total duration of the data was 3 min. From this set we generated two more test sets by adding two different levels of noise, *i.e.* high noise (SNR=2.51 dB) and low noise (SNR=3.55 dB).

3 Results

3.1 Application with Graphical User Interface

Users can select one of two data streams, *i.e.* WAV formatted files in the off-line operation mode and through a A/D data acquisition board (National Instruments, NI PCI-6250) for the on-line operation mode. Users can shift from one mode to the other at anytime. In each operation mode, the application provides three utilities: signal viewer only with event detection, templates learning with

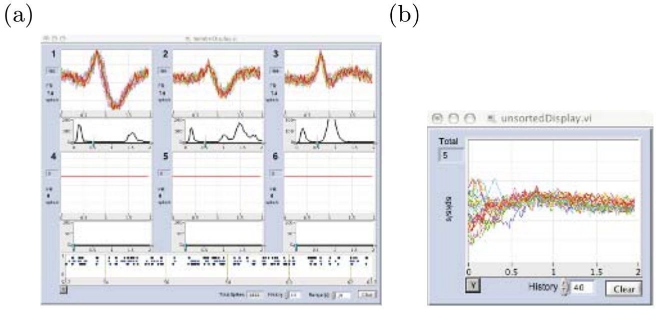


Fig. 2. a. Status view of spike sorting. There are six charts numbered 1 to 6, that display the sorted events, grouped in six different clusters. Notice that in this figure only the first three templates were detected. Below of each chart, histogram of distances are displayed. At the bottom of the window, a raster plot (time series of the event occurrences) is shown. b. A window shows all events which were not classified as members of any clusters.

error distributions, and spikes sorting, which is available after having run the templates learning at least once. The main window of the application (Fig. 1) shows the waveform of the raw signal and the occurrences of detected events. Spike sorting is performed according to the learned templates. The application provides a window that displays all spikes which were sorted into clusters with information about the amount of sorted events, uniqueness index of the cluster, the firing rate (Fig. 2a). On this window users can manipulate and tune the value of the template-specific threshold for each template. Besides, there is a window to show spikes which could not be sorted into any cluster (Fig. 2b). The application with command line interface (CUI) provides the possibility to work in batch mode, which is particularly useful for off-line processing of large amount of data saved into WAV formatted files.

3.2 Performance Test

The performance of the unsupervised spike sorting (USS) was examined using artificially generated test data. Table 1 shows the dependency of USS performance on the threshold for event detection. For the test data with low level noise, the number of Type I and II errors were kept small. For the test data with high level noise, sorting with larger threshold showed relatively high percentage of Type I errors, but Type II error was little.

3.3 Example of Clinical Data

The unsupervised spike sorting was applied to the electrophysiological data recorded from patients with Parkinson's disease during the surgical operation aimed to implant a micro-electrode for chronic deep brain stimulation of STN in

Table 1. Dependency of USS performance on the threshold for the event detection. Error types are shown in percentage with respect to the noiseless data test. T1, T2 and T3 represents templates 1, 2, 3 observed in the noiseless data test. σ represents the SD used for event detection in the test data set. Notice that Type III error mentioned in the text never occurred in this test.

		High Noise			**Low Noise**		
Threshold		1.9σ	2.0σ	2.3σ	1.9σ	2.0σ	2.3σ
Type I error	T1	1.4	3.7	5.2	1.0	1.0	1.1
	T2	0.4	0.4	0.4	0.5	0.1	0.3
	T3	3.1	3.1	4.2	0.1	0.1	0.1
Type II error	T1	0.0	0.0	0.0	0.0	0.0	0.0
	T2	0.0	0.0	0.0	0.0	0.0	0.0
	T3	0.1	0.1	0.1	0.2	0.2	0.2

the University Hospital of Grenoble [2]. The event detection threshold was fixed at 2σ. Figure 3 shows obtained templates.

4 Discussion

Template based spike sorting techniques are appropriate for real-time application but they usually request the users to choose the templates. This implies that the results of spike sorting depend on the knowledge and experience on signal analysis by the users. The spike sorting method presented here, USS, can find templates without user's supervision. The unsupervised learning algorithm provides an opportunity for unexperienced users to save time to find templates and obtain high quality results of spike sorting. Template learning is the most computationally intensive task, because the algorithm calculates the all-to-all dissimilarity between all detected events. Conversely, spike sorting consists to associate a newly detected event to the closest template and requires little calculation power. This is suited for fast signal processing tasks, *e.g.* on-line real-time signal processing.

Event detection and its subsequent classification are crucial for a spike sorter to work properly. Even if additional types of events, which are not neural spikes, are wrongly detected this is not relevant if these events are eliminated by the classification procedure. More events require more computational resources for their processing, but this becomes not critical because modern computers are fast enough for this task. However. if events corresponding to neural spikes are missed, this immediately pulls down the performance of the spike sorter, because it increases the type I error. This is indeed the main source of potential trouble due to unsupervised learning and subsequent classification procedures. The USS presented here has proven to be very robust to this respect in all tests performed so far. We are looking for additional tests with several types of noise but we believe that USS sets the premisses for its application in clinical care conditions.

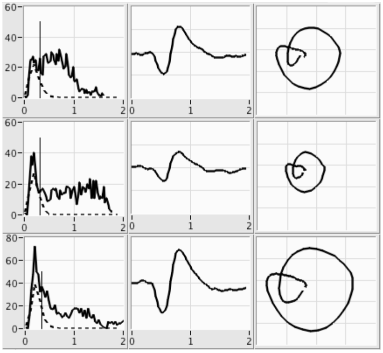


Fig. 3. Three clusters identified from the electrophysiological recording in the human subthalamic nucleus. The raw signal trace is shown in Fig. 1. In the left column, each panel shows the statistical distribution of the distances from the template to all other events (the solid curve), and its fit by a Gamma probability density function (dotted curve) used to calculate the template-specific threshold indicated by a vertical tick. The raw signal profiles of the representative neural spikes of each cluster are shown in the center column. In the right column are displayed the orbits in the phase space of the traces shown in the central column.

Acknowledgments. The authors wish to thank Dr. Jan Eriksson and Javier Iglesias for technical suggestions. This research was partly supported by the Canon Foundation in Europe Research Fellowship, and JSPS Research Fellowships for Young Scientists No. 0407735.

References

1. Aksenova TI, Chibirova O, Dryga AO, Tetko IV, Benabid Al, Villa AEP. An unsupervised automatic method for sorting neuronal spike waveforms in awake and freely moving animal. Methods; 2003; 30; 178-187
2. Benabid AL, Pollak P, Gross C, Hoffmann D, Benazzouz A, Gao DM, Laurent A, Gentil M, Perret J. Acute and long-term effects of subthalamic nucleus stimulation in Parkinson's disease. Stereotact Funct Neurosurg; 1994; 62; 76–84.
3. Chibirova OK, Aksenova TI, Benabid A-L, Chabardes S, Larouche S, Rouat J, Villa AEP. Unsupervised Spike Sorting of extracellular electrophysiological recording in subthalamic nucleus of Parkinsonian patients. Bio Systems; 2005; 79; 159-171.
4. Letelier JC, Weber PP. Spike sorting based on discrete wavelet transform coefficients. Journal of Neuroscience Methods 2000; 101; 93–106.
5. Lewicki MS. Bayesian modeling and classification of neural signals. Neural Comp; 1994; 6; 1005-1030
6. Takahashi S, Anzai Y, Sakurai Y. A new approach to spike sorting for multi-neuronal activities recorded with a tetrode – how ICA can be practical. Neuroscience Research; 2003; 46; 265–272
7. Zhang PM, Wu JY, Zhoub Y, Liang PJ, Yuan JQ. Spike sorting; Template-matching; χ^2-Test; Overlapping; Principal component analysis; Subtractive clustering. Journal of Neuroscience Methods; 2004; 135; 55–65.

A Spiking Neural Sparse Distributed Memory Implementation for Learning and Predicting Temporal Sequences

J. Bose, S.B. Furber, and J.L. Shapiro

School of Computer Science,
University of Manchester,Manchester, UK, M13 9PL
joy.bose@cs.manchester.ac.uk,
{steve.furber, jonathan.shapiro}@manchester.ac.uk

Abstract. In this paper we present a neural sequence machine that can learn temporal sequences of discrete symbols, and perform better than machines that use Elman's context layer, time delay nets or shift register-like context memories. This machine can perform sequence detection, prediction and learning of new sequences. The network model is an associative memory with a separate store for the sequence context of a pattern. Learning is one-shot. The model is capable of both off-line and on-line learning. The machine is based upon a sparse distributed memory which is used to store associations between the current context and the input symbol. Numerical tests have been done on the machine to verify its properties. We have also shown that it is possible to implement the memory using spiking neurons.

1 Introduction

Time is important in many real world tasks. Many applications are sequential in nature, where the time order of events is important. A sequence machine is a system capable of storing and retrieving temporal sequences of patterns that represent discrete symbols. Neural models of temporal sequence learning have been a source of interest [7,1,2]. We are interested in a model that does one-shot learning, can learn on-line, has good memory capacity, can work with a variety of sequences and can be implemented using spiking neurons. We do not feel a model with all these features has been developed earlier.

2 Sequence Learning

One thing we need to consider is that two different sequences may have certain symbols in common. If we use an associative memory to learn the two sequences, where the association is between the current symbol and the next symbol, it cannot decide the next symbol after the common because the two sequences have different successors of the common symbol. It needs to have some idea of the context as well. Thus the basic sequence machine needs to have four

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 115–120, 2005.

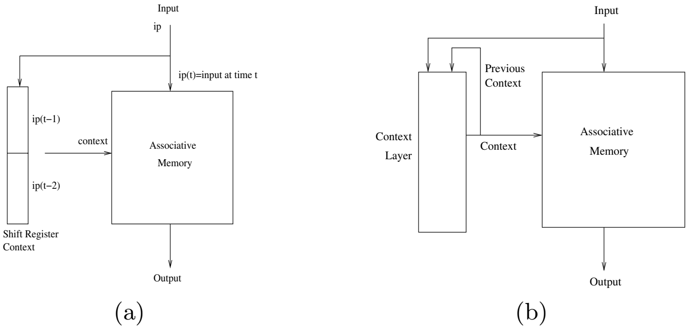


Fig. 1. (a)Shift register model and (b)a separate neural layer as context

components: input, output, main memory and context memory. The ideal online sequence machine is one that can look back from the most recent inputs, as far as is necessary to find a unique context for deciding the next character to be predicted. The machine should be able to 'lock-on' or converge to a context (and thus predict the next output) if it has seen it earlier, and to learn the new association if it has not. It should have infinite look-back, yet should be able to distinguish between different contexts.

When a new symbol is presented at the input of the on-line sequence machine, first of all it learns to associate the new input symbol with the present value of the context. Based on the new input and the present context, it creates a new context. Finally it predicts the next output by presenting the modified context to the memory. The above steps incorporate both prediction and learning. If the memory has seen a similar input and context before, the expected next output will be predicted. On the other hand, if it is given a new association, it writes it to the memory. In this case the predicted output will be erroneous, but the new association written into the memory should improve its performance if a similar context is encountered subsequently.

2.1 The Shift Register Model

One way to represent the context could be to have a fixed length time window of the past, and associate the next output with inputs in the time window, as is done in Time Delay Neural Nets (TDNN)[6]. Such a memory acts like a shift register. However, the time window is of fixed size, and the number of common symbols might be greater than this. Fig. 1(a) shows a shift register with look-back of 2.

2.2 The Context Neural Layer Model

Another approach is to use a separate 'context' neural layer, with fixed weights, to represent the entire history of the sequence. Fig. 1(b) gives the structure of the context layer based model. The influence of the old context can be modulated by multiplying the old context outputs by a constant λ and feeding them back as inputs. The context encodes the entire past history or 'state' of the sequence. Such a model resembles a finite state machine. A similar model with feedback was used by Elman [4]. A problem with the context neural layer model is that to retrieve a sequence we need to start retrieval from the beginning, else the context will be different.

2.3 Combined Model

The shift register model and the separate context layer model both have their advantages and disadvantages. We combine the two in a new memory model by using a separate context layer with modulated context, where the new context is determined by both the input and a shifted version of the present context. The new context is formed from the input and old context as follows (see fig. 2(b)): First we scramble of the old context, which is equivalent to passing it through a neural layer. Then the old context is mapped on to a high dimension, expanded and added to the expanded input. The sum is then contracted. The intention is that the result should be strongly dominated by the present input, but should have some bits of the past context in it as well.

3 An Implementation Using Sparse Distributed Memory

We implemented the sequence machine using a modified Kanerva Sparse Distributed Memory (SDM)[3] using N-of-M codes[5]. We used an N-of-M code to

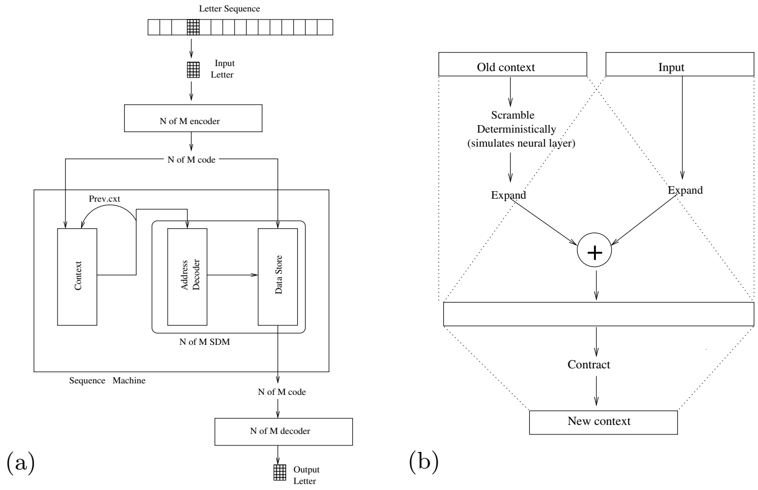
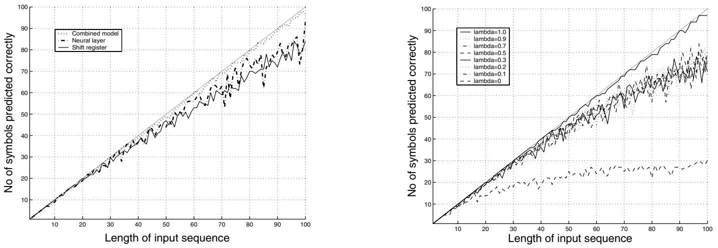


Fig. 2. shows a sequence machine using an N-of-M Kanerva SDM, having address decoder, data memory and context neural layers. shows how the new context is created from the old context and the input in the combined model. The model has aspects of both the neural layer and the shift register.



(a) (b)

Fig. 3. Compares the performance of three types of sequence memories: Shift register, neural layer and the combined model. The straight line represents the ideal case, when the complete sequence is recovered. The combined model(dotted line) performs better than the others. shows the memory capacity of the combined model in which the context sensitivity λ has been varied to see what effect λ has on memory performance. The topmost curve is for λ=1.0 and lowest for λ=0.

encode the symbols, in which N of a total M components are active simultaneously to give a valid code. Such memories have been shown to have good information efficiency, scalability and error tolerance[5].

The N-of-M SDM has two layers of neurons: an address decoder layer, whose primary purpose is to cast the input symbols into high dimensional space to make it linearly separable, and a second correlation-matrix layer called the data store, which associates the first symbol as decoded by the first neural layer, with the second symbol. Learning takes place only in this layer, while the weights of the first address decoder layer are set to a constant random value. The complete system with context is shown in fig. 2(a).

4 Numerical Tests on the Sequence Machine

We conducted some tests on the sequence machines described above, to compare their behaviour with different kinds of sequences. There are three kinds of sequence machine we are comparing, namely the shift register, context neural layer and combined model. In each case we used the same SDM of size 512 by 256, with 1024 address decoder neurons. Each input symbol was encoded as an ordered 11-of-256 code vector. The learning algorithm used in the data store of the SDM is that the new weight matrix is formed by taking the maximum of the old weight matrix and the outer product of the two vectors that are being associated.

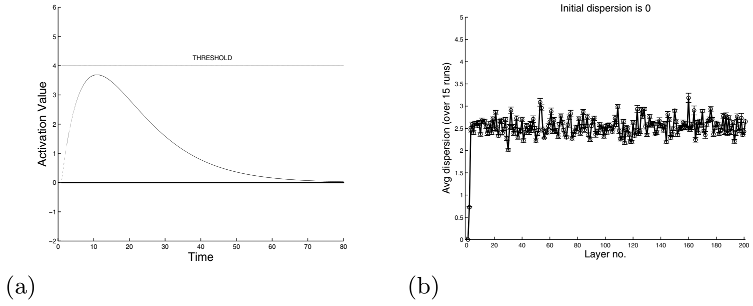
Comparison of different models: Here we compare the three models of sequence machine with optimised parameters and analyse their memory capacity for different sequence lengths. Repeated characters are guaranteed for sequence length greater than 15, which is the alphabet size. Figure 3(a) shows the results. For each point on the figure we started with a blank memory and input the sequence twice. The memory learns the sequence on the first presentation of the input, and in the second time we check the predicted output sequence to see how accurate the prediction is. We see that the combined model performs the best of the three and obtains near perfect recall.

Effect of the context sensitivity factor: Here we vary the context sensitivity factor λ and see the memory performance of the combined model. We see three clear zones. When lambda is 0, the machine is not at all sensitive to context and it performs badly. When it is 1, which means that the context is given equal priority as the current input, it performs quite well, giving near perfect matching. When it is between these two zones, the combined model behaves effectively like a shift register. Fig. 3(b) shows one such experiment.

5 Implementation Issues Using Spiking Neurons

In this section we mention a few issues when implementing this sequence machine using low level asynchronous spiking neurons. One of the reasons we used SDM's to create our memory was their suitability to spiking neural implementation. We define a symbol in our sequence machine as being represented by a burst of spikes emitted by a layer of neurons, the information being encoded in the choice of neurons firing and times of firing. When modelling such high level synchronous systems using low level components, we need to ensure that the stability and coherence of the symbols is maintained. By stability we mean that the spike burst should not either die out or saturate when propagated through different layers, but maintain the same average level. By coherence we mean that the different bursts should not interfere with each other, else it would destroy the information being propagated as symbols.

For our modelling we used a modification of the standard leaky integrate and fire (LIF) neural model [8], in which incoming spikes increment the driving force or first derivative of the activation, rather than the activation itself. Like in the



(a) (b)

Fig. 4. Is the plot of a typical neural activation with time, when the neuron receives a single input spike at t=0. plots the average dispersion over 16 runs of a burst of spikes over a network of 200 feed-forward layers.

standard model, if the activation exceeds a local threshold, the neuron fires a spike and the activation and the driving force are reset. Both the activation and the driving force decay with time, the rates of decay being governed by their respective time constants. Fig. 4(a) shows the shape of the activation curve following a single input spike at time 0. We see that the activation at first increases due to the increased driving force caused by the incoming spike, but after a time the decay becomes dominant. There is an inherent time lag between the input spike and the maximum activation reached by the neuron. If the system contains a feedback loop, such a time lag is necessary, or else at least one input neuron would have to fire a spike at the same time as an output neuron fires a spike, and there would be no temporal separation between the input and output bursts. The standard LIF model cannot achieve this property. This motivated our choice of neural model.

We then simulated a network of 200 feed forward layers with same average connectivity but random connection weights. We gave the first layer a random input spike burst of firings, and propagated the output burst to the successive layers. We ensured that the bursts were stable through feedback reset inhibition. We then plotted the temporal separation of the bursts in each layer. The results of one such experiment are plotted in fig. 4(b). We see that the average burst width tends to settle around a narrow time range. If we ensure that the separation between different bursts is large compared to this average time of one burst (by adding extra delays), we can prevent different spike bursts from interfering. This shows it is possible for a spike burst to maintain coherence. So by tuning the delays between the bursts it is possible to implement the sequence machine using spiking neurons.

6 Conclusions and Further Work

We have developed a model that is capable of on-line sequence learning and prediction. More work needs to be done on issues of implementation by real time spiking neurons. Work also needs to be done to develop suitable applications.

References

1. Vocal interface for a man-machine dialog. Dominique Beroule. ACL Proceedings, First European Conference, 1983.
2. Learning Speech as Acoustic Sequences with the Unsupervised Model, TOM. S. Durand and F. Alexandre. NEURAP, 8th Intl. conference on neural networks and their applications, France, 1995.
3. P. Kanerva. Sparse Distributed Memory. MIT Press, 1988
4. J. L. Elman. Finding structure in time. Cognitive Science, 1990, 14.
5. S.B. Furber, J.M. Cumpstey, W.J. Bainbridge and S. Temple. Sparse distributed memory using N-of-M codes. Neural Networks. 2004, 10
6. K.J. Lang and G. E. Hinton. The development of the time delay neural network architecture for speech recognition. Tech.Report 88152, Carnegie Mellon, 1988.
7. R. Sun and C.L. Giles (Eds.) Sequence Learning. Springer-Verlag, 2000
8. W. Maass and C.M. Bishop (eds.) Pulsed Neural Networks MIT Press, 1998

ANN-Based System for Sorting Spike Waveforms Employing Refractory Periods

Thomas Hermle[1,2], Martin Bogdan[1], Cornelius Schwarz[2], and Wolfgang Rosenstiel[1]

[1] Universität Tübingen, Wilhelm-Schickard-Institut für Informatik,
Technische Informatik, Sand 13, D-72076 Tübingen, Germany
{hermle, bogdan, rosenstiel}@informatik.uni-tuebingen.de
http://www-ti.informatik.uni-tuebingen.de

[2] Universitätsklinik Tübingen, Hertie-Institut für klinische Hirnforschung
Kognitive Neurologie, Otfried-Müller-Str. 27, D-72076 Tübingen, Germany
cornelius.schwarz@uni-tuebingen.de
http://www.hih-tuebingen.de

Abstract. We describe a modification of a growing grid neural net for the purpose of sorting neuronal spike waveforms from extracellular recordings in the central nervous system. We make use of the fact, that real neurons exhibit a refractory period after firing an action potential during which they can not create a new one. This information is utilized to control the growth process of a growing grid, which we use to classify spike waveforms. The new algorithm is an alternative to a standard self-organizing map used in our previously published spike sorting system. Using simulated data, we show that this modification can further improve the accuracy in sorting neuronal spike waveforms.

1 Introduction

Extracellular recordings in the central nervous system using micro electrodes is an important technique for investigating the interaction between single neurons or groups of neurons in the brain. Recordings are done either with a number of single electrodes or with multi electrode arrays comprising many electrodes.

Each electrode records the superimposed signals of several neurons in proximity to the electrode. Current multi electrode arrays allow recording from as many as one hundred neurons [1]. To assess the activity of each neuron, spike waveforms have to be extracted from the signal and to be classified and assigned to single units.

The importance of this step must not be underestimated. “The accuracy of the spike sorting critically affects the accuracy of all subsequent analyses”, therefore “development of the best possible spike sorting algorithms must be an important goal”[2]. With increasing number of electrodes, the level of automation becomes an important factor in addition to the accuracy and fully automated algorithms are required.

In this report, we present our work on a spike sorting system, meeting these requirements. An earlier version of our spike sorting system, has already proofed its ability to classify spike waveforms from tetrode recordings [3] with high accuracy and in a fully automated way [4]. Our system is able to handle cross-talk between electrodes, a feature, up to now, not yet found in other spike sorting systems [5]. More results and details

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 121–126, 2005.

of the algorithms are about to be published. The focus of the following chapters will be on possible improvements of the accuracy by observing refractory periods during the spike sorting process.

2 Methods

Our spike sorting system comprises several processing steps. In the first step, physical cross-talk between electrodes is removed by applying Independent Component Analysis (ICA). From the resulting signals, spikes are identified by application of an amplitude threshold. Few milliseconds before and after crossing of the threshold are used for a cut-out containing the complete spike waveform. Cut-outs are stored and clustered using a self-organizing map (SOM)[6] or, as an alternative, with a modified growing grid (GG), which is described in this paper. Subsequently, clusters within the trained artificial neural net (ANN) are identified with an algorithm using distances between SOM– or GG–neurons.

2.1 Removal of Cross-Talk in Multi Electrode Recordings

Due to capacitive coupling between electrodes, signals recorded on one electrode can cross-talk and thus be introduced as artifacts on neighboring electrodes. As such artifactual spikes can compromise subsequent analyses of spatiotemporal firing patterns, removal of these spikes is necessary, reconstructing the original signals.

We have tested several ICA-algorithms with simulated and real multielectrode data [5]. Many of these algorithms were able to successfully remove cross-talk from the raw signals as well as from the cut-out data. For offline analysis, the Infomax algorithm of Bell and Sejnowski[7] with the natural gradient feature of Amari, Cichocki and Yang [8], as implemented in the EEGLAB toolbox [9], provided the best results.

2.2 Classification of Spike Waveforms

Using a two-dimensional SOM with fixed size (10x10) showed already a high accuracy, which we have quantified earlier using simultaneous intra- and extracellular tetrode recordings [3]. We could show, that in many cases, cut-outs can be mapped onto a two-dimensional SOM without significant folding. Often however, the structure of the data might be represented better by a rectangular map instead of a square map and therefore we investigated the use of Growing Grid [10].

To further improve the accuracy of spike sorting and in order to use all information available, we modified the original GG algorithm to include timing information of the spikes. The fact, that the firing of real neurons exhibits a refractory period should be used to improve the quality of single unit isolation.

Growing Grid Employing Refractory Periods. GG is basically a growing SOM, which adapts the size of the map according to the structure of the data during an unsupervised growth process starting with a 2x2 map and inserting new rows or columns. In each step, a part of the data is used to train the map for few iterations in order to collect information about the structure of the data and to determine the best place to insert a new row or column.

Fritzke proposed to use the hit frequency or the quantization error of GG–neurons to determine the region which is not yet represented well enough. Starting from the GG–neuron with the highest hit frequency or the highest quantization error, a new row or column is inserted in direction to the neighboring GG–neuron with the biggest Euclidean distance.

Growth can be stopped, when most of the GG–neurons have similar hit frequencies or when the quantization error is below some threshold or if a fixed total number of GG–neurons is reached. Following the growth, a regular SOM training is used for fine tuning.

Our modified GG–algorithm uses statistics about spike timing to determine GG–neurons which unite spikes from several single units. Whenever two spikes following each other within a period less than the refractory period share the same winner GG–neuron, a counter for this GG–neuron is incremented. The GG–neuron with the highest number of refractory period violations is the starting point for growth.

The spikes originating from several single units should be covered by new or surrounding GG–neurons to improve single unit isolation. Therefore we choose the nearest neighbor instead of the most distant neighbor for insertion of a new row or column. To measure the distance between GG–neurons, the Mahalanobis distance is used if applicable.

For a set of N spike cut-out vectors $x_t(1 \leq t \leq N)$ and associated spike times $z_t(1 \leq t \leq N)$ the modified GG–algorithm is:

1. **Initialization**
   Set $k = 2$ and $m = 2$. Create a $k \times m$ grid $A = [a_{ij}](1 \leq i \leq k, 1 \leq j \leq m)$ of GG–neurons. Initialize the associated n-dimensional reference vectors $w_c(1 \leq c \leq k \cdot m)$ with small random numbers. Set the counter for violations of the refractory period ρ_c to zero.

2. **Training During Growth Phase**
   Use $\lambda_g \cdot k \cdot m$ training vectors and fixed neighborhood width σ and learning rate ϵ_g. For every training vector x_t determine the winner unit s with $\|w_s - x_t\| \leq \|w_c - x_t\|(\forall c \in A)$ and adapt all weight vectors w_c by $\triangle w_c = \epsilon_g \exp\left(-\frac{d^2(c,s)}{2\sigma^2}\right)(x_t - w_c)$, where $d(c_1, c_2) = |i_1 - i_2| + |j_1 - j_2|$ for neurons $c_1, c_2 \in A$ with $c_1 = a_{i_1 j_1}$ and $c_2 = a_{i_2 j_2}$.

3. **Insertion of New Rows and Columns**
   Determine the winner unit for all training vectors used, ordered by their spike times. Whenever a GG–neuron c is the winner unit for two training vectors x_{t_k} and $x_{t_{k+1}}$ whose spike times differ by less than the refractory period (e.g. 2 ms), increment ρ_c by 1.
   Select the GG–neuron f with $\rho_f \geq \rho_c(\forall c \in A)$ for inserting new GG–neurons in order to increase the density of reference vectors in the vicinity of f.
   Determine the neighboring GG–neuron h with the smallest Mahalanobis or Euclidean distance to f. If f and h are in the same row, then insert a new column between f and h, else insert a new row between f and h.
   Initialize the new reference vectors with the mean value of the previous neighbors. Adjust k and m and reset the counters τ_c and ρ_x.

If the stopping criteria (e.g. number of GG–neurons) is not yet met, continue with Step 2.

4. **Fine Tuning**
   Do a standard SOM training with $\lambda_f \cdot k \cdot m$ vectors and decreasing learning rate ϵ_f.

Standard parameters are: $\lambda_g = 30, \sigma = 0.7, \epsilon_g = \epsilon_f = 0.005$ and $\lambda_f = 100$.

After the SOM training in the fine tuning phase, the resulting map has to be analyzed in order to identify clusters. Spikes originating from one single unit are usually distributed over several neighboring GG–neurons forming a cluster within the trained map. Identification can be done manually by visual inspection of the waveforms and the Euclidean distances between GG–neurons. Several methods have been developed to ease identification of clusters like using hit frequencies [11] or the u-matrix [12] to visualize cluster borders, or fully automated algorithms like clusot [13] or our new algorithm esort, using the local contrast of distances to determine cluster borders (to be published). GG– or SOM–neurons with few hits, which do not clearly belong to one of the identified clusters, can be marked automatically as outliers for later inspection.

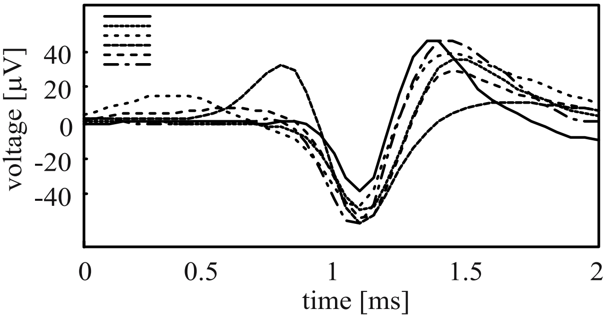


Fig. 1. Six template waveforms (*depicted with different line styles*) have been extracted from real recordings in rats' CNS. Multi-unit data has been simulated by mixing these templates together with colored noise. Spike waveforms have been inserted into the simulated recording at frequencies between 4 and 35 Hz observing refractory periods while allowing superposition of waveforms originating from different templates.

Data Generation. In order to assess the performance of the new modified GG, a synthetic data set has been generated. Real multi electrode recordings from rats' CNS have been analyzed to extract 6 template waveforms with similar amplitudes (see Fig. 1). Adding noise to these template waveforms and subsequent filtering allows to create synthetic waveforms showing similar shape and variability like the original data.

Two data sets have been constructed by using colored noise with frequencies between 10 and 3000 Hz, similar to the one observed in the real recordings and by inserting the generated waveforms at random times with the according firing frequency while assuring refractory periods.

Firing frequencies of the six simulated neurons ranged from 4 Hz to 35 Hz. For the two data sets, different levels of overlap have been used. While in the first data set (d0), full superpositions of an arbitrary number of spikes are allowed, in the second data set (d1) spikes only overlap up to 1 ms (which is half of a cut-out).

Although the simulated data looks very comparable to real data, our goal was not an exact reproduction but a synthetic data set with known spike labels and characteristics reasonably similar to real recordings.

Implementation. The spike sorting algorithms and the user interface have been implemented in MATLAB®(The MathWorks, Inc) and C, ensuring compatibility with other tools for waveform or spike train analysis on different platforms. Many data analysis algorithms are freely available from different groups as MATLAB®toolboxes, from which we have used EEGLAB's `runica()` [9] for independent component analysis.

3 Results and Conclusion

We have compared the accuracy of our modified Growing Grid algorithm employing refractory periods to the previously employed SOM using simulated data. A total number of 3092 spike waveforms has been classified, from which 820 spike waveforms contained superpositions of two or more spikes. The signal-to-noise ratios have been very low, between 1.4 and 4.3. This together with the high similarity between the spike prototypes renders this data set an extremely difficult task. However, our standard system using a 10x10 SOM has been able to achieve a good recall of 92.05% for d1 and of 80.92% for d0. Using our modified growing grid, we could improve this to 94.37% for d1, with a 10x10 grid and to 83.76% for d0 with a 12x9 grid. Furthermore, for d1, all false negatives have been marked for inspection. There have been no false positives. In addition, the new algorithm has been able to identify the correct number of single units, while the algorithm using a fixed size SOM has split up two clusters into subclusters. Even for d0, the false positives have been below 5% in both cases.

We have presented a new clustering technique for our spike sorting system, which uses refractory periods in order to improve the quality of spike sorting. Information about refractory periods has previously only been used in few works (e.g. [14]) or in order to do manual merging or splitting of clusters after application of a clustering algorithm [15]. We think that this information must not be neglected and can lead to improvements in accuracy. Further quantification of the improvement will be investigated in future studies.

Acknowledgment

Part of this work was supported by BMBF #0311858 to Cornelius Schwarz.

References

1. Buzsáki, G.: Large-scale recording of neuronal ensembles. Nat Neurosci **7** (2004) 446–51
2. Brown, E.N., Kass, R.E., Mitra, P.P.: Multiple Neural Spike Train Data Analysis: State-of-the-Art and Future Challenges. Nat Neurosci **7** (2004) 456–61

3. Harris, K.D., Henze, D.A., Csicsvari, J., Hirase, H., Buzsaki, G.: Accuracy of tetrode spike separation as determined by simultaneous intracellular and extracellular measurements. J Neurophysiol **84** (2000) 401–14
4. Hermle, T., Bogdan, M., Schwarz, C.: Spike Sorting Algorithm Based on Independent Components Analysis (INCA) and a Self-Organizing Map (SOM) Minimizing User Interaction - Application to Multineuron Recordings in CNS. In: Proceedings of the World Congress on Neuroinformatics, Wien, Austria (2001)
5. Hermle, T., Schwarz, C., Bogdan, M.: Employing ICA and SOM for Spike Sorting of Multielectrode Recordings from CNS. J Physiol Paris (2005) Accepted.
6. Kohonen, T.: Analysis of a simple self-organizing process. Biological Cybernetics **44** (1982) 135–140
7. Bell, A., Sejnowski, T.: An information-maximization approach to blind separation and blind deconvolution. Neural Comput **7** (1995) 1129–59
8. Amari, S., Cichocki, A., Yang, H.H.: A new learning algorithm for blind signal separation. In: Advances in Neural Information Processing Systems - Proceedings of the 1995 Conference. Volume 8., Cambridge, MA, MIT Press (1996) 757–763
9. Delorme, A., Makeig, S.: EEGLAB: an Open Source Toolbox for Analysis of Single-Trial EEG Dynamics Including Independent Component Analysis. J Neurosci Methods **134** (2004) 9–21
10. Fritzke, B.: Growing Grid - a Self-Organizing Network with Constant Neighborhood Range and Adaptation Strength. Neural Processing Letters **2** (1995) 9–13
11. Cottrell, M., de Bodt, E.: A Kohonen Map Representation to Avoid Misleading Interpretations. In: Proc. ESANN'96, European Symp. on Artificial Neural Networks, Bruges, Belgium, D facto conference services (1996) 103–110
12. Ultsch, A.: Self-Organizing Neural Networks for Visualization and Classification. In: Information and Classification, London, UK, Springer (1993) 307–313
13. Bogdan, M., Rosenstiel, W.: Detection of cluster in self-organizing maps for controlling a prostheses using nerve signals. In: Proceedings of 9th European Symposium on Artificial Neural Networks ESANN'2001, Brugge, Belgium, D-Facto (2001) 131–6
14. Fee, M.S., Mitra, P.P., Kleinfeld, D.: Automatic sorting of multiple unit neuronal signals in the presence of anisotropic and non-Gaussian variability. J Neurosci Methods **69** (1996) 175–88
15. Harris, K.: KlustaKwik (2004) http://sourceforge.net/projects/klustakwik/.

Emergence of Oriented Cell Assemblies Associated with Spike-Timing-Dependent Plasticity

Javier Iglesias[1,3,4], Jan Eriksson[3], Beatriz Pardo[2], and Marco Tomassini[1],
and Alessandro E.P. Villa[1,3,4]

[1] Information Systems Department, University of Lausanne, Switzerland
{Javier.Iglesias, Marco.Tomassini}@unil.ch
http://inforge.unil.ch/

[2] Centro de Biologia Molecular Severo Ochoa, Universidad Autonoma, Madrid, Spain
bpardo@cbm.uam.es

[3] Laboratory of Neuroheuristics, University of Lausanne, Switzerland
jan@lnh.unil.ch
http://www.nhrg.org/

[4] Inserm U318, Laboratory of Neurobiophysics, University Joseph Fourier,
Grenoble, France
Alessandro.Villa@ujf-grenoble.fr

Abstract. We studied the emergence of cell assemblies out of a locally connected random network of 10,000 integrate-and-fire units distributed on a 100×100 2D lattice. The network was composed of 80% excitatory and 20% inhibitory units with balanced excitatory/inhibitory synaptic weights. Excitatory–excitatory synapses were modified according to a spike-timing-dependent synaptic plasticity (STDP) rule associated with synaptic pruning. In presence of a stimulus and with independent random background noise (5 spikes/s), we observed that after $5 \cdot 10^5$ ms of simulated time, about 8% of the exc–exc connections remained active and were reinforced with respect to the initial strength. The projections that remained active after pruning tended to be oriented following a feed-forward converging–diverging pattern. This result suggests that topologies compatible with synfire chains may appear during unsupervised pruning processes.

1 Introduction

Massive synaptic pruning following over-growth is a general feature of mammalian brain maturation [1]. Pruning starts near time of birth and is completed by time of sexual maturation. Trigger signals able to induce synaptic pruning could be related to dynamic functions that depend on the timing of action potentials. Spike-timing-dependent synaptic plasticity (STDP) is a change in the synaptic strength based on the ordering of pre- and post-synaptic spikes. This mechanism has been proposed to explain the origin of long-term potentiation (LTP), i.e. a mechanism for reinforcement of synapses repeatedly activated shortly before the occurrence of a post-synaptic spike [2]. STDP has also

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 127–132, 2005.

been proposed to explain long-term depression (LTD), which corresponds to the weakening of synapses strength whenever the pre-synaptic cell is repeatedly activated shortly after the occurrence of a post-synaptic spike [3]. The relation between synaptic efficacy and synaptic pruning [4], suggests that the weak synapses may be modified and removed through competitive "learning" rules. Competitive synaptic modification rules maintain the average neuronal input to a post-synaptic neuron, but provoke selective synaptic pruning in the sense that converging synapses are competing for control of the timing of post-synaptic action potentials [5].

The originality of our study stands on the size of the network, 10,000 units, the duration of the experiment, 500,000 ms, and the application of an original bio-inspired STDP modification rule compatible with hardware implementation [6]. In this study the synaptic modification rule was applied only to the exc–exc connections. This plasticity rule might produce the strengthening of the connections among neurons that belong to cell assemblies characterized by recurrent patterns of firing. Conversely, those connections that are not recurrently activated might decrease in efficiency and eventually be eliminated. The main goal of our study is to determine whether or not, and under which conditions, such cell assemblies may emerge from a large neural network receiving background noise and content-related input organized in both temporal and spatial dimensions.

2 Model

The complete neural network model is described in details in [7] and we present here only a sketch description of the model. 10,000 integrate-and-fire units (80% excitatory and 20% inhibitory) were laid down on a 100×100 2D lattice according to a space-filling quasi-random Sobol distribution. Sparse connections between the two populations of units were randomly generated according to a two-dimensional Gaussian density function such that excitatory projections were dense in a local neighborhood, but low probability long-range excitatory projections were allowed. Edge effects induced by the borders were limited by folding the network as a torus. The state of the unit (spiking/not spiking) was a function of the membrane potential and a threshold. The states of all units were updated synchronously and the simulation was performed at discrete time steps corresponding to 1 ms. After spiking, the membrane potential was reset, and the unit entered a refractory period lasting 2 time steps. For the simulation runs presented here each unit received a background activity following an independent Poisson process and the "spontaneous" mean firing rate of the units was $\lambda = 5$ spikes/s.

It is assumed *a priori* that modifiable synapses are characterized by discrete activation levels that could be interpreted as a combination of two factors: the number of synaptic *boutons* between the pre- and post-synaptic units and the changes in synaptic conductance as a result of Ca^{2+} influx through the NMDA receptors. In the current study we attributed a fixed activation level (meaning no synaptic modification) $A_{ji}(t) = 1$, to exc–inh, inh–exc, and inh–inh synapses while activation levels were allowed to take one of $A_{ji}(t) = \{0, 1, 2, 4\}$ for exc–exc synapses, $A_{ji}(t) = 0$ meaning that the projection was permanently pruned out.

3 Simulation

The simulator was a custom, Open Source, C program that relies on the GNU Scientific Library (GSL) for random number generation and quasi-random Sobol distribution implementations. With our current implementation and setup at the University of Lausanne, a 10,000 units network simulation for a duration of $5 \cdot 10^5$ time steps lasted approximatively 3 hours, depending on the network global activity. A complete simulation lasted for $5 \cdot 10^5$ discrete time steps, corresponding to about 8.5 minutes of simulated time. After a stabilization period of 1000 ms without any external input, a 200 ms long stimulus was presented every 2,000 ms for the rest of the simulation duration, i.e. 250 presentations of the stimulus. The stimulus was composed of 10 vertical bars uniformly distributed over the 100×100 2D lattice surface, each bar being 1 column wide. At each time step, the 10 bars were simultaneously moved one column to the right, such that each bar slipped over the complete surface of the network twice per presentation. The stimulus applied on a particular unit provoked a strong depolarization that always induced the unit to discharge except if the stimulation occurred during the refractory period. For each input unit, one stimulus presentation corresponded to a sequence of 20 external inputs regularly distributed in time every 10 ms. At network level, each stimulus presentation resulted in 10 groups of about 80 synchronously spiking excitatory units repeating 20 times a 10 ms long spatio-temporal sequence. The two following presentation protocols have been applied:

Fixed stimulus. Before simulation started, 10% of the excitatory units (800 units) were randomly chosen to become permanently the input units of the network for the entire simulation.

Random stimulus. Before each stimulus presentation, 10% of the excitatory units (800 units) were randomly chosen to become the input units of the network for that particular presentation only.

4 Results

The complete status of the network was dumped at fixed intervals, providing information on the strength of the connections after the STDP–driven synaptic plasticity and pruning. Network activity was recorded as a multivariate time series formatted like a multi site spike train recordings at a resolution of 1 ms. The firing pattern of each unit could be characterized by first- and second-order time domain analyses using the programs and tools accessible from the OpenAdap.Net project[1]. We performed all simulations with both fixed and random

[1] `http://www.openadap.net/`

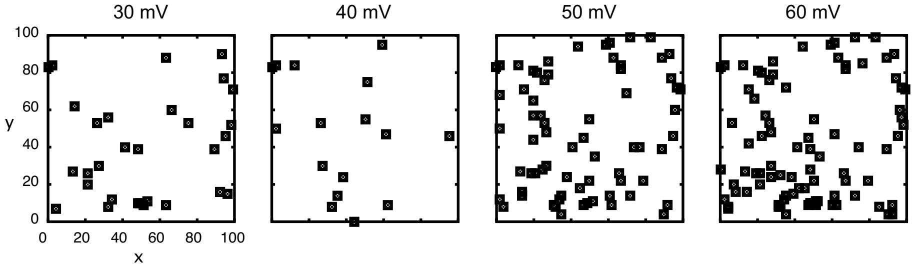


Fig. 1. Example of the location of strongly interconnected units as a function of the intensity of the fixed stimulation. Some units of this pool appeared already at low stimulation intensities but an increasing number of units appeared with an increase of the stimulation intensity.

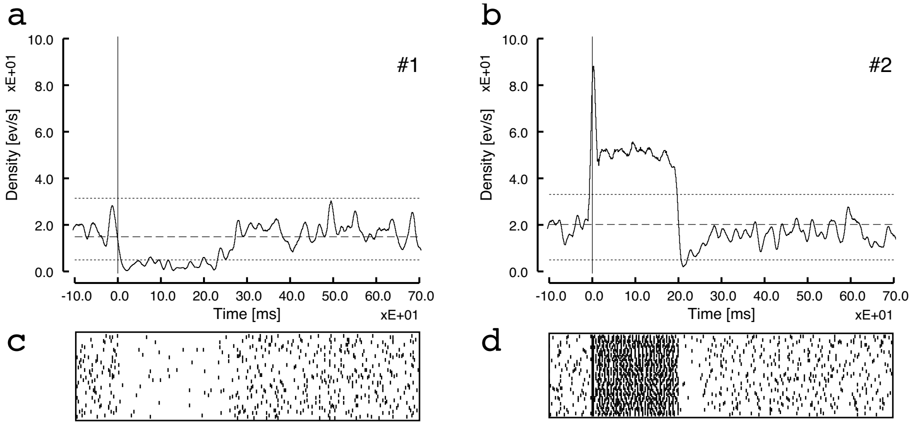


Fig. 2. Response of 2 sample strongly interconnected units responding to 50 presentations of the fixed stimulation between time $t = 450$ and $t = 500$ seconds from the simulation start. (*a*) and (*b*) peri-event densities (PSTH) for the last 50 presentations of the stimulus; (*c*) and (*d*) corresponding raster plots.

stimuli, using identical parameters and pseudo-random number generator seed. The emergence of stimulus-driven cell assemblies was determined by the comparison of the networks obtained after fixed stimulation vs. random stimulation. We considered the 7,200 excitatory units that were not directly stimulated. Among these units we could identify a group that maintained throughout the whole simulation run at least one strong (*i.e.*, $A_{ji} = 4$) incoming and one strong outgoing projection from and to other units showing the same properties. We dubbed these units *strongly interconnected units.* The count of strongly interconnected units was strongly dependent on the type of stimulation. An increase of the intensity of the fixed stimulation provoked a increase in the count of strongly interconnected units as shown in Fig. 1. On the opposite, the intensity of the random stimulation had a weak effect on the count of strongly interconnected units.

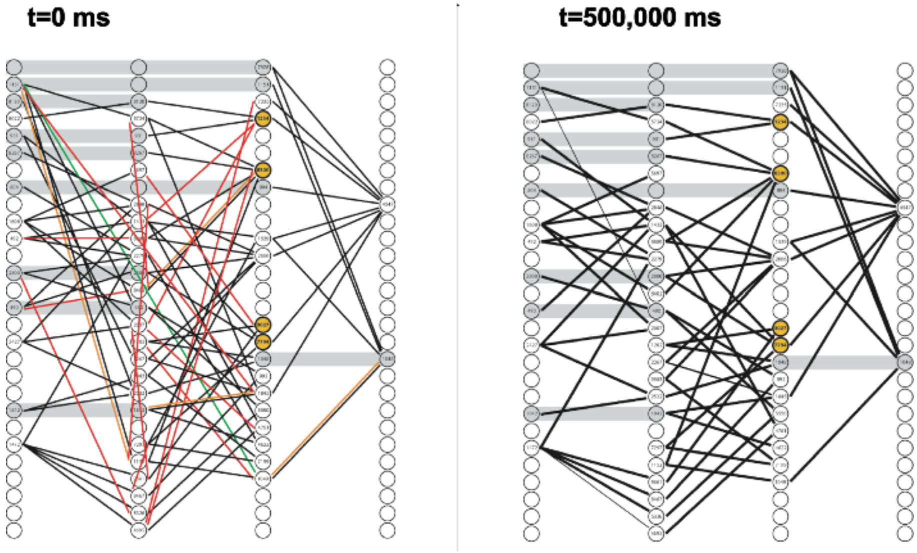


Fig. 3. Selected strongly interconnected units appear to be embedded into a layered circuit. The left panel shows the connections at the begin of the simulation and the right panel at the end. See text for more details.

The majority of the strongly interconnected units (79%) were excited by the fixed stimulation (*e.g.* Fig. 2b) despite none of these units was directly stimulated. The remaining strongly interconnected units (21%) showed a significant inhibition during the presentation, followed by an offset inhibition generally extending nearly 50 ms after the offset of the stimulus (*e.g.* Fig. 2a). The other units, that were not directly stimulated neither belonging to the strongly interconnected group, were mainly inhibited during the stimulation presentation, as the balanced network reacted to the large stimulation input by increasing the inhibition.

Fig. 3 shows the evolution of the interconnections among a group of strongly interconnected units, two of which were represented at Fig. 2. The left panel shows that at the begin of the simulation the assembly is interconnected by a mixture of feed-forward and feed-back projections. These projections were indeed set at random according to the topographic rules described elsewhere [7]. After 500,000 ms the effect of synaptic pruning driven by the stimulus and by STDP let emerge an oriented feed-forward topology because only feed-forward connections remained active.

5 Discussion

Synfire chains are diverging / converging chains of neurons discharging synchronously to sustain the propagation of the information through a feed-forward neural network [8]. This theoretical model has proven to be very efficient for the transmission of precisely timed information through the cerebral cortex but the mechanisms that may underlie its appearance in the mature brain have never been deeply investigated. This work is aimed at investigating whether synfire chains partially embedded in a large circuit characterized by initial "random" connections may emerge following activity-driven mechanisms. The rationale is that selected synaptic pruning may drive the emergence of synfire chains following certain stimulus patterns. Our results suggest that topologies compatible with synfire chains may appear during unsupervised pruning processes but further investigation is required to determine if self-sustained synfire activity may appear in the emerging networks that we have observed [9].

Acknowledgments. The authors thank Dr. Yoshiyuki Asai for discussions and comments on the results and the manuscript. This work was partially funded by the European Community Future and Emerging Technologies program, grant #IST-2000-28027 (POETIC), and Swiss grant OFES #00.0529-2 by the Swiss government.

References

1. Rakic, P., Bourgeois, J. P., Eckenhoff, M. F., Zecevic, N., Goldman-Rakic, P. S.: Concurrent overproduction of synapses in diverse regions of the primate cerebral cortex. Science **232** (1986) 232–5
2. Bi, G. Q., Poo, M. M.: Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. J Neurosci. **18** (1998) 10464-72
3. Karmarkar, U. R., Buonomano, D. V.: A model of spike-timing dependent plasticity: one or two coincidence detectors? J Neurophysiol. **88** (2002) 507–13
4. Chechik, G., Meilijson, I., Ruppin, E.: Neuronal Regulation: A Mechanism for Synaptic Pruning During Brain Maturation. Neural Computation **11** (1999) 2061–80
5. Song, S., Abbott, Larry F.: Cortical Development and Remapping through Spike Timing-Dependent Plasticity. Neuron **32** (2001) 339–50
6. Eriksson, J., Torres, O., Mitchell, A., Tucker, G., Lindsay, K., Rosenberg, J., Moreno, J.-M., Villa, A.E.P.: Spiking Neural Networks for Reconfigurable POEtic Tissue. Lecture Notes in Computer Science **2606** (2003)
7. Iglesias, J., Eriksson, J., Grize, F., T., Marco, Villa, A.E.P.: Dynamics of Pruning in Simulated Large-Scale Spiking Neural Networks. Biosystems **79** (2005) 11–20
8. Abeles, M.: Corticonics: Neural Circuits of the Cerebral Cortex. Cambridge University Press (1991)
9. Tetzlaff, T., Morrison, A., Geisel, T., Diesmann, M.: Consequences of realistic netowrk size on the stability of embedded synfire chains Neurocomputing **58-60** (2004) 117–21

An Information Geometrical Analysis of Neural Spike Sequences

Kazushi Ikeda

Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan
kazushi@i.kyoto-u.ac.jp

Abstract. Statistical measures for analyzing neural spikes in cortical areas are discussed from the information geometrical viewpoint. Under the assumption that the interspike intervals of a spike sequence of a neuron obey a gamma distribution with a variable spike rate, we formulate the problem of characterization as a semiparametric statistical estimation. We derive an optimal statistical measure under certain assumptions and also show the meaning of some existing measures, such as the coefficient of variation and the local variation.

1 Introduction

Recently, the characteristics of neurons in cortical areas have been discussed based on the statistical properties of the interspike intervals (ISIs) of a spike sequence, such as the coefficient of variation, C_V, and the skewness coefficient, S_K [1,2,3,4]. Especially, the local variation, L_V has been reported to be useful for classification of individual neurons in cortical areas since L_V is robust against changes of the spike rate [4]. However, the statistical meaning of the L_V measure has yet to be clarified. This is a cause of some difficulty for finding an optimal criterion from the information-theoretic viewpoint [5].

It is known that ISIs can be modeled as a gamma distribution [6]. From the information-geometrical viewpoint, gamma distributions form a two-dimensional e- and m-flat manifold S since they are an exponential family with two parameters [7,8]. Since the spike rate fluctuates in time and hence is useless in general, we should refer to a statistical parameter orthogonal to the parameter corresponding to the spike rate for analyzing individual neurons. In this paper, we formulate the characterization task in an information-geometrical manner and derive natural criteria from a statistical viewpoint. We also clarify the meanings of some existing measures, C_V, S_K and L_V.

2 Interspike Intervals in Spiking Neurons

When a spike sequence is given and its N ISIs are $t_1, t_2, \dots, t_N$, the C_V and S_K measures are defined as

$$C_V = \frac{\sqrt{V}}{\bar{t}}, \qquad S_K = \frac{1}{N-1}\sum_{n=1}^{N}(t_n - \bar{t})^3 \Big/ V^{3/2}, \tag{1}$$

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 133–138, 2005.

where

$$\bar{t} = \frac{1}{N}\sum_{n=1}^{N} t_n, \qquad V = \frac{1}{N-1}\sum_{n=1}^{N}(t_n - \bar{t})^2. \tag{2}$$

Since they are based on the average $\bar{t}$, both C_V and S_K tend to take a large value when the spike rate is modulated [9]. Due to this property, they are not suitable for classifying such neurons in cortical areas that change their spike rate in time.

To overcome this problem, the L_V measure is proposed in [4], defined as

$$L_V = \frac{1}{N-1}\sum_{n=1}^{N-1}\frac{3(t_n - t_{n+1})^2}{(t_n + t_{n+1})^2}, \tag{3}$$

where the factor 3 is taken so that the expectation of L_V becomes 1 when the sequence obeys a stationary Poisson process. Since the L_V measure reflects the stepwise variability of ISIs, L_V can take a small value even for a sequence with a variable spike rate. They confirmed that L_V does not undergo a large change but C_V does for a sequence generated by a time-dependent Poisson process.

Although L_V is effective in classifying neurons, it is rather heuristic except for the invariance in the time-scaling and symmetry in time [5]. Hence a theoretical background is necessary to guarantee performance and to find better criteria.

3 Information Geometry of Interspike Intervals

3.1 Information Geometry of Gamma Distributions

Information geometry [7, 8] is a general framework of Riemannian manifolds with dual affine connections, and has widely been applied to statistical inference, information theory, neural networks, and other areas. Since the manifold S of gamma distributions is an exponential family with one variable t and two parameters λ and z, we can introduce dual affine coordinate systems, the natural parameters (θ^1, θ^2) and the expectation parameters (η_1, η_2), with the Riemannian metrics g_{ij} and g^{ij} and the dual connections $\nabla^{(e)}$ and $\nabla^{(m)}$ which make the manifold dually flat [7, 8]. An important property of dual affine coorinate systems is biorthogonality, that is, the coordinate curves of θ^i and η_j for $i \neq j$ are orthogonal at any point in S. This leads us another coordinate system (η_1, θ_2) called the mixed coordinate system [10, 11]. This coordinate system is useful when we discuss projection or decomposition by virtue of biorthogonality.

The log likelihood of a gamma distribution is indeed written as

$$p(t; \lambda, z) = \frac{\lambda^z}{\Gamma(z)} t^{z-1} \exp(-\lambda t), \tag{4}$$

$$\begin{aligned} l(t; \lambda, z) &= -\log t - \lambda t + z \log t + z \log \lambda - \log \Gamma(z) \qquad (5)\\ &= f(t) + \theta^1 x_1(t) + \theta^2 x_2(t) - \Psi(\theta^1, \theta^2), \qquad (6) \end{aligned}$$

where $\Gamma(z)$ is the gamma function and

$$\theta^1 = -\lambda, \qquad x_1(t) = t, \qquad \eta_1 = \mathrm{E}[x_1(t)], \tag{7}$$

$$\theta^2 = z, \qquad x_2(t) = \log t, \qquad \eta_2 = \mathrm{E}[x_2(t)]. \tag{8}$$

Since the average of ISIs is $\mathrm{E}[t] = \mathrm{E}[x_1(t)] = \eta_1$, it is always orthogonal to $\theta^2 = z$ in the mixed coorinates. In the following, for brevity, we denote η_1 and θ^2 by η and z, respectively. When we want to exclude the effect of the average η in ISI analysis, we can simply consider z. In other words, we project a point in S into the e-flat submanifold M defined as $\eta = \eta_0$ along an m-geodesic which is orthogonal to M due to the generalized Pythagoras theorem [7, 8], where η_0 is a positive constant (Figure 1). For example, if we evaluate the distance of two points p and q in S, represented as $(\eta(p), z(p))$ and $(\eta(q), z(q))$ in the mixed coordinate, without the effect of the average, then we should consider only $z(p)$ and $z(q)$ along the submanifold M. Note that the Riemannian metric g of M is written as

$$g(z) = \mathrm{E}\left[-\frac{\mathrm{d}^2 l}{\mathrm{d}z^2}\right] = \psi'(z) - \frac{1}{z} \approx \frac{1}{2z^2} \tag{9}$$

where $\psi(z)$ and $\psi'(z)$ are the digamma and the trigamma functions, respectively, and the approximation is given by ignoring the last term of the Binet formula

$$\log\Gamma(z) = \left(z - \frac{1}{2}\right)\log z - z + \frac{1}{2}\log(2\pi) + 2\int_0^\infty \frac{\arctan(y/z)}{e^{2\pi y} - 1}\mathrm{d}y. \tag{10}$$

The distance of two points p and q, respectively represented as (η_0, z_p) and (η_0, z_q), in the one-dimensional submanifold M, is naturally defined as the length of the curve and hence is written as

$$\int_{z_p}^{z_q} \sqrt{g(z)}\mathrm{d}z \approx \frac{1}{\sqrt{2}}(\log z_q - \log z_p). \tag{11}$$

Note that the average η must be constant in the above formulation even though it does not appear in (9). If we assume its variability, we need another formulation, as discussed in the next subsection.

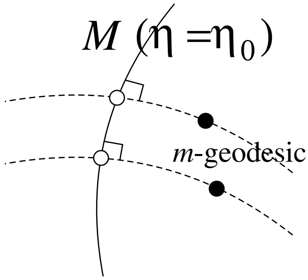


Fig. 1. Estimates (black circles) are projected to the points (white circles) on an e-flat submanifold M along m-geodesics

3.2 Semiparametric Formulation

In order to treat a variable η in time, we assume that each of interspike intervals t_n, $n = 1, \dots, N$, independently obeys a gamma distribution with parameters $(\eta^{(n)}, z)$ in the mixed coordinate (η, z) where the parameters $\eta^{(n)}$, $n = 1, \dots, N$, are randomly chosen from a fixed unknown probability density function $k(\eta^{(1)}, \dots, \eta^{(n)})$ whereas z is fixed. We would like to ascertain only z and take no interest in $\eta^{(n)}$ or $k(\eta)$. Such a problem is called semiparametric statistical estimation. It is known that the maximum likelihood method does not work well for this kind of problems. Instead, the estimating function method was proposed [12, 13], which has also been exhaustively studied from the information geometrical viewpoint [14]. However, it is still difficult to find an estimating function and to date we have not found one for ISI analysis.

4 Statistical Measures of ISIs

The difficulty of semiparametric estimation results from the total randomness of $\eta^{(n)}$. If we make some more assumptions on $\eta^{(n)}$, we can estimate and utilize them for estimation of z. In the following, we discuss how we should estimate z under a certain assumption and clarify the meaning of the C_V, S_K and L_V measures.

4.1 Spike Rate Is Constant Through Sequence

Suppose that the parameter of the spike rate, η, is fixed through a given sequence and that all ISIs are independently identically distributed. As discussed in Section 3.1, we should estimate the parameters (η, z) in the mixed coordinates from the given data and use only z in such a case. Since MLEs $(\hat{\eta}_1, \hat{\eta}_2)$ of (η_1, η_2) in the m-affine coordinates are the optimal estimators in terms of the variance of an unbiased estimator [7, 8], we calculate (η, z) from $(\hat{\eta}_1, \hat{\eta}_2)$ by coordinate transformation where

$$\hat{\eta}_1 = \frac{1}{N}\sum_{n=1}^{N} x_1(t_n) = \frac{1}{N}\sum_{n=1}^{N} t_n, \qquad \hat{\eta}_2 = \frac{1}{N}\sum_{n=1}^{N} x_2(t_n) = \frac{1}{N}\sum_{n=1}^{N} \log t_n. \tag{12}$$

This means that the estimate $\hat{z}$ of z is the solution of

$$\log \hat{\eta}_1 - \hat{\eta}_2 = \log \hat{z} - \psi(\hat{z}) \tag{13}$$

and an approximate solution using (10) is

$$\hat{z} = \frac{1}{2(\log \hat{\eta}_1 - \hat{\eta}_2)}. \tag{14}$$

From a statistical viewpoint, for estimating z it is optimal to transform the MLEs (η_1, η_2); however, the C_V measure instead employs the following two unbiased estimators

$$\zeta_1 = \frac{\theta^2}{-\theta^1}, \qquad \zeta_2 = \frac{\theta^2}{(\theta^1)^2}, \tag{15}$$

instead of the MLEs (η_1, η_2), and evaluates

$$C_V = \frac{\sqrt{\zeta_2}}{\zeta_1} = \frac{1}{\sqrt{z}} \tag{16}$$

[4] instead of the optimal criterion $\log z$ from (11). For the S_K measure, in a similar way,

$$\zeta_1 = \frac{\theta^2}{(\theta^1)^2}, \qquad \zeta_2 = \frac{2\theta^2}{-(\theta^1)^3}, \qquad S_K = \frac{\zeta_2}{(\zeta_1)^{3/2}} = \frac{2}{\sqrt{z}}. \tag{17}$$

Note that the assumption that the parameter of the spike rate is fixed through a given sequence does not hold when analyzing spike sequences of neurons in cortical areas although $\log z$ is a good measure under the assumption.

4.2 Pairwise Data Have the Same Spike Rate

We assume here that every two adjacent interspike intervals are drawn from the same distribution; in other words, the nth pair of data, t_{2n} and t_{2n+1}, in $2N$ ISIs, $t_0, \ldots, t_{2N-1}$, obey a gamma distribution with $(\eta^{(n)}, z)$ in the mixed coordinates. Under this assumption, we can estimate $(\hat{\eta}^{(n)}, \hat{z}^{(n)})$ of $(\eta^{(n)}, z)$ from t_{2n} and t_{2n+1}. Applying (14) to two-data cases, an approximated optimal $\hat{z}^{(n)}$ is

$$\log \hat{z}^{(n)} = -\log\log \frac{(t_{2n} + t_{2n+1})^2}{4t_{2n}t_{2n+1}}. \tag{18}$$

Now we have N estimates of $\hat{z}$. A reasonable way to make $\hat{z}$ from $\hat{z}^{(n)}$'s is to minimize the sum of the squared distances. Hence, the optimal estimate $\hat{z}$ under this assumption is expressed as

$$\log \hat{z} = \frac{1}{N}\sum_{n=1}^{N} \log \hat{z}^{(n)} = \left[\prod_{n=1}^{N} \log \frac{(t_{2n} + t_{2n+1})^2}{4t_{2n}t_{2n+1}}\right]^{-1/N}. \tag{19}$$

Next, we consider the meaning of an approximate $\tilde{L}_V$ of L_V, defined as

$$\tilde{L}_V = \frac{1}{N}\sum_{n=0}^{N-1} \frac{3(t_{2n} - t_{2n+1})^2}{(t_{2n} + t_{2n+1})^2}, \tag{20}$$

in which each interval appears only once. Since the summand is expressed as a function of $\hat{\eta}_1$ and $\hat{\eta}_2$,

$$\frac{3(t_{2n} - t_{2n+1})^2}{(t_{2n} + t_{2n+1})^2} = 3\left[1 - \frac{\exp(2\hat{\eta}_2)}{\hat{\eta}_1^2}\right], \tag{21}$$

$\tilde{L}_V$ can be regarded as a solution of estimation under the assumption that every two adjacent interspike intervals are drawn from the same distribution. Since the summand is an unbiased estimate of $\frac{3}{2\hat{z}^{(n)}+1}$ [4], $\tilde{L}_V$ is the arithmetic average of N estimates $\frac{3}{2\hat{z}^{(n)}+1}$, $n = 0, \ldots, N-1$. This is equivalent to employing

$$g(z) = \frac{36}{(2z+1)^4} \tag{22}$$

as the metric in (11) instead of (9).

5 Conclusions

Based on the fact that ISIs can be modeled as a gamma distribution, we derived the optimal measure and discussed the meaning of several statistical measures from the information geometrical viewpoint. Since ISIs of neurons in cortical areas have a time-variant spike rate, the characterization becomes a problem of semiparametric statistical estimation. It is, in general, difficult to solve the problem, however, if we add certain assumptions to the model, we can derive an optimal solution according to the assumptions. These assumptions are also useful to clarify the meaning of the existing measures, C_V, S_K and L_V.

References

1. Holt, G.R., et al.: Comparison of discharge variability in vitro and in vivo in cat visual cortex neurons. J. Neurophys. **75** (1996) 1806–1814
2. Shinomoto, S., Sakai, Y., Funahashi, S.: The Ornstein-Uhlenbeck process does not reproduce spiking statistics of neurons in prefrontal cortex. Neural Comp. **11** (1999) 935–951
3. Shinomoto, S., Shima, K., Tanji, J.: New classification scheme of cortical sites with the neuronal spiking characteristics. Neural Networks **15** (2002) 1165–1169
4. Shinomoto, S., Shima, K., Tanji, J.: Differences in spiking patterns among cortical neurons. Neural Comp. **15** (2003) 2823–2842
5. Miura, K., Shinomoto, S., Okada, M.: Search for optimal measure to discriminate random and regular spike trains. Technical Report NC2004-52, IEICE (2004)
6. Tiesinga, P.H.E., Fellous, J.M., Sejnowski, T.J.: Attractor reliability reveals deterministic structure in neuronal spike trains. Neural Comp. **14** (2002) 1629–1650
7. Amari, S.I.: Differential-Geometrical Methods in Statistics. Volume 28 of Lecture Notes in Statistics. Springer-Verlag (1985)
8. Amari, S.I., Nagaoka, H.: Information Geometry. Volume 191 of Translations of Mathematical Monographs. AMS and Oxford Univ. Press, Oxford, UK (1999)
9. Shinomoto, S., Tsubo, Y.: Modeling spiking behavior of neurons with time-dependent poisson processes. Physical Review E **64** (2001) 041910
10. Amari, S.I., Han, T.S.: Statistical inference under multiterminal rate restrictions: A differential geometric approach. IEEE Trans. IT **35** (1989) 217–227
11. Amari, S.I.: Information geometry on hierarchy of probability distributions. IEEE Trans. IT **47** (2001) 1701–1711
12. Godambe, V.P.: Conditional likelihood and unconditional optimum estimating equations. Biometrika **63** (1976) 277–284
13. Godambe, V.P., ed.: Estimating Functions. Oxford Univ. Press, Oxford, UK (1991)
14. Amari, S.I., Kawanabe, M.: Information geometry of estimating functions in semiparametric statistical models. Bernoulli **2 (3)** (1996)

Perceptual Binding by Coupled Oscillatory Neural Network

Teijiro Isokawa[1], Haruhiko Nishimura[2],
Naotake Kamiura[1], and Nobuyuki Matsui[1]

[1] Division of Computer Engineering, Graduate School of Engineering,
University of Hyogo, 2167 Shosha, Himeji 671-2201, Japan
{isokawa, kamiura, matsui}@eng.u-hyogo.ac.jp
[2] Graduate School of Applied Informatics, University of Hyogo,
1-3-3 Higashikawasaki-cho, Chuo-ku, Kobe 650-0044, Japan
haru@ai.u-hyogo.ac.jp

Abstract. The binding problem is a problem on the integration of perceptual properties in our brains. For describing this problem in the artificial neural network, it is necessary to introduce the temporal coding of information. In this paper, we propose a neural network model that can represent the bindings of external stimuli, based on the network that is capable of figure-ground segmentation proposed by Sompolinsky and Tsodyks. This model adopts the coupled oscillators that can represent the temporal coding and the synchronization among them.

1 Introduction

The binding problem in visual system is a problem of conjunctions of representing perceptual properties[1]. Our brains can naturally solve this problem but its mechanism has not been clarified. Since Malsburg formulated the binding problem as a theoretical one[2], much attention has been attracted to this problem by researchers among different disciplines[3,4,5,6,7].

In the early stage of the visual system, sensory signals from external sources are separated into several kinds of information with different properties. Each one of them flows into different pathways and is processed in a different region of our brains. Though several attributes of the sensory signals are processed independently in our brains, we can perceive distinct objects without discrepancy. The binding problem is that how our brains "bind" these different attributes of one object coherently.

The temporal coding of the information is necessary for describing this problem in artificial neural networks[8]. The neural network models that are made up of conventional neuron models have difficulties for the temporal coding, because these neurons adopt the mean firing rate as neural activities and cannot represent the timing among activities of neurons. Instead of using conventional neurons or neural network models, the coupled oscillatory neurons are important candidates as the model of description for the binding problem. In the coupled

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 139–144, 2005.

oscillatory neurons, the phase of oscillation is used for the activity of neurons, so it can handle the timing of neurons.

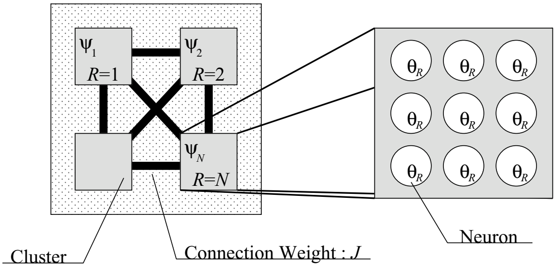
There are a lot of coupled oscillatory systems; we focus on the model proposed by Sompolinsky and Tsodyks[9] that realizes Figure-Ground discrimination of visual perception among them. While neurons of many other oscillatory models are all coherent and information is encoded only on the difference of phases among neurons, this model also utilizes the degree of coherence of neurons so that each of neurons has more flexibility.

2 Model and Method

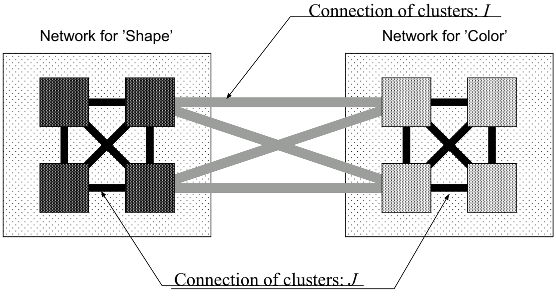
In this section, we first recapitulate the oscillatory model proposed in [9] and then extend it to the model of perceptual binding.

Figure 1(a) shows the structure of the segmentation network. A network consists of oscillatory clusters and each one of them has its phase ψ_i. The connection between clusters is denoted by J_{R_i,R_j} where R_i and R_j are the indices of the clusters i and j, respectively. A cluster is made up of neurons each of which has its preferred orientation θ that satisfies $0 \leq \theta \leq \pi$. The feature of input stimuli at the cluster R is also a form of orientation, denoted by θ^0_R.

This network works as a kind of associative memory. The connection weights are determined by the stored patterns into the network, then the input pattern



(a) A Network composed of clusters and neurons



(b) Two networks with inter-network connection I

Fig. 1. The structure of our proposed network

is imposed on the network. The phases of clusters are interacting with those of other clusters in the network. In this model, the figures can be discriminated from the background by measuring the coherency of cluster phases.

Our proposed model is an extension of this network model, shown in Fig. 1(b). This model deals with two kinds of separated information, named 'color' and 'shape', thus there are two networks, i.e., 'color network' and 'shape network'. We introduce the connection weight I as the connection of clusters between these networks. The phases of clusters are denoted by ψ_{R_s} and ψ_{R_c} where R_s and R_c are indices of the cluster in the network of shape and color, respectively. The dynamics of the cluster phase in the shape network ψ_{R_s} is described by

$$\begin{aligned}\dot{\psi_{R_s}} = \eta_{R_s}(t) - \sum_{R'_s \neq R_s} J_{R_s,R'_s} \sin[\psi_{R_s}(t) - \psi_{R'_s}(t)] \\ -\alpha \sum_{R_c} I_{R_s,R_c} \sin[\psi_{R_s}(t) - \psi_{R_c}(t)],\end{aligned} \tag{1}$$

where $J_{R,R'}$ is a connection weight between the cluster R and R' in a network, I_{R_s,R_c} is a connection weight between clusters in different networks, η is a white gaussian noise, and α is a constant. The cluster phase in the color network ψ_{R_c} can also be described similarly.

The connection weights $J_{R,R'}$ and $W_{R,R'}$ are defined as

$$J_{R,R'} = \sum_{\theta,\theta'} V(\theta - \theta^0_R) W_{R,R'}(\theta,\theta') V(\theta' - \theta^0_{R'}), \tag{2}$$

$$W_{R,R'}(\theta,\theta') = \frac{1}{nN} \sum_{\mu=1}^{P} \xi^\mu_R \xi^\mu_{R'} \tilde{V}(\theta - \theta^\mu_R) \tilde{V}(\theta' - \theta^\mu_{R'}), \tag{3}$$

where N and n are the number of clusters and the number of neurons in a cluster, respectively, and P is the number of stored patterns in a network. ξ_R defines the properties of the stored pattern, $\xi_R = 1$ and $\xi_R = 0$ mean that there exist 'figure' and 'ground' at R of the stored pattern respectively. The functions $V(x)$ and $\tilde{V}(x)$ are defined as follows:

$$V(x) = \begin{cases} 1 & |x| \leq a \\ -2a/(\pi - 2a) & a < |x| < \pi/2 \end{cases} \tag{4}$$

$$\tilde{V}(\theta_R) = V(\theta_R) - \frac{1}{n} \sum_{\theta'_R} V(\theta'_R). \tag{5}$$

Connections between clusters of different networks, I_{R_S,R_C}, W_{R_S,R_C}, are defined as:

$$I_{R_s,R_c} = \sum_{\theta_{R_s},\theta_{R_c}} V(\theta_{R_s} - \theta^0_{R_s}) W_{R_s,R_c}(\theta_{R_s},\theta_{R_c}) V(\theta_{R_c} - \theta^{0_{R_s}}_{R_c}) \tag{6}$$

$$W_{R_s,R_c}(\theta_{R_s},\theta_{R_c}) = \sum_{\nu=1}^{Q} \sum_{\mu=1}^{P} \xi^\nu_{R_s} \xi^\mu_{R_c} \tilde{V}(\theta_{R_s} - \theta^\nu_{R_s}) \tilde{V}(\theta_{R_c} - \theta^\mu_{R_c}) \tag{7}$$

where $P(= Q)$ is the number of stored patterns of shape and color information and $\theta^{0_{R_s}}_{R_c}$ is the information of colors at R_c, corresponding to the cluster R_s of shape information.

3 Simulation Result

The behavior of the proposed model is shown in this section. The clusters of the networks are 6×10 for the shape network and 2×10 for the color network. The number of neurons in each cluster is 100. Figure 2 shows the stored patterns for the shape and color networks, and the input stimuli imposed on the networks are made up of the superposition of stored patterns for each property. Note that, for the input stimuli, the patterns that have the same index for both properties are of the same object, i.e., for example, if the patterns 1, 2, and 3 of shape represent 'triangle', 'square', and 'square', respectively, and if the pattern 1, 2, and 3 of color represent 'red', 'blue', and 'yellow', respectively, then the input stimuli to be imposed are 'red triangle', 'blue square', and 'yellow square', respectively.

In our model, the degree of binding information is evaluated by using the spins of clusters S_{R_s} and S_{R_c}. The spins of clusters are defined as $S_{R_s} = (\cos(\phi_{R_s}), \sin(\phi_{R_s}))$ and $S_{R_c} = (\cos(\phi_{R_c}), \sin(\phi_{R_c}))$. If the phases of two clusters are synchronized, the time averages of their spins have a similar orientation. The representation of binding is accomplished by calculating spins of clusters in color and shape networks.

Figure 3 shows the cluster phases for the shape and color networks. A dot in the cluster expresses its cluster phase. These phases are calculated by the time-average of phases that are expressed in the form of the rectangular coordinates. We see that the phases within each of the objects are synchronized and the phases at clusters without objects(non-shaded clusters) are not synchronized, thereby the extraction of objects is realized. The phases for each of the patterns in the networks are coherent by comparing the phases between color and shape networks, thus the binding of information can be described.

The auto and cross correlograms are introduced for evaluating the degree of coherency between cluster phases. The auto-correlogram of a cluster is defined as

$$A_R(\tau) = C_R(\tau)\cos(\omega\tau), \qquad (8)$$
$$C_R(\tau) = \langle \cos[\psi_R(t) - \psi_R(t+\tau)] \rangle,$$

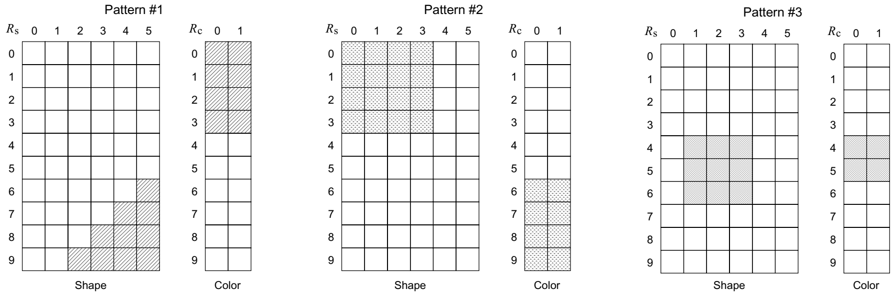


Fig. 2. Stored patterns in the shape and color networks

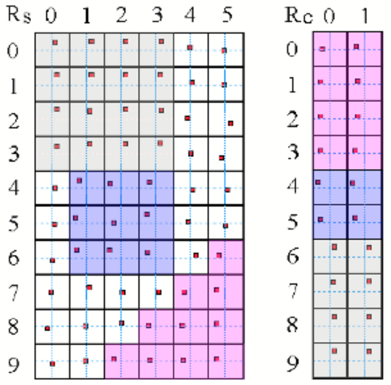


Fig. 3. Time-averaged phases of clusters

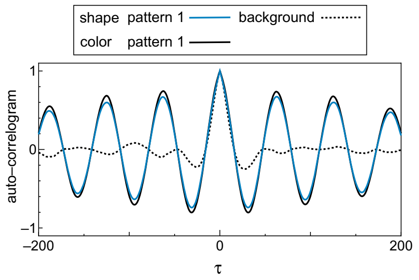


Fig. 4. Auto-correlogram of clusters in the shape and color networks

where τ and ω are the time deviation and the frequency of the cluster, respectively. Figure 4 shows the auto-correlograms of clusters that belong to the patterns and the background in the networks of color and shape. Since clusters belonging to a pattern have strong correlation while clusters of the background have weak correlation, discrimination of figures from the background can be realized as in the Sompolinsky's model.

The cross-correlogram between clusters is defined as

$$\begin{aligned} Cr_{R,R'}(\tau) &= C_{R,R'}(\tau)\cos\left[\omega\tau + \chi_{R,R'}(\tau)\right], \\ C_{R,R'}(\tau) &= \sqrt{a^2+b^2}, \\ \chi_{R,R'}(\tau) &= \arctan\left(a/b\right), \\ a &\equiv \langle \sin\left[\psi_R(t) - \psi_{R'}(t+\tau)\right]\rangle, \\ b &\equiv \langle \cos\left[\psi_R(t) - \psi_{R'}(t+\tau)\right]\rangle. \end{aligned} \tag{9}$$

Figure 5 shows the cross-correlograms of clusters belonging to the three patterns and the background with respect to a cluster belonging to pattern 1 in the shape network. The strongest correlation exists between clusters of the pattern 1 among them, clusters of the patterns 1 and 2 or of patterns 1 and 3 have moderate correlations, but little correlation is found with respect to the background. We also confirm that the cross-correlogram of clusters in the case of the color network shows similar tendencies as that in the case of the shape network.

Finally, we show the cross-correlogram of clusters that belong to the patterns and the background in the shape network with respect to a cluster that belongs to pattern 1 in the color network in Fig.6. We see that clusters that correspond to the same pattern, i.e., clusters belonging to pattern 1, have stronger correlation than clusters belonging to other patterns. This shows that integration of information between the shape and color networks is occurred, thus the binding of different attribute corresponding to the same object is accomplished.

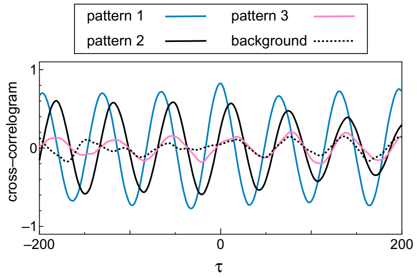


Fig. 5. Cross-correlogram between clusters in the shape network

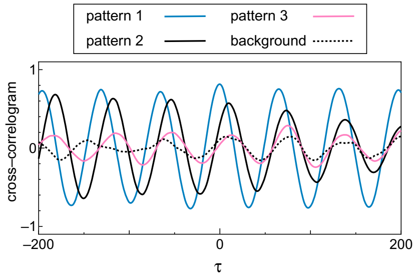


Fig. 6. Cross-correlogram between a cluster in the color network and clusters in the shape network

4 Conclusion

A neural network model for the binding problem is proposed in this paper. It adopts the coupled oscillators whose activities are expressed by their phases, thereby the temporal coding can be realized. This model contains two networks of neurons, each one of the networks being responsible for the corresponding property of the input stimulus, and where interactions of the networks are introduced. Numerical results show that in this model, objects in the scene are extracted by the synchronization of neurons in a network, and the conjunctions of properties for a certain object are expressed by the coherency between networks. Resultant dynamics exhibits the binding of information, thus this model would be a solution for the binding problem.

References

1. Roskies, A.L.: The binding problem. Neuron **24** (1999) 7–9 comprehensive reference list for all reviews, *ibid.* 111–125.
2. v. d. Malsburg, C.: The correlation theory of brain function. Technical Report 81-2, Max-Planck-Institute for Biophysical Chemistry (1981)
3. Blake, R., Yang, Y.: Spatial and temporal coherence in perceptual binding. In: Proc. Natl. Acad. Sci. U.S.A. Volume 94. (1997) 7115–7119
4. Hummel, J.E., Biederman, I.: Dynamic binding in a neural network for shape recognition. Psychological Review **99** (1992) 480–517
5. Wennekers, T., Palm, G.: On the relation between neural modelling and experimental neuroscience. Theory in Biosciences **116** (1997) 273–289
6. Engel, A.K., Fries, P., König, P., Brecht, M., Singer, W.: Temporal binding, binocular rivalry, and consciousness. Consciousness and Cognition **8** (1999) 128–151
7. v. d. Malsburg, C.: The what and why of binding: The modeler's perspective. Neuron (1999) 95–104
8. Abott, L., Sejowski, T.J.: Neural Codes and Distributed Representation. MIT Press (1999)
9. Sompolinsky, H., Tsodyks, M.: Segmentation by a network of oscillators with stored memories. Neural Computation **6** (1994) 642–657

Experimental Demonstration of Learning Properties of a New Supervised Learning Method for the Spiking Neural Networks*

Andrzej Kasinski and Filip Ponulak

Poznan University of Technology,
Institute of Control and Information Engineering,
Piotrowo 3a, 60-965 Poznan, Poland
{Andrzej.Kasinski, Filip.Ponulak}@put.poznan.pl

Abstract. In this article we consider ReSuMe - a new supervised learning method for the Spiking Neural Networks. We present the results of experiments, which indicate that ReSuMe has the following properties: (1) it can learn temporal sequences of spikes and (2) model object's I/O properties; (3) it is scalable and (4) computationally simple; (5) it is fast converging; (6) the method is independent on the used neuron models, for this reason it can be implemented in the networks with different neuron models and potentially also to the networks of biological neurons. All these properties make ReSuMe an attractive computational tool for the real-life applications such as modeling, identification and control of non-stationary, nonlinear objects, especially of the biological neural and neuro-muscular systems.

1 Introduction

Previous works on the supervised learning methods in the Spiking Neural Networks (SNN) have focused on the gradient-following approach [1],[2],[3],[4]. Yet, the explicit calculation of the gradient in SNN is problematic. In order to overcome this problem usually special simplifications are postulated. However, this constrains severely the use of the gradient-following learning methods.

Here we consider ReSuMe [5] - a new method that represents a definitely different approach to supervised learning in SNN. ReSuMe performs the goal oriented learning by adapting the idea of the learning window known from unsupervised techniques [6]. We show experimentally that ReSuMe has many interesting properties that make this method an attractive computational tool.

The goal of ReSuMe learning is to impose on a neural network the desired input-output properties. In order to describe the learning algorithm let us consider a single neuron excited simultaneously with a number of patterns through multiple synapses. The patterns may be any sequences of spikes. Consider also a single signal with the predetermined timing of spikes. This is the signal desired at the neuron output. In order to achieve the learning goal, the synaptic weights

* The work was partially supported by the State Committee for Scientific Research, project 1445/T11/2004/27.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 145–152, 2005.

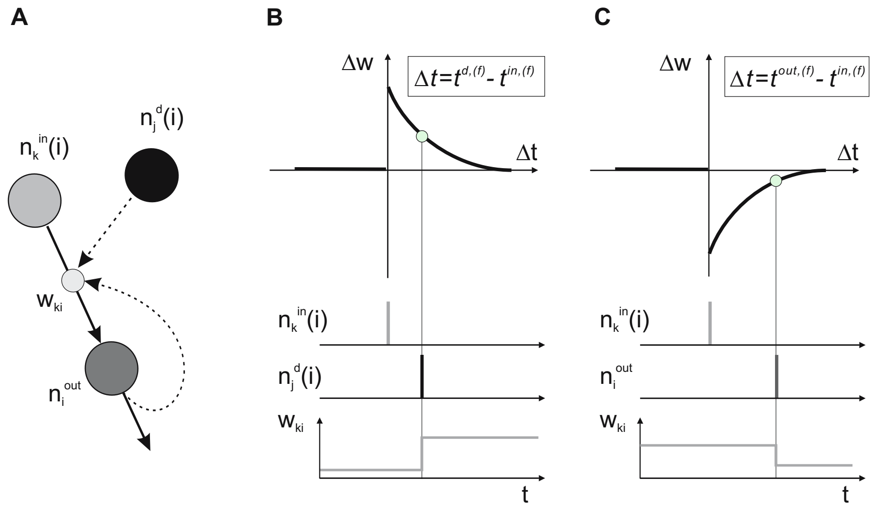


Fig. 1. (A) Basic learning neural structure consists of a presynaptic n_k^{in} and a postsynaptic neuron n_i^{out} as well as a teacher neuron n_j^d. The neurons n_k^{in} and n_i^{out} are connected via synapse w_{ki} (solid lines). The synaptic efficacy is influenced by the firing of the teacher and output neurons (dashed lines). (B),(C) Illustration of the learning rules modifying the synaptic weights. The rules are exponential functions of the time difference Δt between the desired $t^{d,(f)}$ and input $t^{in,(f)}$ spike times (B), as well as between output $t^{out,(f)}$ and input $t^{in,(f)}$ spike times (C).

should be modified in such a way that the given input patterns driving the neuron should result in the firing of the learning neuron at the times specified by the desired signal. In ReSuMe this is achieved by balancing two opposite rules determined over each synapse. These rules are expressed as functions (called learning windows) of the relative time between input and desired spikes, as well as between input and output spikes, respectively. The first rule is applied whenever a spike in the desired signal is expected. According to this rule, the excitatory (inhibitory) synapses are facilitated (depressed) if they deliver spikes directly before the desired spike time (Fig.1.B). The second rule is applied each time the learning neuron generates a spike at its output. In this rule, the excitatory (inhibitory) synapses are depressed (facilitated) if they respond directly before an output spike (Fig.1.C). The synaptic weights remain unmodified in other cases. For the formal definition of the learning rules see [5].

The learning rules are local. The desired signal can be assigned to any learning neuron separately. Therefore the learning procedure presented here holds for any learning neuron in the network.

It can be shown that the combination of the described two learning rules leads to the stable solution satisfying the goal of learning. In the presented method a neuron that delivers the desired signals (a teacher neuron) is not directly connected with the learning neural structures. However, this neuron supervises the learning synapses, i.e. it determines the synaptic weights modification. For this reason the proposed learning approach is called a Remote Supervised Method (ReSuMe).

2 Experiments

We present a set of experiments that confirm high universality of ReSuMe and its ability to impose the desired input-output properties on the learning neural network.

In our experiments we applied ReSuMe to the modified Liquid State Machine (LSM) network architecture [7],[5]. In all experiments the network consisted of a single input neuron n^{in}, the neural microcircuit (NMC) [7] with 800 units, a set of output neurons N^{out} and a corresponding set of teacher neurons N^{d} (the number of output and teacher neurons is specified in each experiment individually). The NMC receives input signals $S^{in}(t)$ from n^{in} and transforms it into a vector of signals $\widehat{S}_i^{in}(t)$ which are presented to the adequate output neurons $n_i^{out} \in N^{out}$ [5]. The particular teacher neurons $n_j^{d} \in N^{d}$ deliver to the network signals $S_j^{d}(t)$ to be taught by the output neurons. The teacher neurons are not directly connected with any other structure, however the correlation of $S^{d}(t)$ with the $\widehat{S}_i^{in}(t)$ determines the modification of the synaptic efficacy of the connections between the NMC and the n_i^{out}.

The simulations were performed in CSIM: A neural Circuit SIMulator [8]. All neurons were modeled as Leaky Integrate-and-Fire (LIF) units, if not stated otherwise. In all experiments the networks were trained over 100 learning sessions. A learning session is a process of learning, during a single presentation of a pair: input-desired output patterns.

For the l-th learning session we define a performance index $P(l)$ as an integral over a simulation time of the difference between the low-pass functions of the desired spike train $S^{d}(t)$ and the output spike train $S^{out}(t)$ generated in the l-th session [5]. The performance index can be treated as a metrics of distance between the considered spike sequences.

We also define a spike-shift error $e(t)$ as a vector of pairs $(t^{d,(f)}, \Delta t^{(f)})$, where $t^{d,(f)}$ is a time of the f-th spike in the desired spike train $S^{d}(t)$ and $\Delta t^{(f)}$ is a time difference between $t^{d,(f)}$ and time of the nearest spike in $S^{out}(t)$.

2.1 Multiple Patterns Learning

The goal of this experiment is to verify an ability of ReSuMe to learn multiple patterns and hence to model the desired input/output properties (Fig.2.A).

A single-input, single-output network has been driven with 3 different input patterns $S^{in}(t)$=#1,#2,#3 each of length 100 ms (Fig.2.B). To each input pattern a separate desired output signal has been assigned (Fig.2.D). The patterns have been presented to the network alternately.

The output signals before and after the training are depicted in Fig.2.C and Fig.2.E, respectively. The trained output sequences are visually undistinguishable from the desired patterns. All spikes are recalled correctly. The maximal

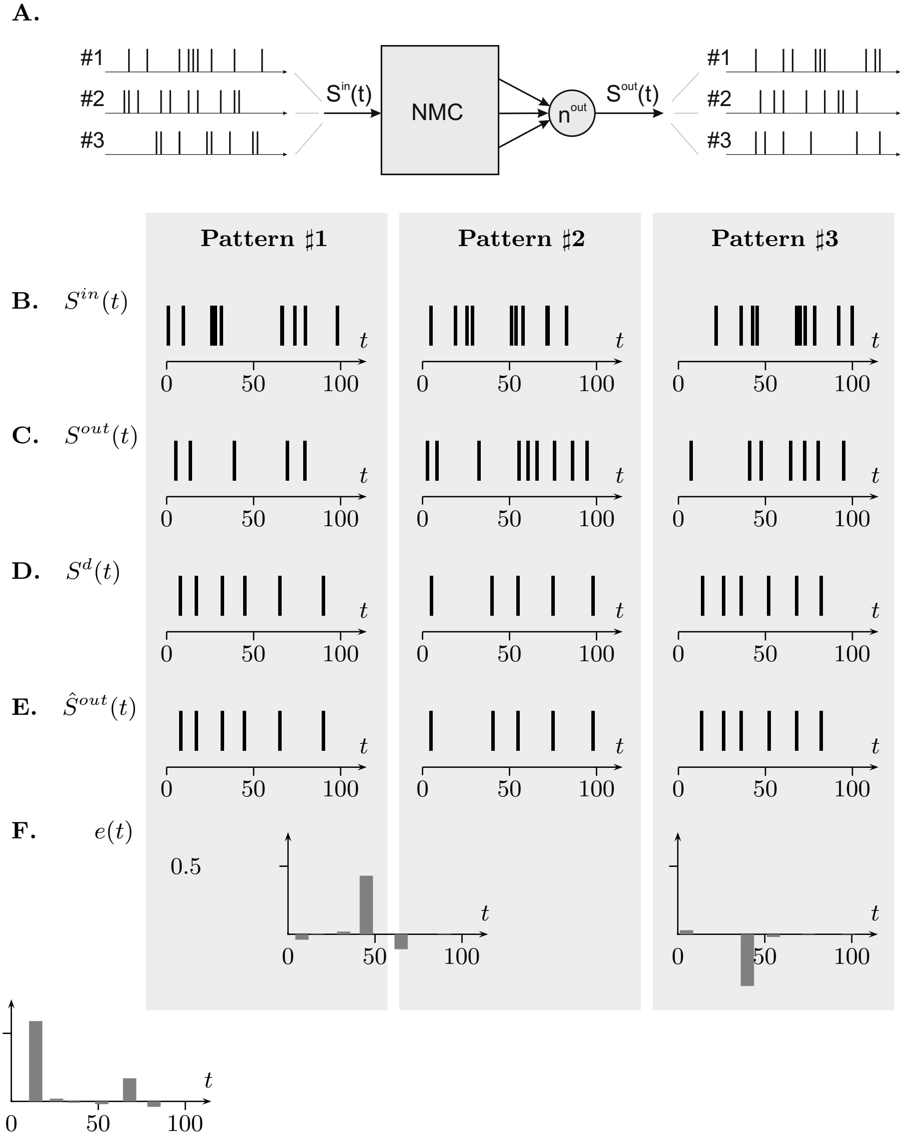


Fig. 2. Multiple patterns learning. (A) The network of 800 LIF neurons with a single input and single output is trained with 3 different pairs of input (B) and desired output (D) spike patterns ♯1, ♯2, ♯3 (each of length 100 ms). The training is performed over 100 learning sessions. The spike trains generated at the network output before the training (C) differ significantly from the desired sequences (D). After the training output signals (E) are undistinguishable from the desired ones. (F) The spike-shift errors calculated after the training are drawn as a function of the simulation time. In all three cases the maximal absolute error does not exceed 0.6 ms and an average error does not exceed 0.15 ms. Such errors are negligible as compared to 10 ms of the minimal distance between the spikes in the desired signal.

absolute spike-shift errors after the training are: 0.43, 0.38, 0.59 ms for the patterns #1,#2,#3, respectively (Fig.2.F). In all three cases an average value of an absolute error does not exceed 0.15 ms. Such errors are negligible as compared to 10 ms of the minimal inter-spike interval in the desired signal.

2.2 Multi-tasking

Consider a network with multiple readout neurons. Due to the locality of ReSuMe learning, each of the readout in this network can be trained to produce individually assigned output patterns in response to the common input signal (Fig.3.A). This is an interesting property enabling the network to perform different computational tasks simultaneously.

In our experiment the network with a single input and three readout neurons was driven with an input pattern $S^{in}(t)$ of length 300 ms (Fig.3.B) resulting in the NMC state trace $\hat{S}^{in}(t)$ (Fig.3.C). The readouts were trained to produce individually assigned signals $S_i^d(t), i = 1, 2, 3$ (Fig.3.D). Already after 50 learning sessions all spikes were correctly recalled at the network outputs (Fig.3.E). The spike-shift errors after the training (Fig.3.F) are an order of magnitude smaller than the minimal inter-spikes intervals of the desired spike sequences and can be neglected in many applications.

2.3 Neuron Model Independence

Since the ReSuMe learning method is based only on the spike times, it is expected that the method should work correctly independently on the used spiking neuron models. In order to verify this hypothesis we compare the results of the experiments in which ReSuMe is applied to the Leaky Integrate-and-Fire (Fig.4, left panel) or to the Hodgkin-Huxley (HH) model of neuron [6] (right panel).

In these experiments the networks with the single LIF and HH learning units were driven with the same input signals $S^{in}(t)$ of the length 200 ms (Fig.4.A). The generated state of NMC (Fig.4.C), the same one in both cases, was projected on the learning units. The networks were trained to produce the same spike trains $S^d(t)$ (Fig.4.B).

We present the output spike sequences $\hat{S}^{out}(t)$ after that training, the corresponding internal state time courses $V_m(t)$ ("neuron membrane potential") of the learning units, the spike-shift errors $e(t)$ and the performance quality $P(l)$ (Fig.4. D,E,F,G respectively).

The diagrams of $V_m(t)$ reveal rich dynamics and qualitatively different characters of the internal processes at the LIF and HH learning units. This had an effect on the learning process, which is manifested by the different courses of the performance index $P(l)$ in the consecutive learning sessions and by the different spike-shift errors $e(t)$ calculated after the training (with an average of the absolute error: 0.01 ms and 0.5 ms for the LIF and HH models respectively).

Nevertheless the quantitative differences of errors the learning process for both models converged toward the desired pattern and already after 80 learning sessions all spikes of the desired sequence were correctly recalled at the output of LIF as well as HH neuron model.

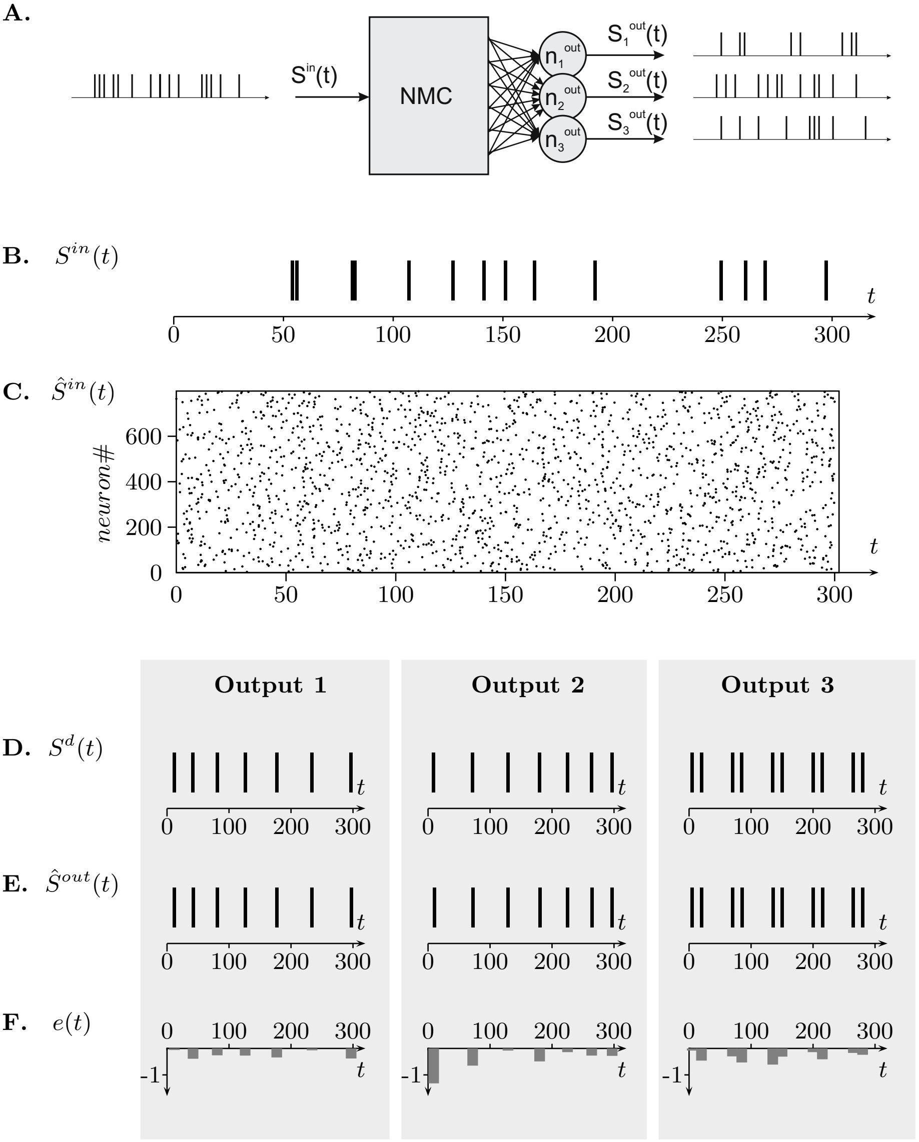


Fig. 3. Multi-tasking. This experiment demonstrates that ReSuMe learning enables the network to perform different computational tasks simultaneously. (A) The readouts are trained to produce individually assigned output patterns in response to the common input signal of length 300 ms. (B) An input signal and (C) the resulting NMC state trace. The consecutive rows of points indicate the spike trains generated by the NMC neurons (labeled here with numbers). (D) A set of the desired patterns and (E) the adequate output sequences after the training. (F) Spike-shift error $e(t)$ between the desired and output signals, with the average of the absolute error: 0.25, 0.46, 0.32 ms for the outputs 1,2,3 respectively.

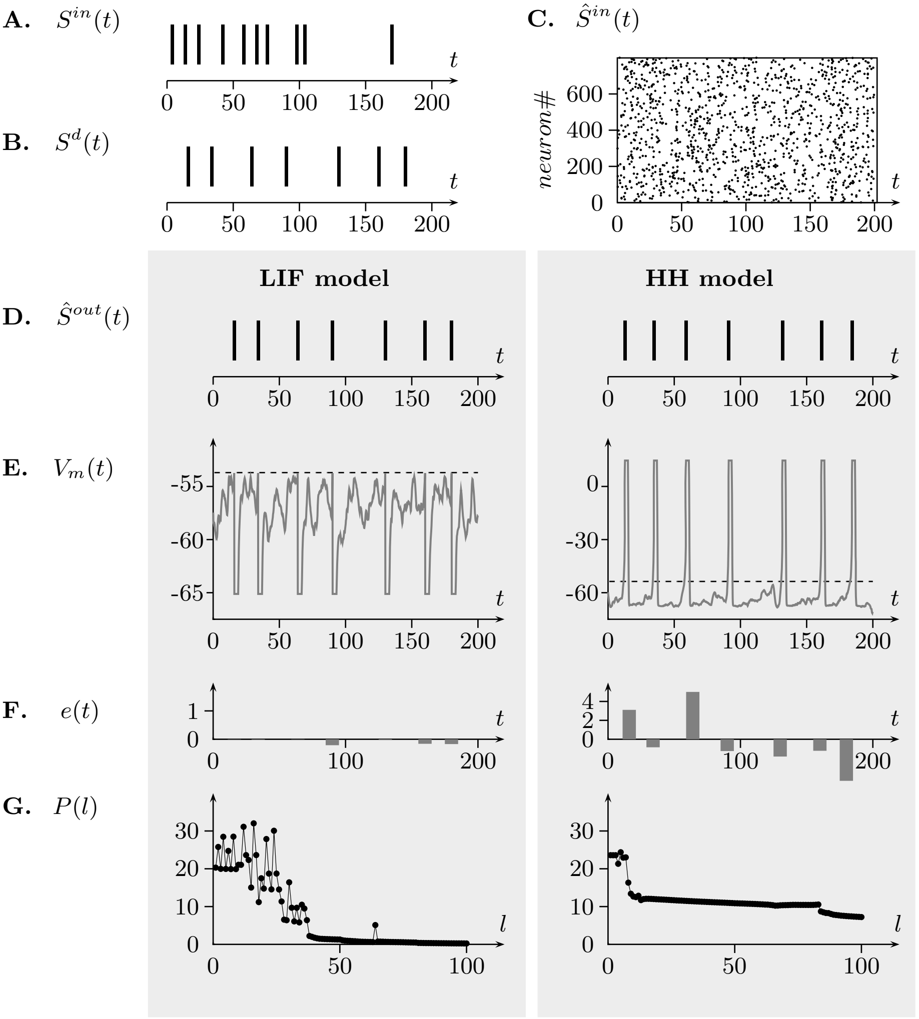


Fig. 4. Illustration of the neuron model independence of the ReSuMe learning. The learning method was applied to 2 networks with the output neurons emulated by the Leaky Integrate-and-Fire (LIF) and the Hodgkin-Huxley (HH) models respectively. Both networks where trained with the same pairs of input (A) and desired output (B) patterns of the length 200 ms. The generated state of NMC (C), the same one in both cases, was projected on the learning units. The results of learning are displayed separately for the LIF model (left panel) and for the HH model (right panel). (D) The output signals after 100 learning sessions. (E) The corresponding internal states (”membrane potentials”) of the output units reveal different characters of the internal dynamics of both outputs. The horizontal dashed lines indicate the approximate threshold values at which the neurons fire. (F) The spike shift error (with average values: 0.01 ms for LIF and 0.5 ms for HH model). (G) In both cases the performance index $P(l)$ decreases gradually in the consecutive learning sessions l, indicating that the distance between the desired and output signals vanishes.

3 Discussion

In a set of experiments we demonstrated some learning properties of ReSuMe. The results prove that our method is able not only to learn the temporal sequences of spikes with the desired accuracy, but also to model input/output properties of static objects. Moreover ReSuMe implemented in the LSM network architecture enables assigning different tasks to the individual network outputs and hence enables the parallel real-time computing. Another advantage of ReSuMe is that the method is independent on the used neuron models. For this reason it can be implemented in the networks with the combined different neuron models and potentially also to the networks of the biological neurons.

The experiments presented here confirm the suitability of ReSuMe for the eventual applications to the real-world applications in modeling, identification and control of non-stationary, nonlinear objects, especially of the biological neural and neuro-muscular systems.

References

1. Maass, W., Natschlaeger, T.: Networks of spiking neurons can emulate arbitrary hopfield networks in temporal coding. Network: Computation in Neural Systems **8** (1997) 355–372
2. Ruf, B., Schmitt, M.: Learning temporally encoded patterns in networks of spiking neurons. Neural Processing Letters **5** (1997) 9–18
3. Bohte, S., Kok, J., Potr'e, H.L.: SpikeProp: Backpropagation for Networks of Spiking Neurons. In: European Symposium on Artificial Neural Networks, ESANN. (2000) 775–777
4. Barber, D.: Learning in spiking neural assemblies. In S. Becker, S.T., Obermayer, K., eds.: Advances in Neural Information Processing Systems 15. MIT Press, Cambridge, MA (2003) 149–156
5. Ponulak, F.: ReSuMe - new supervised learning method for the Spiking Neural Networks. Technical Report, Institute of Control and Information Engineering, Poznan University of Technology (2005) Available at: `http://d1.cie.put.poznan.pl/$\sim$fp/`.
6. Gerstner, W., Kistler, W.: Spiking Neuron Models. Single Neurons, Populations, Plasticity. Cambridge University Press, Cambridge (2002)
7. Maass, W., Natschlaeger, T., Markram, H.: Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations. Neural Computation **11** (2002) 2531–2560
8. Natschlaeger, T., Markram, H., Maass, W.: Computer models and analysis tools for neural microcircuits. In Koetter, R., ed.: Neuroscience Databases. A Practical Guide. Kluwer Academic Publishers (Boston) (2003) 123–138 Chapter 9.

Single-Unit Recordings Revisited: Activity in Recurrent Microcircuits

Raul C. Mureşan[1,2,3,4], Gordon Pipa[1,2], and Diek W. Wheeler[1,2]

[1] Frankfurt Institute for Advanced Studies, Frankfurt a.M., Germany
[2] Max Planck Institute for Brain Research, Frankfurt a.M., Germany
[3] Center for Cognitive and Neural Studies (Coneural), Cluj-Napoca, Romania
[4] Technical University of Cluj-Napoca, Cluj-Napoca, Romania
raulmuresan@yahoo.com

Abstract. We investigated the relevance of single-unit recordings in the context of dynamical neural systems with recurrent synapses. The present study focuses on modeling a relatively small, biologically-plausible network of neurons. In the absence of any input, the network activity is self-sustained due to the resonating properties of the neurons. Recording of single units reveals an increasingly complex response to stimulation as one proceeds higher into the processing stream hierarchy. Results suggest that classical analysis methods, using rate and averaging over trials, fail to describe the dynamics of the system, and instead hide the relevant information embedded in the complex states of the network. We conclude that single-unit recordings, which are still extensively used in experimental neuroscience, need to be more carefully interpreted.

1 Introduction

For a long time, single-unit recordings have been the only available method of recording in experimental neuroscience. Technical difficulties imposed constraints on the experimental setups, such that recording with only a single electrode has already posed tremendous challenges for experimentalists. By using one electrode however, one can record spikes from only one cell (in the case of intra-cellular recording), or, at best, from a few cells separated by spike-sorting (when recording with an extra-cellular electrode). Only relatively recently, multi-unit recording has been introduced and used on a larger scale [9,10].

When recording from a single unit, one reliable way of analyzing the spiking activity is to observe the firing rate of the neuron. In the absence of any other information about the spiking of neurons in the surrounding populations, it seems plausible to characterize the response of single cells in terms of rate-responses to stimulation [3]. Within this framework, many categories can be defined depending on the response properties of a cell ("stimulus-presence" cell, "delayed-stimulus-presence" cell, "don't care" cell, etc).

To compensate for the variability of the spike-trains, a frequently used method is to average the response of the cell over multiple trials, thereby characterizing its response properties in terms of the average response. However, this method assumes a stationary response embedded into a randomly fluctuating signal (noisy signal). As we shall see, in the case of ongoing, non-random, history-dependent activity, the

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 153 – 159, 2005.

assumption of stationary responses is not holding anymore, especially for cells strongly coupled with populations less driven by the input (mostly cells strongly modulated by feed-back).

In order to assess the relevance of single-unit recordings, we investigated a model network of 40 neurons by tracing the activity of only one member of the simulated network at a time (in analogy to single unit recording). Although we do not claim that our model is completely biologically relevant, the results suggest that single-unit recordings have to be very carefully interpreted for a class of systems similar to our model. This might indicate that describing the dynamics of a recorded piece of cortex, consisting of a large population of cells, by a single rate response profile is a very poor approximation.

2 The Model

The artificial neural system consists of a microcircuit composed of 4 layers of neurons Interconnected in a recurrent fashion and an additional stimulation "input" layer of neurons (Fig. 1). Each layer contains 10 neurons, "resonate-and-fire" neurons [4] and "chattering" neurons [6]. The resonance properties of the neurons facilitate the self-sustained activity of the network, even in the absence of any input (there are no nonspecific background currents modeled). Although resonating neurons have been considered in only a few simulation studies so far, there is accumulating evidence that some types of real neurons exhibit resonance properties [2]. One important aspect to keep in mind is that the microcircuit activity is self-sustained without being stimulated with random non-specific background currents, so there is no added external noise.

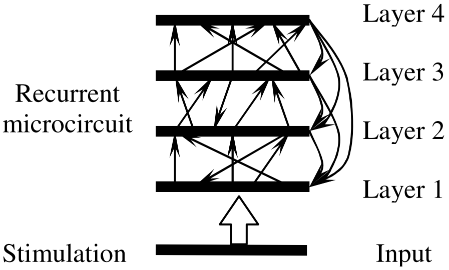


Fig. 1. The model consists of a 4-layer microcircuit with recurrent synapses and a stimulation "input" layer. Each layer is composed of 10 neurons randomly connected to higher-, lower- and same-layer neurons, with certain distributions.

The properties of the system can be summarized as follows:

1. The system has a recurrent hierarchical neural architecture.
2. The synaptic connectivity is mainly retinotopic (for ascending fibers), but lateral connectivity is also present. A neuron in one layer is connected via random synapses to higher, lower and same-layer neurons, within a local neighborhood (except feedback).

3. The receptive-field sizes increase from early to late stages of processing.
4. The neural system never rests; even in the absence of stimulation, there is spontaneous self-sustained activity.
5. Synaptic connections and strengths are randomly distributed to match the retinotopic criteria. Post-synaptic currents are modeled as exponentially decaying alpha functions, with time constants in the range of 10-20 ms.
6. The neural dynamics are mainly characterized by subthreshold oscillations near an Adropov-Hopf bifurcation [5].
7. Neuron parameters were chosen to fit cortical neuron dynamics with a distribution of 80% resonator and 20% chattering neurons (see ref. [6] for details).
8. The input signal consists of a fixed firing frequency (50 Hz) mapped onto the "input" layer (see Fig. 2).

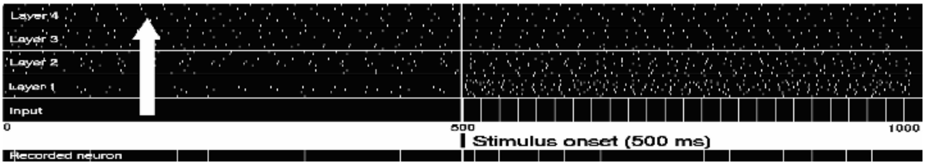


Fig. 2. A plot of the system's dynamics. The top 4 spike rasters show the activity in the 4 layers of the microcircuit. The "Input" raster displays the activity of the "input" layer, whereas the lower spike raster plots the activity of a recorded neuron from the microcircuit. In the first 500 ms, the system has self-sustained dynamics. Then, at 500 ms it is stimulated with a regular input spike-train of 50 Hz. Please note that the system has always ongoing activity, such that the time of 0 ms in the plot corresponds to 500 ms before stimulus presentation (not a real simulation time).

The dynamical system contains only 40 neurons having non-trivial dynamics (activity cannot be described in terms of classical filter responses). An important observation is that the complexity of neural activity increases with the size of the network (by complexity we mean the irregularity of the rate response). Connections are random; 80% are excitatory and 20% are inhibitory. Reversal potentials at synapses are 0 mV for excitatory and –90 mV for inhibitory synapses. Resting potential is around –70 mV, however, the thresholds are dynamic, depending on the history of stimulation (see [6]). Neurons show adaptation, occasional bursting, post-inhibitory rebound, synchronization, and other interesting biologically-plausible behavior. The simulation is continuous; between 2 stimulations, the system is not shut down, but allowed to evolve freely, self-sustained. During recording, a one second window is chosen and the stimulus is presented at the half time (500 ms). Stimulation lasts for 1500 ms, then the system is allowed to evolve for at least 2 seconds between two stimulations (to avoid short-term memory effects). The microcircuit is more realistic than other models in the sense that it "lives" between experiments/stimulations (i.e. the system has self-sustained activity), and the activity ceases only after the circuit is "killed".

For one particular microcircuit experiment, we stimulate and then record one neuron at a time, systematically from all layers, choosing the targets at random. The firing rate of the target neuron is estimated by convolving the spike train with a gaussian kernel (sd = 15 ms) [1]. Each neuron is recorded during at least 10 trials and the mean-rate activity is computed. Analysis reveals many important aspects both on the measured effects and on the method-dependence of the results, as will be shown next.

3 Results

The activity in the first layer (analogous to V1) is strongly dependent on the input. The responses are quite sharp, with little delay. The activity is mainly driven by the input (Fig. 3).

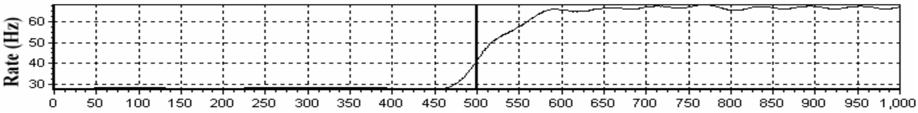


Fig. 3. Mean firing rate (average over 10 trials) of neuron #4 in layer 1. The response to the input is sharp although the neuron is strongly modulated by feedback and has quite an elevated spontaneous activity.

A small percentage of the neurons in the primary layers can show more complicated dynamics which cannot be described in terms of stimulus-locked rate increase. This is mainly due to the strong coupling with the higher layers and lateral interaction (Fig. 4).

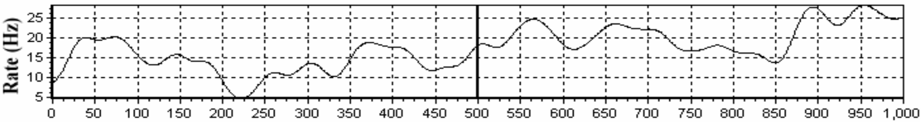


Fig. 4. Mean firing rate of neuron #1 in layer 1. In analogy with experimental data, the input has a modulatory rather than driving effect. This cell is involved in highly-dynamic processing and, averaging over many trials seems to hide the dynamic computation that is going on.

As we proceed to the higher layers, other phenomena can be observed. A smaller number of cells still have a sharp signaling profile (similar to Fig. 3). However, another type of cell emerges: the "dynamical computation" cell. These cells do not have a significant or sharp change in the rate response when the system is stimulated (Fig. 5). It seems that the worst way of analyzing their activity is to average over different trials. This is because the stationary response assumption is not fulfilled, making averaging ineffective. These cells have more a non-stationary, phase/rate variable response to the stimulation. Interpreting the averaged activity in Fig. 5 is probably a misleading method of analysis since sharp peaks and high fluctuations

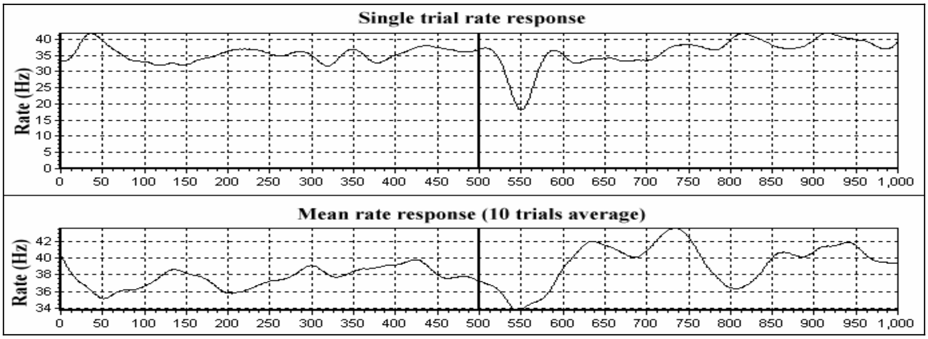


Fig. 5. Trial #10 (top) and average over 10 trials (bottom) rates for neuron #1 in layer 2. Analysis of the average rates over 10 trials reveals an inconsistent transient response (between 550–800 ms) which is only obtained by averaging over certain trials. The mean rate strongly depends both on the number of trials that are averaged (10 being a statistically low number of trials) and on the delay between two stimulations, which is reflecting a strong dependence on the history of the system. The sharp inhibition at 550 ms is either present or totally absent in different trials. Its presence in the averaged rate is only due to the fact that in the 10-trial-average, this inhibition peak was more frequent, although in a 15-trial-average experiment, it may vanish for example.

with variable phase are all averaged together. In single trial measurements however the cell's activity is obviously not changed but merely modulated.

As the neural dynamics get more and more complex, measuring the averaged rate is a less and less reliable method of analysis. We might classify the cell in Fig. 5 as a "don't care" cell, expressing that there is no obvious link between the input and the cell's rate response. However, experiments on the model show that shutting down such neurons can dramatically affect the dynamics of the entire system and its response to the stimulus. It might be that either rate is not a relevant parameter to measure for such

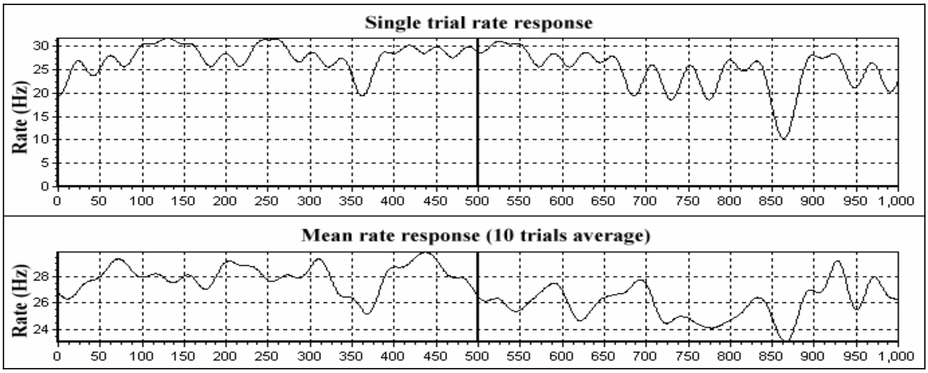


Fig. 6. Complex dynamics of an apparently "don't care cell" in layer 3. Again, the mean rate is totally irrelevant for the type of processing at the cell. It can only be inferred that the cell receives some inhibitory delayed effects from another population of neurons.

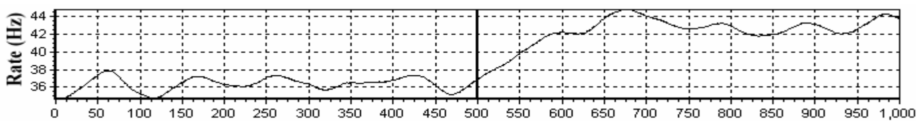


Fig. 7. Delayed signaling cell in layer 3. Mean rate response (average over 10 trials).

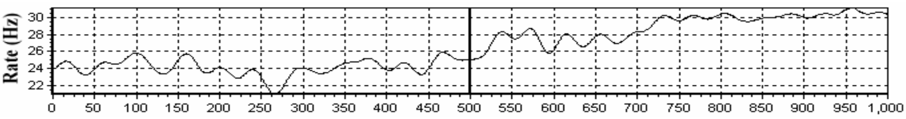


Fig. 8. An interesting cell in layer 4 that can be classified as an intermediate cell between inputs and later stages. Mean rate response (average over 10 trials).

neurons or that the activity of the neuron was critical for the overall processing performed by the population, thus indirectly influencing the response to the stimulus.

An enormous diversity of neural responses can be found in the later stages of processing due to the complexity of neural interactions (Figs. 6-8).

4 Discussion

Average rate response and neural function are an elusive pairing for complex neural interactions. It seems that rate analysis and individual neuron recordings are potentially misleading as one tries to explain the activity in higher areas (where activity is complex). When the system has nontrivial dynamics, which is the case for the neocortex, single-unit recordings and function interpretation seem to be incompatible. On one hand, averaging does not emphasize a stable rate response since fluctuations are not random and independent but related to the network dynamics and history. On the other hand, in many cases, even single trial rate responses might reveal no obvious link to the function of the neuron. This is the case when rate is not the relevant parameter to measure (ie. rate is not coding the information). Analysis of temporal patterns might prove more important in such situations [9].

For the complex activity that is going on in high visual areas like V4, we need novel methods of analysis. Complex interactions among neurons cannot be simply assumed to underlie certain functions unless we understand the entire picture. As an example, a simple readout neuron (connected to a recurrent high-dimensional microcircuit) can be trained to respond in a stable manner to the highly complex patterns of activity whenever a stimulus is presented, although understanding the highly complex dynamics of the individual neurons in the microcircuit seems elusive [7]. Moreover, it seems that such systems are highly effective computational “devices”, and the key to the power of processing is the micro-level cellular and synaptic diversity [8]. In conclusion, we suggest that in the future, single-unit recordings and analyses with averaging techniques must be interpreted with great care.

References

[1] Dayan P., Abbott L.F. Theoretical Neuroscience, MIT Press: Cambridge, MA, 2001.

[2] Fellous J.M., Houweling A.R., Modi R.H., Rao R.P.N., Tiesinga P.H.E., Sejnowski T.J. Frequency dependence of spike timing reliability in cortical pyramidal cells and interneurons. J Neurophysiol, 2001, 85: 1782–1787.

[3] Gerstner W., Kistler W.M. Spiking Neuron Models: Single Neurons, Populations, Plasticity, Cambridge University Press, New York, 2002.

[4] Izhikevich E.M. Resonate-and-Fire Neurons. Neural Networks, 2001, 14:883-894.

[5] Izhikevich E.M. Resonance and Selective Communication Via Bursts in Neurons Having Subthreshold Oscillations. BioSystems, 2002, 67:95-102.

[6] Izhikevich E.M. Simple Model of Spiking Neurons. IEEE Transactions on Neural Networks, 2003, 14:1569- 1572.

[7] Maass W., Natschläger T., Markram H. Computational models for generic cortical microcircuits. In J. Feng, editor, Computational Neuroscience: A Comprehensive Approach, chapter 18, pages 575-605. Chapman & Hall/CRC, Boca Raton, 2004.

[8] Natschläger T., Maass W., Zador A. Efficient temporal processing with biologically realistic dynamic synapses. *Network: Computation in Neural Systems*, 2001, 12:75-87.

[9] Singer W. Response synchronization: a universal coding strategy for the definition of relations. In: The Cognitive Neurosciences, M.S. Gazzaniga, ed., MIT Press, Cambridge MA, 1999.

[10] Super H., Roelfsema P., Chronic multiunit recordings in behaving animals: advantages and limitations, Progress in Brain Research, Vol. 147, 2005, pp. 263-281.

A Hardware/Software Framework for Real-Time Spiking Systems

Matthias Oster, Adrian M. Whatley,
Shih-Chii Liu, and Rodney J. Douglas

Institute of Neuroinformatics, Uni/ETH Zurich,
Winterthurerstr. 190, 8057 Zurich, Switzerland
mao@ini.phys.ethz.ch

Abstract. One focus of recent research in the field of biologically plausible neural networks is the investigation of higher-level functions such as learning, development and modulatory functions in spiking neural networks. It is desirable to explore these functions in physical neural network systems operating in real-time. We present a framework which supports such research by combining hardware spiking neurons implemented in analog VLSI (aVLSI) together with software agents. These agents are embedded in the spiking communication of the network and can change the parameters and connectivity of the network. This new approach incorporating feedback from active software agents to aVLSI hardware allows the exploration of a large variety of dynamic real-time spiking network models by adding the flexibility of software to the real-time performance of hardware.

1 Introduction

Much recent research in biologically plausible, spiking neural networks focuses on the dynamic properties of network models such as learning algorithms based on synaptic plasticity and global reward signals, development of connectivity, and modulatory functions such as gain control. It is desirable to explore such properties in a physical system in real-time: first, because such a system forces the model to include the real-time timing properties that are important for biological systems; and second, only a system that interacts with its physical environment can demonstrate that the model being studied works correctly under real-world conditions. In order to achieve this real-time behaviour, aspects of network models are often implemented in hardware [1].

We present a framework that allows the exploration of the dynamic properties of network models in real-time neural networks by combining hardware spiking neurons and software agents. Local analog and continuous-time computation is performed in the hardware, while *higher-level functionality* is implemented in software. By higher-level functionality we understand whatever algorithms are not implemented in the currently available hardware, e.g. learning algorithms based on synaptic plasticity and global reward signals, development of connectivity, and modulatory functions such as gain control. This new approach allows

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 161–166, 2005.

a wide variety of algorithms to be tested quickly and will enable real-time systems with large computational power to be assembled.

Several projects have focused on combining hardware based spiking neurons with dynamically reconfigurable connectivity: the Silicon Cortex (SCX) project [2] proposed connecting multiple chips using spiking communication and already incorporated the possibility of building integrated hardware and software models of the kind we propose here, although no such models were actually implemented at that time due to the presence of a critical bug in the host communication channel of the SCX. Similar systems are used by various groups. The IFAT system [3] aims for a similar goal to that of this work, however, we separate the hardware and software parts to achieve greater flexibility, higher performance and easier implementation of the algorithms (e.g. in MATLAB).

2 System Architecture

Hardware systems implementing the analog and continuous-time computation performed by neurons and synapses in transistor circuits have long been a subject of research and many examples can be found in the literature, e.g. [4]. The circuits approximate models of biological neurons which are then integrated in large arrays on a chip using Very Large Scale Integration (VLSI).

The connectivity between neurons is implemented by the transmission of spikes over a multiplexed bus using the address-event representation (AER) protocol [5]. Each spike is represented by the address of the source neuron or the receiving synapse and is transmitted asynchronously. A mapper translates the addresses of the sending neurons to lists of receiving synapse addresses using a look-up table, thus allowing for arbitrary intra- and inter-chip connectivity between neurons. Various networks and input sensors can be combined to form a real-time multi-chip system (see Fig. 1). A monitor translates spikes from hardware to software, while a sequencer provides the reverse translation. The mapper, monitor and sequencer are integrated on a PCI-AER board [6] which plugs into a PCI slot in a desktop computer.

Software agents embedded in this system perform the higher-level functions as defined in section 1. In this framework, an agent is an independent software process that implements a particular higher-level algorithm. At present, there are agents for analysis, on-line display, learning, modulation functions and stimulation. Multiple agents can run concurrently. Each agent communicates with the hardware neural network by receiving spike trains or activity from the network, and can change the synaptic connectivity and adjust the parameters of the neurons. Agents can also stimulate the network with artificial spike trains, providing input from parts of the system which are not implemented in hardware. Event-based agents, i.e. agents that perform computation based on single events, are implemented in C++, while agents that operate on the statistics of the activity of the network (as discussed in section 2.1) and do not require a low latency, can also be implemented in MATLAB.

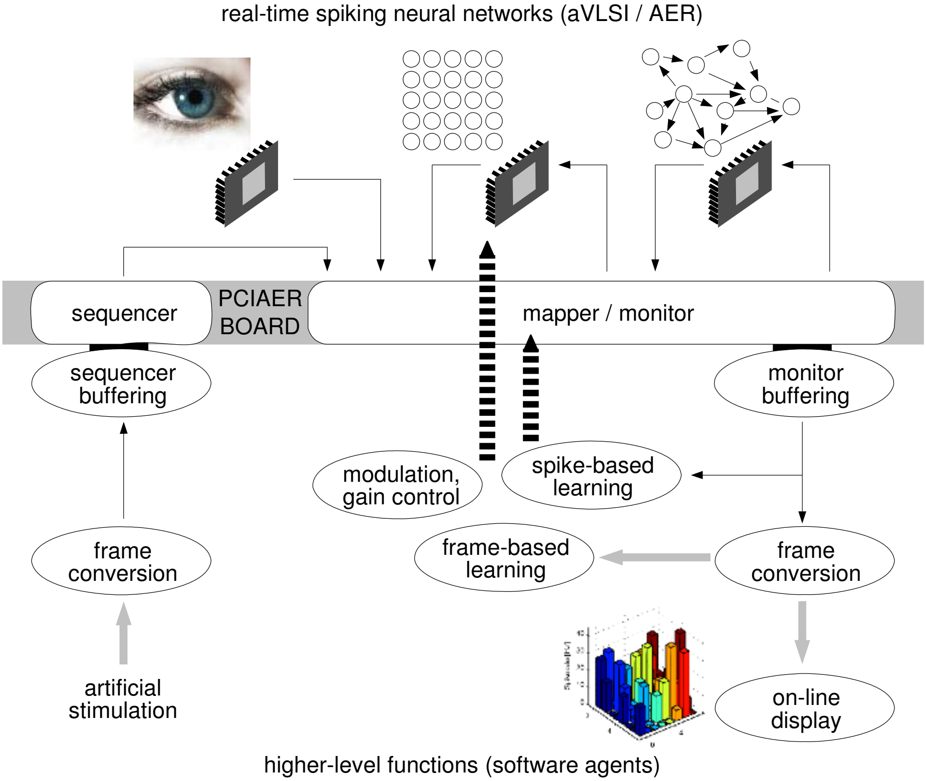


Fig. 1. Overview of the system architecture. Real-time spiking neural networks are implemented in VLSI and integrated on a chip (top). As examples, a retina, a feed-forward network and a recurrent network are shown. The neurons communicate using the address-event representation (AER) protocol (black arrows). A PCI-AER board monitors, sequences and remaps the spikes to implement the connectivity. Higher-level functions such as on-line analysis, learning algorithms, modulatory functions and artificial stimulation are implemented in C++ software agents (bottom). The agents transmit and receive spikes to and from the hardware using AER network packets and can change the connectivity and parameters of the network by modifying the mapping table and bias voltages (dashed arrows). Analysis agents transform the spike trains into a frame-based format which represents the activity of a neuron population in the chosen coding scheme (gray arrows). This allows agents implemented in slow environments such as MATLAB to be integrated into the framework. As an example, a 3D bar chart displaying the instantaneous firing rate is shown.

2.1 Software AER and Frame-Based Representation

In the software, a spike is represented by a data structure containing an address and a timestamp recorded when the spike is captured. The timestamp is required to preserve timing information when the spikes are buffered. The monitor agent sends blocks of spikes including their timestamps as network packets to receiving agents. We chose UDP for this software spiking communication because it is fast and allows several agents to receive the spike trains at the same time using multicasting.

For many applications, we are not interested in the spike train itself, but rather in the statistics of the activity of the neurons in the network. Depending on the chosen coding scheme, this can be an instantaneous spike rate, a spike count, time-to-first spike, or any other suitable measure. Analysis agents transform the spike train into one of these activity-based representations.

An agent further along the processing chain can request this activity. To do so, it first specifies the addresses of the neurons it is interested in. The analysis agent then transmits the activities of these neurons as a vector. We call this a frame-based representation. In contrast to conventional frame-based representations, the timing is asynchronous since the frame can be requested at any time and the analysis agent will calculate the contents of the frame at that time. Frames are transmitted between the agents using a TCP network connection.

The frame-based representation makes it possible to include agents in the framework that have such a long response time that they could not keep up with the real-time spike train. This allows agents to be implemented in slow environments such as MATLAB. As an example, an on-line display agent can request frames and display them with a fixed refresh rate independently of the amount of spikes received.

2.2 Learning Agents

The framework allows a variety of learning and modulation algorithms to be explored by implementing them as agents. An example of an event-driven agent is an agent that implements spike-time-dependent plasticity (STDP) [7]. The agent is configured with the addresses of the post-synaptic neurons and their presynaptic afferents. All incoming spikes are buffered and the agent checks whether a post-synaptic neuron spiked. If so, the buffer is scanned for spikes from presynaptic neurons that fall within the time window around the post-synaptic spike and long-term depression or potentiation is calculated. The synaptic efficacy is then changed on the fly in the mapper's look-up table using burst length variation [8]. Exploring STDP with this software-based approach has advantages over a hardware implementation in that the implementation time is shorter and testing is easier, since no new hardware has to be added on chip and all of the algorithm's variables are accessible.

2.3 Performance

Fig 2 shows the performance of the framework in the current state. We show both maximal values for standalone agents and values for a typical setup using an agent implementing STDP learning and a display agent. The main limitation is transferring data (spikes and synaptic weight updates) over the PCI bus. The driver for the PCI-AER board does not yet support interrupts, and the current board does not support bus mastering. Even with these limitations,

Maximum values (standalone agent)			
	rate [s^{-1}]	avg.(max.) latency	CPU load [%]
AER communication (up to 4 chips)	1.2MSpikes	1.2μs	-
Monitoring	310kSpikes	10 (80) ms	97
Synaptic efficacy updates	129kUpdates	-	98
Typical setup (multiple agents)			
Monitor agent	53kSpikes	15 (90) ms	8
STDP spikes of postsynaptic neurons	24kSpikes		35
synaptic efficacy updates	16kUpdates	44 (220) ms	
Spike-rate to frame conversion	53kSpikes	25 (120) ms	3
On-line display (MATLAB/X)	2.3	-	24

Fig. 2. Performance measurements. All rates given are maximal rates at which no or very few spikes are lost (< 1 packet in 1s). 'latency' denotes the mean (maximum) latency from a spike being recorded by the PCI-AER board until it is received and processed by an agent. All measurements were done on a standard PC (2.4GHz Pentium IV, Linux kernel 2.4.26).

the measured throughput is sufficient for many experiments because it refers to the continuous spike rate, whereas biologically plausible networks typically have short high-frequency bursts of spikes, and the average spike rate remains well below the maximum throughput.

3 Conclusion and Outlook

With its modular architecture, our framework supports multiple agents using event or activity based computation. Software spike trains are broadcast to multiple receivers and statistics relating to different spike coding schemes can be requested in a frame-based representation. Thanks to the use of standard network protocols, the system is scalable and can be distributed across several computers.

New hardware interfaces that implement the individual functionalities of the PCI-AER board in single hardware modules are being developed as part of a current project [9]. These hardware modules can be inserted where needed into the data flow of the system. They will also support much higher spike rates than the current PCI-AER board, of up to 32MSpikes/s.

The framework can be used to quickly explore higher-level functionality in a real-time system. Through the use of software agents, it provides a rapid prototyping tool to test learning and modulation algorithms in a real-time system. With input sensors such as a silicon retina, it can be used to build more complex spike-based neural network systems than presented here that are capable of reacting to their real-world environment.

Acknowledgments

We would like to acknowledge Vittorio Dante and Paolo Del Giudice (Istituto Superiore di Sanità, Rome, Italy) for the original design of the PCI-AER board, and Gerd Dietrich and other members of the Institute of Neuroinformatics involved in the development of the PCI-AER board, of its drivers, and software library components. We thank Wolfgang Einhäuser for fruitful discussions relating to the implementation of STDP. This work was supported in part by the IST grant IST-2001-34124 (CAVIAR).

References

1. Douglas, R., Mahowald, M., Mead, C.: Neuromorphic analog VLSI. Annual Review of Neuroscience **18** (1995) 255–281
2. Deiss, S.R., Douglas, R.J., Whatley, A.M.: A pulse-coded communications infrastructure for neuromorphic systems. In Maass, W., Bishop, C.M., eds.: Pulsed Neural Networks. MIT Press, Cambridge, Massachusetts (1999) 157–178
3. Vogelstein, R., Mallik, U., Cauwenberghs, G.: Beyond address-event communication: dynamically-reconfigurable spiking neural systems. In: The Neuromorphic Engineer. Volume 1. Institute of Neuromorphic Engineering (INE) (2004)
4. Douglas, R., Mahowald, M.: Silicon neurons. In Arbib, M., ed.: The Handbook of Brain Theory and Neural Networks. MIT Press, Boston (1995) 282–289
5. Lazzaro, J., Wawrzynek, J., Mahowald, M., Sivilotti, M., Gillespie, D.: Silicon auditory processors as computer peripherals. IEEE Trans. Neural Networks **4** (1993) 523–528
6. Dante, V., Del Giudice, P.: The PCI-AER interface board. In Cohen, A., Douglas, R., Horiuchi, T., Indiveri, G., Koch, C., Sejnowski, T., Shamma, S., eds.: 2001 Telluride Workshop on Neuromorphic Engineering Report. (2001) 99–103
7. Abbott, L.F., Nelson, S.B.: Synaptic plasticity: taming the beast. Nature Neuroscience **3** (2000) 1178–1183
8. Oster, M., Liu, S.C.: A winner-take-all spiking network with spiking inputs. In: 11th IEEE International Conference on Electronics, Circuits and Systems (ICECS). (2004)
9. IST-2001-34124: Caviar - convolution aer vision architecture for real-time. http://www.imse.cnm.es/~bernabe/CAVIAR (2002)

Efficient Source Detection Using Integrate-and-Fire Neurons

Laurent Perrinet

INCM/CNRS, 31, ch. Joseph Aiguier, 13402 Marseille Cedex 20, France
Laurent.Perrinet@incm.cnrs-mrs.fr
http://incm.cnrs-mrs.fr/perrinet/

Abstract. Sensory data extracted by neurons is often noisy or ambiguous and a goal of low-level cortical areas is to build an efficient strategy extracting the relevant information. It is believed that this is implemented in cortical areas by elementary *inferential* computations dynamically extracting the most likely parameters corresponding to the sensory signal. We explore here a neuro-mimetic model of the feed-forward connections in the primary visual area (V1) solving this problem in the case where the signal may be idealized by a linear generative model using an over-complete dictionary of primitives. Relying on an efficiency criterion, we derive an algorithm as an approximate solution which provides a distributed probabilistic representation of input features and uses incremental greedy inference processes. This algorithm is similar to Matching Pursuit and mimics the parallel and event-based nature of neural computations. We show a simple implementation using a network of integrate-and-fire neurons using fast lateral interactions which transforms an analog signal into a list of spikes. Though simplistic, numerical simulations show that this *Sparse Spike Coding* strategy provides an efficient representation of natural images compared to classical computational methods.

1 Cortical Processing as Solving Inverse Problems

The primary visual area in the human (V1) is a cortical area specialized in low-level visual processing from which the majority of the visual information diverges to higher visual areas. It may be regarded as a "blackboard" analyzing images so as to represent them efficiently and versatilely, as well for the detection of familiar faces than to detect complex motions. In fact, for any image, V1 has to rapidly (in the order of a fraction of a second) represent a set of features relevant to any natural image. The resulting representation, including for instance the location and orientation of the edges that outline the shape of an object, is then relayed to higher level areas to allow, for instance by grouping features, a recognition of useful patterns. Over a longer period (in the order of hours to years), the V1 area should adapt to "reverse-engineer" these scenes so as to progressively build a "model" of their structure. The success of both these learning and coding algorithms over the long term (in the order of days to generations) allows then to validate the model that was learned through the pressure of evolution. We focus in this paper on the coding algorithm and we will propose an implementation using the computational bricks provided by neural computations: parallel dynamical processing, integration of local information and event-based computations.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 167–172, 2005.

We will first define the fixed internal image model as a *Linear Generative Model* (LGM) as is often assumed for natural images [1]. It defines the space of all observed natural images $\mathcal{I} = \{\mathbf{x}\}$ that we wish to characterize as the superposition of N images of the "primitive shapes"[1] $\mathbf{A}_j = \{A_{ij}\}_{1 \leq i \leq M}$ from a dictionary $\mathbf{A}$ defined as the matrix $\mathbf{A} = \{\mathbf{A}_j; 1 \leq j \leq N\}$ at different intensities which correspond in our framework to scalar "hidden states". The image will be written $\mathbf{x} = \{x_i\}_{1 \leq i \leq M}$ over the set of spatial positions denoted by their address i and will correspond to a set of scalars $\mathbf{s} = \{s_j\}_{1 \leq j \leq N}$.

$$\mathbf{x} = \sum\nolimits_{1 \leq j \leq N} s_j . \mathbf{A}_j \tag{1}$$

In fact, particular care should be put on the dictionary. A robustness constraint to usual transform of the image suggests that the dictionary should be over-complete [2], *i.e.* that the number of dictionary elements should be of several orders of magnitude larger than the dimension of the image space (that is $N >> M$). The resulting inverse problem — finding $\mathbf{s}$ knowing $\mathbf{x}$— is in this case ill-posed.

2 Sparse Spike Coding Using a Greedy Inference Pursuit

Focusing on the event-based nature of axonal information transduction and in order to reflect the parallel architecture of the nervous system, we will here propose a solution for the inverse problem using successively two steps: Matching (M) and Pursuit (P).

(*M*) Neurons compete in parallel to find the most probable *single source* component by integrating evidence and the first source to be detected should be the one corresponding to the highest activity.

(*P*) We take into account this information before performing any further computations and then resume the algorithm for a new match (M).

2.1 Matching: Detection of the Most Probable Source Component

First, given the signal $\mathbf{x} \in \mathcal{I}$, we are searching for the *single* source $s^*.\mathbf{A}_{j^*} \in \mathcal{I}$ that corresponds to the maximum *a posteriori* (MAP) realization for $\mathbf{x}$ (and knowing it is a realization of the LGM as defined in Eq. 1). We will address in general a single source by its index and strength by $\{j, s\}$ so that the corresponding vector in $\mathcal{S}$ corresponds to a vector of zero values except for the value s at index j. The MAP is defined by:

$$\{j^*, s^*\} = \text{ArgMax}_{\{j,s\}} P(\{j, s\}|\mathbf{x}) \tag{2}$$

To compute the likelihood we will first assume that knowing one component $\{j, s\}$, the only "noise" from the viewpoint of neuron j is the combination of the unknown sources $\alpha_k, 1 \leq k \leq N$.

$$\mathbf{x} = s.\mathbf{A}_j + \nu \text{ with } \nu = \sum\nolimits_k \alpha_k.\mathbf{A}_k \tag{3}$$

For enough sources, this noise is gaussian with a covariance matrix characteristic of natural images. We may use another metric to yield a normalized spherical probability

[1] In the following, we will denote vectors and matrices by bold characters.

distribution centered around the origin [2]. From $P(\mathbf{x}|\{j,s\}) = P(\mathbf{x} - s.\mathbf{A}_j) = P(\nu)$ and the definition of conditional probability, it follows (we assume here an uniform prior across sources and scalar values)

$$\begin{aligned}\{j^*, s^*\} &= \text{ArgMax}_{\{j,s\}}[\log P(\mathbf{x}|\{j,s\}) + \log P(\{j,s\})] \\ &= \text{ArgMin}_{\{j,s\}}[s^2.\|\mathbf{A}_j\|^2 - 2.s. < \mathbf{x}, \mathbf{A}_j >] \end{aligned} \tag{4}$$

so that we found

$$j^* = \text{ArgMax}_j < \mathbf{x}, \frac{\mathbf{A}_j}{\|\mathbf{A}_j\|} > \text{ and } s^* = \frac{< \mathbf{x}, \mathbf{A}_{j^*} >}{\|\mathbf{A}_{j^*}\|^2} \tag{5}$$

Finally, as defined in Eq. 2, the source component that maximizes the probability is the projection[2] of the signal on the normalized elements of the dictionary.

2.2 Pursuit: Lateral Interaction and Greedy Pursuit of the Best Components

As we found the MAP source knowing the signal $\mathbf{x}$, we may pursue the algorithm by accounting for this inference on the signal knowing the element that we found, that is on the residual signal. In this recursive approach, we will note as n the rank of the step in the pursuit (which begins at $n = 0$ for the initialization). The first scalar projection that we have to maximize —and which will serve as the initialization of the algorithm—is given by[3] :

$$\text{(Initialization)} \boxed{C_j^{(0)} =< \mathbf{x}, \mathbf{A}_j >} \tag{6}$$

While the energy is greater than a threshold (for instance an estimate of the measurement noise), we compute :

$$\text{(Matching)} \quad \boxed{j^{(n)} = \text{ArgMax}_j [C_j^{(n-1)}]} \tag{7}$$

$$\text{(Pursuit)} \boxed{C_j^{(n)} = C_j^{(n-1)} - C_{j^{(n)}}^{(n-1)}.R_{j,j^{(n)}}} \tag{8}$$

where $R_{j,j^{(n)}} =< \mathbf{A}_j, \mathbf{A}_{j^{(n)}} >$ is the correlation of any element j with the winning element $j^{(n)}$ and relates to the reproducing kernel in wavelet theory. The greedy pursuit therefore transforms an incoming signal $\mathbf{x}$ in a list of ranked sources $\{j^{(n)}, s^{(n)}\}$ such that finally the signal may be reconstructed as

$$\mathbf{x} = \sum\nolimits_{k=1...n} s^{(k)}.\mathbf{A}_{j^{(k)}} + \mathbf{x}^{(n)}$$

which is an approximation of the goal set in inverting Eq. 1 as the norm of the residual signal $\mathbf{x}^{(n)}$ converges to zero (see this theorem and other properties of the MP algorithm in [2]).

[2] By symmetry, we could choose the absolute value to modelize ON and OFF neurons with similar receptive profiles.

[3] Since the choice of j^* is independent of the norm of the filters, we have normalized them all to 1 for clarity. However, we can weight the preference for different neurons by adding a multiplicative gain in Eq. 7.

2.3 Implementation Using Integrate-and-Fire (IF) Neurons

We will derive an implementation of this algorithm using a network of spiking neurons based on the same feed-forward architecture as the perceptron but implementing the greedy pursuit using *lateral interactions*. The activity is represented by a driving current $C_j(t)$ that drives the potential V_j of leaky Integrate-and-Fire neurons [3] from the initialization time. For illustration purposes, the dynamics of the neurons will here be modeled by a simple linear integration of the driving current C_j (other monotonic integration schemes lead to similar formulations):

$$\tau.\frac{d}{dt}V_j = -V_j + R.C_j \tag{9}$$

where τ is the time constant of membrane integration and R a gain (in Ohm if C_j is a current). Neurons generate a spike when their potential reach an arbitrary threshold that we set here to 1. In our framework, the image is presented at an initial time and the activities are constant for $t \geq 0$ (see Eq. 6). From the monotonous integration and while at least one activity is greater than one, the next neuron to generate a spike will be

$$j^* = \mathrm{ArgMax}_j[C_j] \tag{10}$$

with an interspike interval of

$$t^* = \tau.\log(\frac{1}{1-1/C_{j^*}}) \tag{11}$$

To implement the greedy algorithm, we then need to implement a lateral interaction on the neighboring neuron similar to the observed lateral propagation of information in V1 by updating activities of all neurons by

$$C_j \leftarrow C_j - C_{j^*}.R_{\{j,j^*\}} \tag{12}$$

and therefore of the potential of every neuron by $C_{j^*}/R_{\{j,j^*\}}.(1-e^{-t^*/\tau})$, that is simply:

$$V_j \leftarrow V_j - R_{\{j,j^*\}} \tag{13}$$

and then resume the algorithm. This lateral interaction is here immediate and behaves as a refractory period on the winning neuron ($C_{j^*} \leftarrow 0$ and $V_{j^*} \leftarrow 0$) but also on correlated neurons. In this case, piecewise jumps of activity will lead to piecewise exponential traces for the membrane potential, interrupted at the spike times. We have shown that this simple architecture provides an explanation for some complex behavior of cooperation in the cortex as the constancy of the selectivity tuning of simple cells in V1 [4]. Finally, we have shown that this simple implementation implements the Matching Pursuit algorithm that we defined in Eq. 7 and 8.

3 Results: Efficiency of Sparse Spike Coding

We compared the method we described in this paper with similar techniques used to yield sparse and efficient codes such as the conjugate gradient method used by Olshausen [1]. We used a similar context and architecture as these experiments and used in

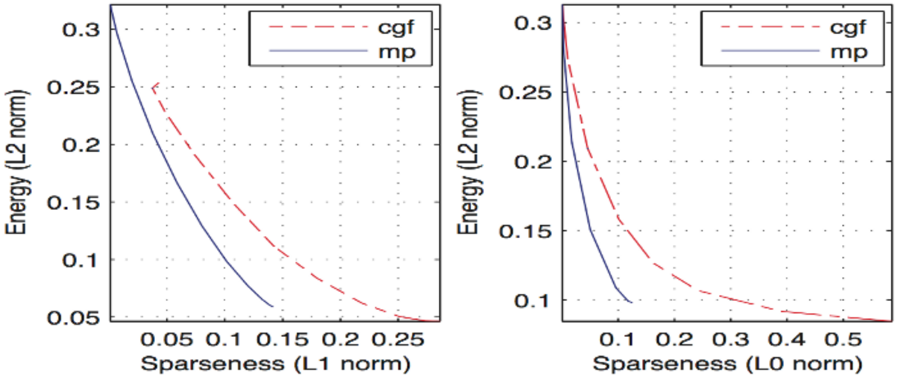


Fig. 1. Efficiency of the matching pursuit compared to conjugate gradient. We compared here the matching pursuit ('mp') method with the classical conjugate gradient function ('cgf') method as is used in [1] for different dictionaries. We present the results for the coding of a set of image patches drawn from a database of natural images. These results were obtained with the same fixed dictionary of edges for both methods. We plot the mean final residual error for two definitions of sparseness: **(Left)** the mean absolute sum of the coefficients and **(Right)** the number of active (or non-zero) coefficients (the coding step for MP). For this architecture, the sparse spike coding scheme appears to be adapted to efficiently code natural images and in particular to compress the information needed to code the image.

particular the database of inputs and the dictionary of filters learned in the SPARSENET algorithm. Namely, we used a set of 10^5 10×10 patches (so that $M = 100$) from whitened images drawn from a database of natural images. The weight matrix was computed using the SPARSENET algorithm with a 2-fold over-completeness ($N = 200$) that show similar structure as the receptive of simple cells in V1. From the relation between the likelihood of having recovered the signal and the squared error in the new metric, the mean squared reconstruction error (L2-norm) is an appropriate measure of the coding efficiency for these whitened images. This measure represents the mean accuracy (in terms of the logarithm of a probability) between the data and the representation. We compared here this measure for different definitions and values for the "sparseness".

First, by changing an internal parameter tuning the compromise between reconstruction error and sparsity (namely the estimated variance of the noise for the conjugate gradient method and the stopping criteria in the pursuit), one could yield different mean residual error with different mean absolute value of the coefficients (see Fig. 1, left) or L1-norm. In a second experiment, we compared the efficiency of the greedy pursuit while varying the number of active coefficients (the L0-norm), that is the rank of the pursuit. To compare this method with the conjugate gradient, a first pass of the latter method was assigning for a fixed number of active coefficients the best neurons while a second pass optimized the coefficients for this set of "active" vectors (see Fig. 1, right).

Computationally, the complexity of the algorithms and the time required by both methods was similar. However, the pursuit is by construction more adapted to provide a progressive and dynamical result while the conjugate gradient method had to be recomputed for every set of parameter. Best results are those giving a lower error for a given sparsity or a lower sparseness (better compression) for the same error. In both cases, the Sparse Spike Coding provides a coding paradigm which is of better efficiency as the conjugate gradient.

4 Conclusion

We presented here a model for neural processing which provides an alternative to the feed-forward and spike-rate coding approaches. Focusing on the parallel architecture of cortical areas, we based our computations on spiking events which represent successive elementary decision processes. We propose a simple implementation which exhibit efficient and complex dynamics. This model thus provides an algorithm of *Sparse Spike Coding* which is particularly adapted to understand computational aspects of the neural code for low-level visual tasks.

This simple strategy thus suggest that the inherent complexity of the neural activity is perhaps not the consequence of the computational complexity of detailed models of neurons but may rather be the consequence of the parallel event-based dynamics of the neural activity. Although our model is a simplistic caricature compared to the behavior of biological neurons, it provides a simple algorithm which is compatible with some complex characteristic of the response of neuronal populations. It thus proposes a challenge for discovering the mechanisms underlying the efficiency of nervous systems by focusing on the emergent computational properties of large-scale networks of spiking neurons.

Reproducible Research. Scripts reproducing the figure may be obtained from the author upon request.

References

1. Olshausen, B., Field, D.J.: Sparse coding with an overcomplete basis set: A strategy employed by V1? Vision Research **37** (1998) 3311–25
2. Perrinet, L., Samuelides, M., Thorpe, S.: Coding static natural images using spiking event times : do neurons cooperate? IEEE Transactions on Neural Networks, Special Issue on 'Temporal Coding for Neural Information Processing' **15** (2004) 1164–75 `http://incm.cnrs-mrs.fr/perrinet/publi/perrinet03ieee.pdf`.
3. Lapicque, L.: Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarisation. Journal of Physiology, Paris **9** (1907) 620–35
4. Perrinet, L.: Feature detection using spikes : the greedy approach. Journal of Physiology, Paris (2004) `http://incm.cnrs-mrs.fr/perrinet/publi/perrinet04tauc.pdf`.

A Model for Hierarchical Associative Memories via Dynamically Coupled GBSB Neural Networks

Rogério M. Gomes[1], Antônio P. Braga[2], and Henrique E. Borges[1]

[1] CEFET/MG, Laboratório de Sistemas Inteligentes,
Av. Amazonas, 7675 - Belo Horizonte - MG - Brasil - CEP 30510-000
{rogerio, henrique}@lsi.cefetmg.br
[2] PPGEE-UFMG, Laboratório de Inteligência e Técnicas Computacionais,
Av. Antônio Carlos, 6627 - Belo Horizonte - MG - Brasil - CEP 31270-010
apbraga@cpdee.ufmg.br

Abstract. Many approaches have emerged in the attempt to explain the memory process. One of which is the Theory of Neuronal Group Selection (TNGS), proposed by Edelman [1]. In the present work, inspired by Edelman ideas, we design and implement a new hierarchically coupled dynamical system consisting of GBSB neural networks. Our results show that, for a wide range of the system parameters, even when the networks are weakly coupled, the system evolve towards an emergent global associative memory resulting from the correlation of the lowest level memories.

Keywords: Hierarchical memories, Coupled neural networks, Dynamical systems, Artificial neural networks, TNGS.

1 Introduction

Presently, studies in neuroscience have revealed, by means of experimental evidences, that memory process can be described as being organized, functionally, in hierarchical levels, where higher levels would coordinate sets of functions of the lower levels [1] [2]. One of the theories that is in compliance with these studies is the Theory of Neuronal Group Selection (TNGS) proposed by Edelman [2].

TNGS establishes that correlations of the localized neural cells in the cortical area of the brain, generate clusters units denoted as: neuronal groups (cluster of 50 to 10.000 neural cells), local maps (reentrant clusters of neuronal groups) and global maps (reentrant clusters of neural maps).

A neuronal group (NG) is a set of tightly coupled neurons which fire and oscillates in synchrony. Each neuron belongs only to a single neuronal group, which is spatially localized and functionally hyper-specialized. According to TNGS, NG are the most basic structures in the cortical brain, from which memory and perception processes arise, and can been seen as performing the most primitive sensory-effector correlations.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 173–178, 2005.

A local map is a composition of NG, also spatially localized in the cortical area. Two local maps, functionally different, can develop reentrant connections, resulting in what Edelman [1] calls categorization. Edelman [1] states that a significant number of different neuronal groups could have the same functionality within a given map, that is, could respond to the same stimuli.

A global map is a dynamic structure containing multiple reentrant local maps which are capable of interacting with non-mappable areas of the brain, such as the limbic system [2]. A global map is a set of connected local maps and perform "categorizations" (correlations) of local maps. They are not spatially localized but, in fact, they are spread throughout the cortex. Global maps provide a "global or emergent behaviour" of the cortical activities (perception in action) and generate a complete experience in the world, *i.e.*, an experience with *qualia.* A continuous selection of existing local maps in a global map by selection of additional reentrant connection allows forming new classification couples.

Inspired by these ideas, a model of hierarchically coupled dynamical system, using GBSB (Generalized-Brain-State-in-a-Box) neural networks is described in this paper, which integrates the concepts of dynamical systems theory, TNGS and Artificial Neural Networks (ANNs) aiming at building multi-level memories.

This paper is organized as follows. In Section 2 we propose a model of coupled GBSB neural networks and show how multi-level memories may arise within it. Section 3 illustrates the use of the algorithm developed in Section 2 with an example from the literature [3] [4]. Finally, Section 4 concludes the paper and presents some relevant extensions of this work.

2 Proposal for the Construction of Multi-level Memories

In order to develop this new model we use an extension of the original ***BSB - Brain-State-in-a-Box*** [5] called ***GBSB*** (*Generalized-Brain-State-in-Box*) [6]. The behaviour of the neural network energy in a discrete BSB model was studied by Golden [7]. Cohen and Grossberg [8] discussed a continuous BSB model based on Liapunov equations, while Hui and Zak [6] discussed the stability of the GBSB model in a non-symmetric diagonally dominant weight matrix case.

The GBSB model was chosen due to the fact that its characteristics, which are, in short: asymmetric synapses, different *bias* and maximum and minimum fire rates, redundancy, non-linear dynamics and self-connection for each neuron.

In our proposed model we build a two level hierarchical memory where, in accordance with figure 1, each one of the GBSB networks (A, B and C) plays the role of a neuronal group or, in our case, a first-level memory. In a given cluster, each neuron performs synapses with each other neuron of the same cluster, *i.e.*, the GBSB is a fully connected asymmetric neural network. Beyond this, some selected neurons in a cluster are bidirectionally connected with some selected neurons in the others clusters [4]. These inter-cluster connections can be represented by a weight correlation matrix W_{cor}, which accounts for the contribution of one cluster to another one due to coupling. An analogous procedure could be followed in order to build higher levels in the hierarchy [1].

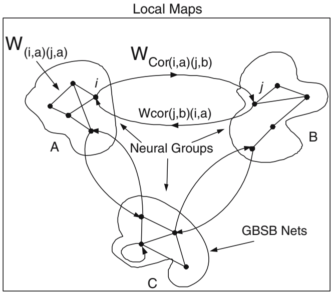


Fig. 1. Network design

Our proposed coupled GBSB model extends the GBSB model for single networks discussed in [6], by means of adding a fourth term that represents the inter-group connections. Consequently, our new model can be defined by the equation:

$$x_{(i,a)}^{k+1} = \varphi \left(x_{(i,a)}^{k} + \sum_{j=1}^{N_a} \beta_{(i,a)} w_{(i,a)(j,a)} x_{(j,a)}^{k} + \beta_{(i,a)} f_{(i,a)} + \right. \\ \left. + \sum_{\substack{b=1 \\ b \neq a}}^{N_r} \sum_{j=1}^{N_q} \gamma_{(i,a)(j,b)} w_{cor(i,a)(j,b)} x_{(j,b)}^{k} \right) , \tag{1}$$

where the three first terms represent the equations of a GBSB model for N_a uncoupled GBSB networks, meaning in our model intra-group synapses (*i.e.*, in the a^{th} network or neuronal group). The sum over j, in the fourth term, labels the N_q neurons in the b^{th} neuronal group that have correlation to neuron i in the a^{th} neuronal group. The strength or density of the inter-group synapses are parameterized by $\gamma_{(i,a)(j,b)}$. The activation function φ is a linear saturating function whose i^{th} component is defined as follows:

$$x_{(i,a)}^{k+1} = \varphi(y_{(i,a)}^{k}), \qquad \varphi(y_{(i,a)}^{k}) = \begin{cases} +1 & \text{if } y_{(i,a)}^{k} > +1 \\ y_{(i,a)}^{k} & \text{if } -1 \leq y_{(i,a)}^{k} \leq +1, \\ -1 & \text{if } y_{(i,a)}^{k} < -1 \end{cases} \tag{2}$$

where $y_{(i,a)}^{k}$ is the argument of the function φ of the equation 1.

In order to complete our model we present now a Lyapunov function (energy) of the coupled system, which can be defined as [9]:

$$E = -\tfrac{1}{2} \left[\sum_{a=1}^{N_r} \sum_{i,j=1}^{N_a} \beta_{(i,a)} w_{(i,a)(j,a)} x_{(i,a)} x_{(j,a)} \right] - \sum_{a=1}^{N_r} \sum_{i=1}^{N_a} \beta_{(i,a)} f_{(i,a)} x_{(i,a)} \\ - \sum_{\substack{a,b=1 \\ a \neq b}}^{N_r} \sum_{i=1}^{N_a} \sum_{j=1}^{N_q} \gamma_{(i,a)(j,b)} w_{cor(i,a)(j,b)} x_{(i,a)} x_{(j,b)} , \tag{3}$$

where the first term represents the energy of the individual neuronal groups. The second term gives the contribution to energy due to external factors (*i.e.*, the *bias field*). Finally, the third term in equation 3 is due to the inter-group connections. A detailed mathematical analysis of equation 3, describing the energy of the coupled system can be found in [9], where it was shown that it presents two important features: the whole system evolves to a state of minimum energy, even when the neuronal groups are weakly coupled; the inter-group coupling, which establishes the second-level correlations, does not destroy the first-level memories structures.

3 Coupled GBSB Experiments

Computational experiment consisting of three GBSB networks connected (Fig. 1) were conducted and the results were compared with the ones presented in [4]. Although the experiment presented here is a quite simple one, it is intended to make it clear the procedure for the construction of multi-level associative memories. More complex computational experiments will be presented elsewhere. In our simulations each network or neuronal group contains 10 neurons and we selected 6 out of 1024 possible patterns to be stored as our first-level memories. The weight matrix of the individual networks followed the definition proposed in [3]. The selected set of patterns stored as first-level memories was:

$$\begin{array}{ll} V_1 = [\,\text{-1}\;\;1\;\;1\;1\;\;1\;\;1\;\text{-1}\;\text{-1}\;\text{-1}\;\text{-1}\,] & V_2 = [\;\;1\;1\;\text{-1}\;\text{-1}\;\text{-1}\;\;1\;\text{-1}\;\text{-1}\;\;1\;\text{-1}\,] \\ V_3 = [\,\text{-1}\;\;1\;\;1\;1\;\text{-1}\;\text{-1}\;\;1\;\text{-1}\;\text{-1}\;\text{-1}\,] & V_4 = [\,\text{-1}\;1\;\text{-1}\;\text{-1}\;\text{-1}\;\text{-1}\;\;1\;\text{-1}\;\;1\;\;1\,] \\ V_5 = [\;\;1\;\text{-1}\;\text{-1}\;1\;\;1\;\text{-1}\;\;1\;\;1\;\;1\;\text{-1}\,] & V_6 = [\;\;1\;1\;\text{-1}\;\;1\;\text{-1}\;\;1\;\;1\;\;1\;\text{-1}\;\text{-1}\,] \end{array} \tag{4}$$

Each network A, B and C was carefully designed to present the same asymptotically stable fixed point structure presented in [3]. To design these networks we followed the approach of [4]. In addition, amongst the $6^3 = 216$ possible combinations of the 3 sets of first-level memories, we have chosen 2 triplets or global patterns to be our second-level memories (local maps). The arrangement of the global patterns determines the inter-group correlation matrix W_{cor} by a generalized Hebb rule.

The system was initialized in one of the networks A, B or C, randomly, and in one of their first-level memories that establish a correlation in accordance with table 1. The two other networks, in turn, were initialized in one of the 1024 possible patterns, also, randomly. Then, we measured the number of times that the system converges to a configuration of triplets[1], considering networks totally or partially coupled. Neurons that took part of the inter-group connections was chosen randomly. Points in our experiments were averaged over 1000 trials for each value of γ.

In our experiment a typical value of β was chosen ($\beta = 0.3$) and the number of correlation (triplets) obtained, as a function of γ, was measured considering that 0%, 20%, 60% and 100% of the inter-group neurons were connected. The results

[1] Triplet is one of the global patterns selected that constitutes a second-level memory.

Table 1. Hebbian rule for correlation of the first-level memories, where $V_{(i,a^{th})}$ - $V_{(j,b^{th})}$ is the correlation between the i^{th} and the j^{th} pattern of the a^{th} and of the b^{th} network, respectively

Inter-Groups	$V_{(j,A)}$	$V_{(j,B)}$	$V_{(j,C)}$
$V_{(i,A)}$		$V_{(1,A)}$ - $V_{(3,B)}$ $V_{(2,A)}$ - $V_{(4,B)}$	$V_{(1,A)}$ - $V_{(5,C)}$ $V_{(2,A)}$ - $V_{(6,C)}$
$V_{(i,B)}$	$V_{(3,B)}$ - $V_{(1,A)}$ $V_{(4,B)}$ - $V_{(2,A)}$		$V_{(3,B)}$ - $V_{(5,C)}$ $V_{(4,B)}$ - $V_{(6,C)}$
$V_{(i,C)}$	$V_{(5,C)}$ - $V_{(1,A)}$ $V_{(6,C)}$ - $V_{(2,A)}$	$V_{(5,C)}$ - $V_{(3,B)}$ $V_{(6,C)}$ - $V_{(4,B)}$	
Global Patterns Selected $V_{(1,A)}$ - $V_{(3,B)}$ - $V_{(5,C)}$ or $V_{(2,A)}$ - $V_{(4,B)}$ - $V_{(6,C)}$			

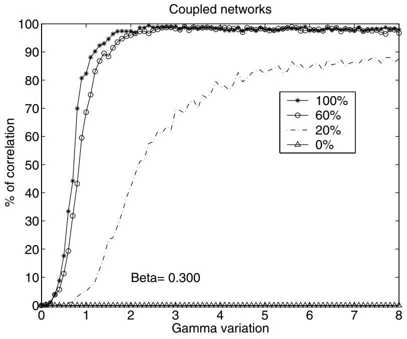


Fig. 2. Triplets obtained to 0%, 20%, 60 % and 100% of inter-group neurons connected

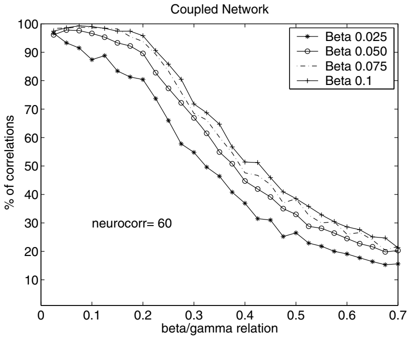


Fig. 3. Triplets obtained to β = 0.025, 0.05, 0.075 e 0.1 as a function of $\frac{\beta}{\gamma}$

can be seen in Fig. 2, which shows that even when only 20% of the inter-group neurons were connected, our model presented a recovery rate of global pattern close to 80%. However, when 60% of the inter-group neurons were connected the number of triplets obtained was close to 100%, in practice, the same result of a completely coupled network. We compared our results with the ones achieved in [4] and we observed that in [4], the recovery capacity of global patterns was close 90% for a completely coupled network, while in our model and for a typical value of β, namely $\beta = 0.3$, the recovery capacity was close to 100%, even with a network having only 60% of inter-group neurons connected.

We have, also, analyzed the influence of the number of correlation (triplets), for a wide range of the parameter β, as a function of $\frac{\beta}{\gamma}$ relation (Fig. 3). We observed that when β value increases it is necessary an increase of the γ value in such way to improve the recovery capacity. Furthermore, we could, also, infer that a typical value of $\frac{\beta}{\gamma}$ relation is 0.075, for an specific value of β, namely $\beta = 0.1$.

4 Conclusions

In this paper, we have presented a new proposal of construction of multi-level associative memories using GBSB neural networks that was inspired by TNGS [1]. We derived a new equation for the whole coupled system that extends previous models by means of a term that represents the inter-group connections.

We performed numerical computations of a two-level memory system and obtained a recovery rate of global patterns close to 100%, even when the networks are weakly coupled showing that it seems possible to build multi-level memories when new groups of ANNs are interconnected.

This present work is currently being generalized in order to include the effects due to different γ values (strength of the inter-groups synapses), such that the model would become more biologically plausible.

References

1. Edelman, G.M.: Neural darwinism: The theory of neuronal group selection. Basic Books, New York (1987)
2. Clancey, W.J.: Situated cognition : on human knowledge and computer representations. Learning in doing. Cambridge University Press, Cambridge, U.K. ; New York, NY, USA (1997)
3. Lillo, W.E., Miller, D.C., Hui, S., Zak, S.H.: Synthesis of brain-state-in-a-box (BSB) based associative memories. IEEE Transactions on Neural Network **5** (1994) 730–737
4. Sutton, J.P., Beis, J.S., Trainor, L.E.H.: A hierarchical model of neocortical synaptic organization. Mathl. Comput. Modeling **11** (1988) 346–350
5. Anderson, J.A., Silverstein, J.W., Ritz, S.A., Jones, R.S.: 22. In: Distinctive features, categorical perception, probability learning: some applications of a neural model. MIT Press, Cambridge, Massachusetts (1985) 283–325
6. Hui, S., Zak, S.H.: Dynamical analysis of the brain-state-in-a-box (BSB) neural models. IEEE Transactions on Neural Networks **3** (1992) 86–94
7. Golden, R.M.: The brain-state-in-a-box neural model is a gradient descent algorithm. Journal of Mathematical Psychology **30** (1986)
8. Cohen, M.A., Grossberg, S.: Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. IEEE Transactions on Systems, Man, and Cybernetics **13** (1983) 815–826
9. Gomes, R.M., Braga, A.P., Borges, H.E.: Energy analysis of hierarchically coupled generalized-brain-state-in-box GBSB neural network. In: Proceeding of V Encontro Nacional de Inteligência Artificial - ENIA 2005, São Leopoldo, Brazil, (to be published) (2005)

Balance Algorithm for Point-Feature Label Placement Problem

Zheng He[1] and Koichi Harada[2]

[1] Graduate School of Engineering, Hiroshima University, Japan
ethen@hiroshima-u.ac.jp
[2] Department of Integrated Arts & Sciences, Hiroshima University, Japan
harada@mis.hiroshima-u.ac.jp

Abstract. This paper proposes an improved discrete Hopfield Neural (HN) network, balance algorithm, to optimize the Point-Feature Labeling Placement (PFLP) problem. The balance algorithm attains the balance between penalty function and original objective function based on the principle of weight balance, and can converge to the solution with better stability. This improved algorithm also allows HN network to be competitive with other traditional algorithms such as Genetic Algorithm (GA) and Simulated Annealing (SA) algorithm in solving PFLP problem and other constrained problems.

1 Introduction

The general aim of constrained optimization is to transform the problem to be solved into some easier solvable sub-problems which then are used as the basis of iterative process. A typical characteristic of the large number of early methods was the translation of constrained problems into basic unconstrained problems, in which a penalty function was determined for those constraints near or beyond the constraint boundary. Thus the constrained problem could be solved with a sequence of parameterized and unconstrained optimizations which would converge to the constrained problem in the limit (of the sequence). However, if using this method to solve a problem, the weighting factors for penalty functions should be sufficiently large in order to ensure the method to converge to a feasible solution. As the influence of the penalty term's effect becomes stronger, the solutions are found to be much affected by the penalty terms than by the original objective; then the problem becomes ill-conditioned.

This paper proposes a new balance algorithm to find proper weighting factors for penalty functions. Then the Hopfield Neural (HN) network with a typical penalty, is improved by using this balance algorithm, and applied in solving Point-Feature Label Placement (PFLP). The results are compared with those obtained by conventional Hopfield Neural Network (CHN), Genetic Algorithm (GA) and Simulated Annealing (SA) algorithm.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 179 – 184, 2005.

2 Balance Algorithm

2.1 Balance Principle in Constraint Optimization

A general constraint optimization problem can be converted into a non-constraint problem by adding penalty function to original objective, as expressed by Eq.(1). The original objective function *f* and penalty function *g* should be balanced in order to get optimal and feasible solution.

$$F = \alpha \cdot f + \beta \cdot g \tag{1}$$

Therefore we assign two weighting factors *α* and *β* to each function, respectively to maintain the balance condition, which is quite similar to a commonly used balance in physical experiments. According to the weighting principle of balance, the weights are usually added onto the lighter side of the balance from large to small in sequence until the balance scales reach the critical position. In the proposed problem, we treat *α* and *β* as weights and try to obtain the balance between *f* and *g* by regulating the weights accordingly. Here we assume the scales reach balance if the optimization solution is feasible.

Because the described algorithm includes optimizing process for many times, a much faster optimization algorithm is necessary to implement since optimization is a time-consuming work. The Conventional Hopfield Neural (CHN) network proposed by the physicist John J. Hopfield in1982 [1], which was based on gradient descendent, will be used as the optimization algorithm considering its simple way and short calculation time.

However, the gradient descendent property of CHN network makes itself easily end up with infeasible solution or traps into local optimal solution [2].Therefore in this paper, the CHN will be adopted as the basic optimization algorithm to set up the balance algorithm, which is referred as Improved HN (IHN) network, to keep balance between original function and penalty function.

2.2 Balance Algorithm

The whole process of balance algorithm is shown by Fig.1. Small weights *α* and *β* are assigned to build CHN network at first. If such a network cannot find a valid solution, increase weight *β* of constraint function smoothly and rebuild the CHN network until valid solution is available. Then the number of times of obtaining optimal solution n is counted as the evaluation parameter, and if n is larger than the certain predefined value N_{max}, the calculation stops. The final valid solution is output as optimal. Otherwise,

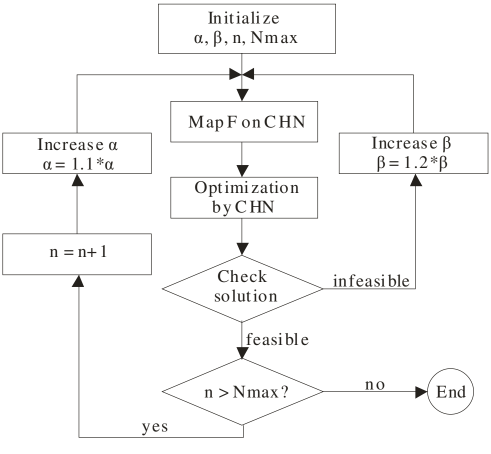


Fig. 1. Balance algorithm flow chart

increase the weight α of objective function and rebuild the CHN network to seek for a better solution.

3 Optimization Model of PFLP

PFLP is an optimization problem occurring frequently in production of informational graphics, though it arises most often in automated cartography [3]. Complexity analysis reveals that the basic PFLP problem and the most interesting variant are NP-hard [4].

The PFLP problem can be modeled into the combinatorial optimization stated as: a set of n points is given; each of the points must be labeled by assigning its label to one of m predefined positions. A complete label placement is represented by a vector $\vec{x}=(x_1, x_2,\ldots,x_{\mathrm{n}})$, and each component $x_i \in \{1,2,\ldots,m\}$ $(i=1,2,\ldots,n)$ identifies the assigned position of point i, as shown in Fig.2.

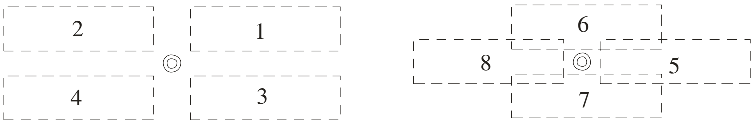


Fig. 2. Possible label positions and their desirability relative to a given point

Two objects are of particular importance [5]: Minimizing the degree to which labels overlap and obscure other features; maximizing the degree to which labels are unambiguously and clearly associated with the features they identify. We define *Conb*$_i$, *Conp* and *Conl*$_i$ to express the state for a specific label placement overlaps map boundary, other feature point and other label placement respectively; and use A, B and C as corresponding penalty. D is a position penalty related to desirability rank of each actual label position. The objective function for evaluating a label placement is given by Eq.(2) [6], [7].

$$f(\vec{x}) = \sum_{i=1}^{n}(A\cdot Conb_i+B\cdot Conp_i+C\cdot Conl_i) + D\cdot\sum_{i}^{n}\frac{x_i-1}{m} \tag{2}$$

In addition, the constraint conditions which limit each point to own and only own one label placement can be express by Eq.(3) [8]:

$$g(\vec{x}) = [\sum_{i=0}^{N-1}\sum_{m=0}^{N-1}\sum_{j=0}^{M-1}\sum_{n=0}^{M-1}\delta_{im}(1-\delta_{jn})x_i^j x_m^n - \sum_{i=0}^{N-1}\sum_{j=0}^{M-1}x_i^j + \sum_{i=0}^{N-1}1] \tag{3}$$

Substitute Eq.(2) and Eq.(3) into Eq.(1), optimization function can be obtained. Then comparing with the standard energy function of Hopfield neural network, refer to [1], PFLP can be mapped onto CHN and weight ω_{ijmn} and bias θ_{ij} is given by Eq.(4) accordingly [9]:

$$\omega_{ijmn} = -C\alpha Conl_{ij}^{mn} - \beta\delta_{im}(1-\delta_{jn}), \quad \delta_{mn}=\begin{cases}1 & m=n\\ 0 & m\neq n\end{cases}$$
$$\theta_{ij} = -\frac{\alpha}{2}(AConb_i^j + BConp_i^j + Dj) + \frac{\beta}{2} \tag{4}$$

4 Test Verifications

4.1 Setting for Test

Tests on the PFLP problem were carried out in order to verify the proposed IHN network in solving PFLP problem. The optimization model is described as the following: *N* point features with fixed-size labels (30×7 units) are randomly placed on a region with size of 792×612 units. Optimization is run for *N* = 100, 200… 1000. Labels were allowed to be placed in the eight positions (*M* = 8) around the point feature as Fig.2 shows.

In simulation, parameters in Eq.(2) and Eq.(3) are defined as: *A*=100, *B*=200, *C*=16, *D*=1. In GA and SA, Eq.(2) is used as objective function. When using IHN network to optimize, Eq.(3 is used as objective function, and parameters *α* and *β* in Eq.(4) should be dynamically adjusted to obtain valid solutions. Final solutions of all algorithms are converted into calculate result using Eq.(2).

SA using in test is based on section 3.6 of [6], all control parameters are same configured as this paper. The implementation of GA was carried out by a software package named Galib (version 2.4.5) (http://lancet.mit.edu/ga/), which was instantiated to implement a steady-state genetic algorithm, with 1% of the population replaced each generation. The genetic operator was an edge recombination crossover operator (partial match crossover was also tried, but performed poorly). The population size was specified as 100. The optimal solution was obtained when the difference between two best solutions from consecutive populations was smaller than a predefined error tolerance of 0.01.

4.2 Comparisons

First test is to compare the stability of CHN and IHN. The optimization is carried out for 100 times for a map consists of 100 point features. It is to be noted that although the CHN networks can be ensured to convergent to valid solutions by setting large values to the weight ratio of *β/α* in Eq.(3), the solution will become very bad since the constraint condition is too much emphasized, and the stability comparison would be of no significance. On the contrary, this ratio should not be too low for similar reason. The values of *α* and *β* can be determined through repeated tests, from which it is deduced that when *α*=1 and *β*=5, most of the produced solutions by CHN is valid and of good quality.

Table 1 compares the test results in stability and convergence time by IHN and CHN. It is can be seen that solutions of IHN are valid obviously, and they are more stable and better than CHN. But the calculation time of IHN is much longer than CHN because it will call CHN for many times during optimization.

Table 1. Comparison Stability and convergence time

	Valid times	Best result	covariance	Average solution	Average time
CHN	77%	334	14082.24	367	15 ms
IHN	100%	18	164	20	397 ms

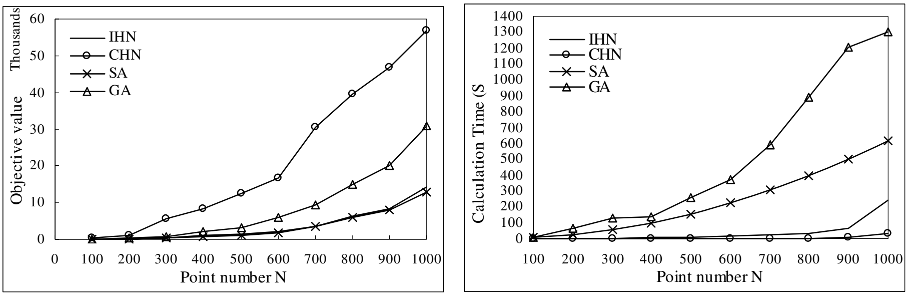
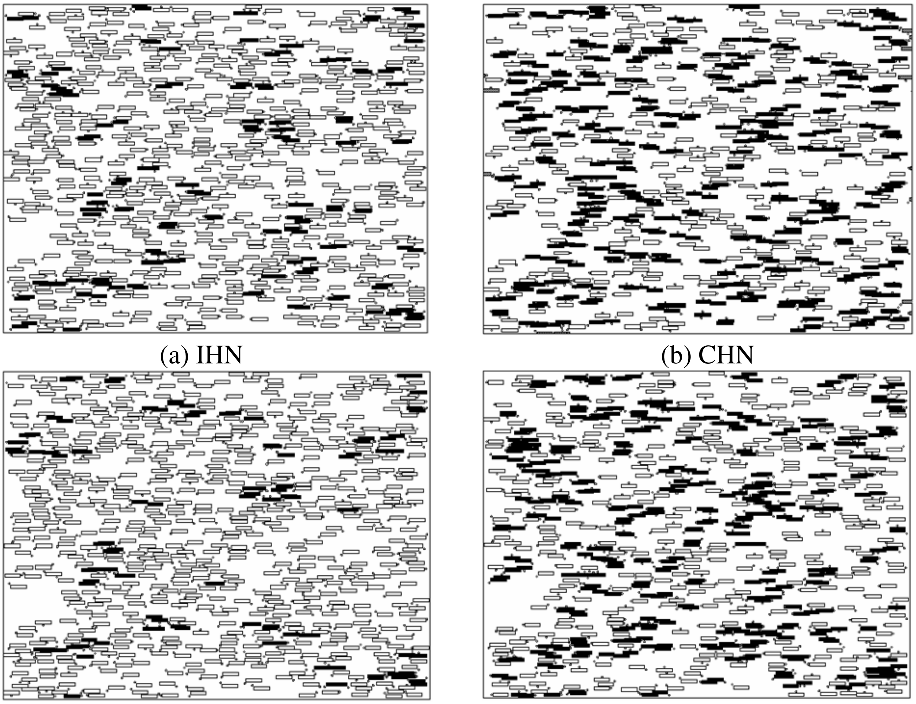


Fig. 3. Average solution and calculation time


(a) IHN

(b) CHN

(c) SA

(d) GA

Fig. 4. A sample map of 700 point-features with labels placed by four different algorithms

Furthermore, 10 maps which include different numbers of point-feature (N=100, 200, …, 1000) are randomly created. Optimization process is carried out 20 times for each map respectively, and the result is the average value of these 20 results. Fig 3 gives comparison the average result of objective value and calculation time. As the figures show, the IHN performed surprisingly well. The solution quality of IHN is much better than CHN and GA; and is as good as that of SA, and eventually better than SA. Moreover, its calculation time is shorter than that of GA and SA.

Four optimized maps contented 700 points as examples show the same conclusion. Fig.4 indicates the map with optimized label-placements obtained from each algorithm. Labels printed in solid marks overlap other labels, points or boundary of the map. Labels printed in open marks are free of overlaps.

5 Conclusion

This paper proposes an algorithm that applies balance principle on CHN networks to solve constraint optimization problems. On this basis, an IHN network was set up and applied to deal with a typical combinatorial optimization problem: PFLP. Result shows that IHN successfully overcome the disadvantages of CHN network such as: unstable, invalid or bad solution. In addition, this IHN network demonstrates surprisingly good performance in solution quality and convergence time compared with CHN network or traditional optimization algorithms such as GA and SA.

References

1. Hopfield JJ, Tank DW (1985) Neural Computation of Decisions in Optimization Problems. Biological Cybernetics, 52:141-152
2. Gee AH, Prager RW (1995) Limitations of neural networks for solving traveling salesman problems. IEEE Trans. Neural Networks, 6:280-282
3. Cook, A. C., Jones BC (1990) A Prolog rule-based system for cartographic name placement. Computer Graphics Forum, 9:109-126
4. Marks J, Shieber S (1991) The Computational Complexity of Cartographic Label Placement, Technical Report TR-05-91, Harvard University
5. Imhof, E.(1975) Positioning names on maps. The American Cartographer, 2(2):128-144
6. Christensen J, Marks J, Shieber S: An Empirical Study of Algorithms for Point-Feature Label Placement. ACM Transactionson Graphics, 14(3):203-232
7. Christensen J, Marks J, Shieber S (1994) Placing Text Labels on Maps and Diagrams. Graphic Gems IV pp 497-504
8. Zoraster S (1990) The Solution of Large 0-1 Integer Programming Problems Encountered in Automated Cartography. Operations Research, 38(5):752-759
9. Ramanujam J, Sadayappan P (1995) Mapping Combinatorial Optimization Problems onto Neural Networks. Information Sciences 82:239 -255

Models of Self-correlation Type Complex-Valued Associative Memories and Their Dynamics

Yasuaki Kuroe[1] and Yuriko Taniguchi[2]

[1] Center for Information Science
[2] Department of Electronics and Information Science,
Kyoto Institute of Technology,
Matsugasaki, Sakyo-ku, Kyoto 606-8585, Japan
kuroe@dj.kit.ac.jp

Abstract. Associative memories are one of the popular applications of neural networks and several studies on their extension to the complex domain have been done. One of the important factors to characterize behavior of a complex-valued neural network is its activation function which is a nonlinear complex function. We have already proposed a model of self-correlation type associative memories using complex-valued neural networks with one of the most commonly used activation function. In this paper, we propose two additional models using different nonlinear complex functions and investigated their behaviors as associative memories theoretically. Comparisons are also made among these three models in terms of dynamics and storage capabilities.

1 Introduction

In recent years, there have been increasing research interests of artificial neural networks and many efforts have been made on applications of neural networks to various fields. As applications of the neural networks spread more widely, developing neural network models which can directly deal with complex numbers is desired in various fields. Several models of complex-valued neural networks have been proposed and their abilities of information processing have been investigated.

One of the most useful and most investigated areas of applications of neural networks addresses implementations of associative memories. Among them associative memories of self-correlation type are easy to implement and have been extensively studied. Some models of complex-valued associative memories of self-correlation type have been proposed [1,2,3,4,5,6].

One of the important factors to characterize behavior of a complex-valued neural network is its activation function which is a nonlinear complex function. In the real-valued neural networks, the activation is usually chosen to be a smooth and bounded function such as a sigmoidal function. In the complex region, however, there are several possibilities in choosing an activation function because of a variety of complex functions. In [7] the properties that a suitable activation should possess are discussed for complex-valued backpropagation of complex-valued feedforward neural networks. [8] discusses the properties of activation functions from the standpoint of existence of an energy function for complex-valued recurrent neural networks.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 185–192, 2005.

The purpose of this paper is to present models of self-correlation type associative memories using complex-valued neural networks and to investigate their qualitative behaviors theoretically. We have already presented a model of complex-valued self-correlation type associative memories using one of the most commonly used complex functions in complex-valued neural networks [2,3]. In this paper we propose two additional models of complex-valued self-correlation type associative memories using different complex functions We treat the models of complex-valued associative memories as nonlinear dynamical systems and study their qualitative behaviors theoretically. In particular, we investigate the structures and asymptotic behaviors of solution orbits near each memory pattern. Comparisons are also made among these three models in terms of asymptotic behaviors of solution orbits near each memory pattern and storage capabilities.

In the following, the imaginary unit is denoted by i ($i^2 = -1$). The n-dimensional complex (real) space is denoted by $\boldsymbol{C}^n(\boldsymbol{R}^n)$ and the set of $n \times m$ complex (real) matrices is denoted by $\boldsymbol{C}^{n\times m}(\boldsymbol{R}^{n\times m})$. For $A \in \boldsymbol{C}^{n\times m}$ ($\boldsymbol{a} \in \boldsymbol{C}^n$), its real and imaginary parts are denoted by A^R ($\boldsymbol{a}^R$) and A^I ($\boldsymbol{a}^I$), respectively.

2 Complex-Valued Associative Memory of Self-correlation Type

2.1 Models

Let m be the number of memory patterns to be stored and each memory pattern be an N dimensional complex vector, denoted by $\boldsymbol{s}^{(\gamma)} \in \boldsymbol{C}^N$, $\gamma = 1, 2, \cdots, m$. Suppose that the set of memory patterns satisfies the following orthogonal relations.

$$\boldsymbol{s}^{(\gamma)*}\boldsymbol{s}^{(l)} = \begin{cases} N, \gamma = l \\ 0, \gamma \neq l \end{cases} \tag{1}$$

$$|s_j^{(\gamma)}| = 1, \quad j = 1, 2, \cdots, N \tag{2}$$

for all $\gamma, l = 1, 2, \cdots, m$ where $\boldsymbol{s}^*$ is the conjugate transpose of $\boldsymbol{s}$ and $s_j^{(\gamma)}$ is the jth element of $\boldsymbol{s}^{(\gamma)}$.

Consider a complex-valued neural network described by difference equations of the form:

$$x_j[t+1] = f(\sum_{k=1}^{N} w_{jk}x_k[t]), \qquad j = 1, 2, \cdots, N \tag{3}$$

where $x_j[t] \in \boldsymbol{C}$ is the output of the jth neuron at time t, $w_{jk} \in \boldsymbol{C}$ is the connection weight from the kth neuron to the jth neuron and $f(\cdot)$ is the activation function which is a nonlinear complex function ($f : \boldsymbol{C} \rightarrow \boldsymbol{C}$).

Let us determine the weight matrix $W = \{w_{jk}\} \in \boldsymbol{C}^{N\times N}$ and the activation function $f(\cdot)$ so that the neural network (3) can store the set of memory vectors and act as an associative memory.

The weight matrix W is determined after the model of conventional real-valued associative memories of self-correlation type as follows. Taking account of the orthogonal structure of the set of memory vectors $\{\boldsymbol{s}^{(\gamma)}\}$, we determine the weight matrix W by the sum of the autocorrelation matrix of each memory vector $\boldsymbol{s}^{(\gamma)}$:

$$W = \frac{1}{N} \sum_{\gamma=1}^{m} \boldsymbol{s}^{(\gamma)} \boldsymbol{s}^{(\gamma)*} \tag{4}$$

One of the important factors to characterize behavior of a complex-valued neural network is its activation function $f(\cdot)$. In the real-valued neural networks, the activation is usually chosen to be a smooth and bounded function such as sigmoidal functions. In the complex region, however, there are several possibilities in choosing an activation function because of a variety of complex functions. As a candidate of the activation function $f(\cdot)$, we will choose a complex function which satisfies the conditions:

(i) $f(\cdot)$ is a smooth and bounded function by analogy with the sigmoidal function of real-valued neural networks, and
(ii) each memory vector $\boldsymbol{s}^{(\gamma)}$ becomes an equilibrium point of the network (3).

The condition (ii) is accomplished if the following condition hold.

$$f(s_j^{(\gamma)}) = s_j^{(\gamma)}, \;\; j = 1, 2, \cdots, N, \;\; \gamma = 1, 2, \cdots, m. \tag{5}$$

In regard to the condition (i), we recall the Liouville's theorem, which says that 'if $f(u)$ is analytic at all $u \in \boldsymbol{C}$ and bounded, then $f(u)$ is a constant function'. Since a suitable $f(u)$ should be bounded, it follows from the theorem that if we choose an analytic function for $f(u)$, it is constant, which is clearly not suitable [7]. In place of analytic function we choose functions which have the continuous partial derivatives $\partial f^R/\partial u^R$, $\partial f^R/\partial u^I$, $\partial f^I/\partial u^R$ and $\partial f^I/\partial u^I$ where $f(u) = f^R(u^R, u^I) + i f^I(u^R, u^I)$.

We consider the following three complex functions as the activation function of (3).

- Model A: $$f(u) := \frac{\eta u^R}{\eta - 1 + \sqrt{2}|u^R|} + i \frac{\eta u^I}{\eta - 1 + \sqrt{2}|u^I|}, \tag{6}$$
- Model B: $$f(u) := \frac{\eta |u|}{\eta - 1 + |u|} \exp\left[i\{\arg u - \frac{1}{2^n} \sin(2^n \arg u)\}\right], \quad -\pi \leq \arg u < \pi \tag{7}$$
- Model C: $$f(u) := \frac{\eta u}{\eta - 1 + |u|} \tag{8}$$

where η is a real number satisfying $\eta - 1 > 0$. Note that, in the function (6) the real and imaginary parts of an input go through the nonlinear function separately, and in (7) the magnitude and the phase of an input go through the nonlinear function separately, and in (8) only the magnitude of an input go through the nonlinear function with the phase being unchanged. The functions (6) and (8) have been often used as activation functions in complex-valued neural networks. The function (7) is obtained by extending the activation function used in [6] so as to satisfy the condition (i), that is, it becomes a smooth function. The functions (6), (7) and (8) are all not analytic, but have the continuous partial derivatives $\partial f^R/\partial u^R$, $\partial f^R/\partial u^I$, $\partial f^I/\partial u^R$ and $\partial f^I/\partial u^I$ and are bounded. We call the complex-valued associative memory described by (3), (4) and (6) Model A, by (3), (4) and (7) Model B, and by (3), (4) and (8) Model C, respectively. Model C was already proposed and its dynamics was studied in [2].

2.2 Memory Patterns

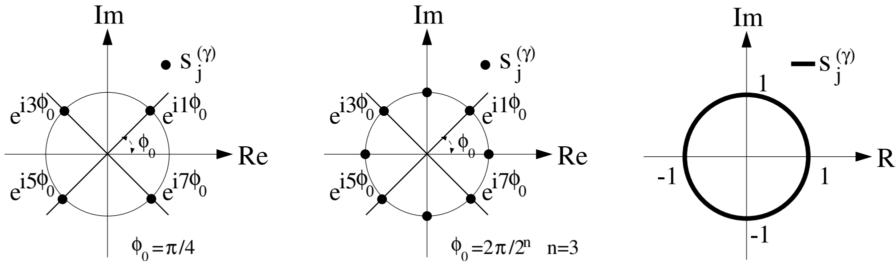
For Models A, B and C of complex-valued associative memories, each memory vector $\boldsymbol{s}^{(\gamma)}$ to be stored is demanded to becomes an equilibrium point of the network (3) and (4), which is accomplished if the condition (5) holds. This requirement makes each element of the memory vectors $\boldsymbol{s}^{(\gamma)}$, $(\gamma, l = 1, 2, \cdots, m)$ being only allowed to take restricted values satisfying $|s_j^{(\gamma)}| = 1$. In Model A, each element s_j^γ of the memory vectors $\boldsymbol{s}^{(\gamma)}$ is allowed to take the values

$$s_j^\gamma \in \{e^{i\pi/4}, e^{i3\pi/4}, e^{i5\pi/4}, e^{i7\pi/4}\}, \qquad j = 1, 2, \cdots, N \tag{9}$$

and, in Model B, each element s_j^γ of the memory vectors $\boldsymbol{s}^{(\gamma)}$ is allowed to take the values

$$s_j^\gamma \in \{e^{i0\pi/2^n}, e^{i\pi/2^n}, \ldots, e^{i(2^n-1)\pi/2^n}\}, \qquad j = 1, 2, \cdots, N. \tag{10}$$

In Model C each element s_j^γ of the memory vectors $\boldsymbol{s}^{(\gamma)}$ is able to take all the values satisfying $|s_j^{(\gamma)}| = 1$. Figure 1 shows the values which each element s_j^γ of the memory vectors is allow to take in Models A, B and C, respectively. Note that, in Model B, the number of the allowed values can be increased arbitrarily by increasing n.



(a) model A (b) Model B ($n = 3$) (c) Model C

Fig. 1. The values which each element s_j^γ of the memory vectors is allowed to take

By choosing such values for each element, each memory vector $\boldsymbol{s}^{(\gamma)}$ becomes an equilibrium point of the network (3) and (4). For Models A and B, it is easy to check that each memory vector is an isolated equilibrium point of the network. On the other hand, in Model C, each memory vector $\boldsymbol{s}^{(\gamma)}$ is not an isolated equilibrium point of the network as is discussed in [2], because the function (8) satisfies

$$e^{i\alpha} s_j^{(\gamma)} = f\left(\sum_{k=1}^{N} w_{jk} e^{i\alpha} s_k^{(\gamma)}\right), \qquad j = 1, 2, \cdots, N \tag{11}$$

for any real number α. This implies that $\boldsymbol{s}^{(\gamma)}$ is a point in the set of equilibrium points defined by

$$\varPhi^{(\gamma)} = \{e^{i\alpha}\boldsymbol{s}^{(\gamma)} : \forall\alpha \in \boldsymbol{R}\} \subset \boldsymbol{C}^N \tag{12}$$

which is a closed curve in the complex N dimensional state space $\boldsymbol{C}^N$. From this fact we identify all the points in the equilibrium set $\Phi^{(\gamma)}$ and regard $\Phi^{(\gamma)}$ as a memory pattern for Model C [2].

3 Qualitative Analysis of Behaviors Near Memory Patterns

In this section we will study qualitative behaviors of Models A, B and C of the complex-valued associative memories. Especially, we investigate asymptotic behaviors of solution orbits near each memory pattern $\boldsymbol{s}^{(\gamma)}$.

3.1 Linearized Models Near Memory Patterns

The qualitative behavior of a nonlinear dynamical system near an equilibrium point can be studied via linearization with respect to that point. We will derive the linearized model at each memory vector $\boldsymbol{s}^{(\gamma)}$. Note that the activation functions $f(u)$ in (6), (7) and (8) are not differentiable with respect to u for all $u \in \boldsymbol{C}$, but their real and imaginary parts, f^R and f^I, are continuously partially differentiable with respect to u^R and u^I. It is, therefore, possible to derive the linearized model by separating each model into its real and imaginary parts.

We linearize Models A and B at each memory vector $\boldsymbol{s}^{(\gamma)}$ and Model C at a point $\boldsymbol{q}^{(\gamma)} = e^{i\alpha}\boldsymbol{s}^{(\gamma)}$ of each equilibrium set $\Phi^{(\gamma)}$ where α is a real number. Let $\Delta x_i[t] := x_i[t] - s_i^{(\gamma)}$ and define $\boldsymbol{y}[t] \in \boldsymbol{R}^{2N}$ by

$$\boldsymbol{y}[t] = (\Delta x_1^R[t], \Delta x_2^R[t], \cdots, \Delta x_N^R[t], \Delta x_1^I[t], \Delta x_2^I[t], \cdots, \Delta x_N^I[t])^T \tag{13}$$

where $(\cdot)^T$ denotes the transpose of $(\cdot)$. The linearized model for Models A and B is obtained as

$$\boldsymbol{y}[t+1] = J(\boldsymbol{s}^{(\gamma)})\boldsymbol{y}[t], \quad \gamma = 1, 2, \cdots, m \tag{14}$$

where $J(\boldsymbol{s}^{(\gamma)}) = F(\boldsymbol{s}^{(\gamma)})Y \in \boldsymbol{R}^{2N\times 2N}$. The matrices $F(\boldsymbol{s}^{(\gamma)}) \in \boldsymbol{R}^{2N\times 2N}$ and $Y \in \boldsymbol{R}^{2N\times 2N}$ are given as follows.

$$F(\boldsymbol{s}^{(\gamma)}) = \begin{bmatrix} F_{RR} & F_{RI} \\ F_{IR} & F_{II} \end{bmatrix} \qquad Y = \begin{bmatrix} W^R & -W^I \\ W^I & W^R \end{bmatrix}$$

where

$$F_{RR} = \mathrm{diag}(\left.\frac{\partial f^R}{\partial z^R}\right|_{z=s_1^{(\gamma)}}, \left.\frac{\partial f^R}{\partial z^R}\right|_{z=s_2^{(\gamma)}}, \cdots, \left.\frac{\partial f^R}{\partial z^R}\right|_{z=s_N^{(\gamma)}})$$

$$F_{RI} = \mathrm{diag}(\left.\frac{\partial f^R}{\partial z^I}\right|_{z=s_1^{(\gamma)}}, \left.\frac{\partial f^R}{\partial z^I}\right|_{z=s_2^{(\gamma)}}, \cdots, \left.\frac{\partial f^R}{\partial z^I}\right|_{z=s_N^{(\gamma)}})$$

$$F_{IR} = \mathrm{diag}(\left.\frac{\partial f^I}{\partial z^R}\right|_{z=s_1^{(\gamma)}}, \left.\frac{\partial f^I}{\partial z^R}\right|_{z=s_2^{(\gamma)}}, \cdots, \left.\frac{\partial f^I}{\partial z^R}\right|_{z=s_N^{(\gamma)}})$$

$$F_{II} = \mathrm{diag}(\left.\frac{\partial f^I}{\partial z^I}\right|_{z=s_1^{(\gamma)}}, \left.\frac{\partial f^I}{\partial z^I}\right|_{z=s_2^{(\gamma)}}, \cdots, \left.\frac{\partial f^I}{\partial z^I}\right|_{z=s_N^{(\gamma)}})$$

and the elements of $F(\boldsymbol{s}^{(\gamma)})$ are given by

$$\frac{\partial f^R}{\partial u^R}\bigg|_{u=s_j^\gamma} = \frac{\partial f^I}{\partial u^I}\bigg|_{u=s_j^\gamma} = 1 - \frac{1}{\eta}, \qquad \frac{\partial f^R}{\partial u^I}\bigg|_{u=s_j^\gamma} = \frac{\partial f^I}{\partial u^R}\bigg|_{u=s_j^\gamma} = 0$$

for Model A and

$$\frac{\partial f^R}{\partial u^R}\bigg|_{u=s_j^\gamma} = \left(1 - \frac{1}{\eta}\right)(s_j^{\gamma R})^2, \qquad \frac{\partial f^I}{\partial u^I}\bigg|_{u=s_j^\gamma} = \left(1 - \frac{1}{\eta}\right)(s_j^{\gamma I})^2$$

$$\frac{\partial f^R}{\partial u^I}\bigg|_{u=s_j^\gamma} = \frac{\partial f^I}{\partial u^R}\bigg|_{u=s_j^\gamma} = \left(1 - \frac{1}{\eta}\right)s_j^{\gamma R} s_j^{\gamma I}$$

for Model B. The linearized model for Model C is obtained as the same form as (14) with $J(\boldsymbol{s}^{(\gamma)})$ being replaced by $J(\boldsymbol{q}^{(\gamma)})$, where the elements of $F(\boldsymbol{q}^{(\gamma)})$ are given by

$$\frac{\partial f^R}{\partial u^R}\bigg|_{u=q_j^\gamma} = 1 + \frac{-1 + (q_j^{\gamma I})^2}{\eta}, \qquad \frac{\partial f^I}{\partial u^I}\bigg|_{u=q_j^\gamma} = 1 + \frac{-1 + (q_j^{\gamma R})^2}{\eta}$$

$$\frac{\partial f^R}{\partial u^I}\bigg|_{u=q_j^\gamma} = \frac{\partial f^I}{\partial u^R}\bigg|_{u=q_j^\gamma} = \frac{-q_j^{\gamma R} q_j^{\gamma I}}{\eta}$$

3.2 Structure of Solution Orbits Near Memory Patterns

We now analyze the qualitative behavior and structure of the solution orbits near each memory pattern. This can be done by investigating the eigenvalues and eigenvectors of the coefficient matrix $J(\boldsymbol{s}^{(\gamma)})$ (or $J(\boldsymbol{q}^{(\gamma)})$) of the linearized model (14). Let $\lambda_i(J(\boldsymbol{s}^{(\gamma)}))$ be the ith eigenvalue of $J(\boldsymbol{s}^{(\gamma)})$ The following theorems are obtained for Models A, B and C.

Theorem A.1. *All the eigenvalues of the coefficient matrix $\boldsymbol{J}(\boldsymbol{s}^\gamma)$ of Model A are real and satisfy the following condition:*

$$|\lambda_j(\boldsymbol{J}(\boldsymbol{s}^\gamma))| \leq (\eta - 1)/\eta < 1, \quad j = 1, 2, \cdots, 2N. \tag{15}$$

Theorem A.2. *The matrix $\boldsymbol{J}(\boldsymbol{s}^\gamma)$ of Model A has $2m$ eigenvalues $(\eta - 1)/\eta$ and $2(N - m)$ eigenvalues 0. Two of the corresponding eigenvectors to the eigenvalue $(\eta - 1)/\eta$ are*

$$\boldsymbol{r}^{(\gamma)} := \left((\boldsymbol{s}^{(\gamma)R})^T, (\boldsymbol{s}^{(\gamma)I})^T\right)^T \tag{16}$$

$$\boldsymbol{p}^{(\gamma)} := \left((\{i\boldsymbol{s}^{(\gamma)}\}^R)^T, (\{i\boldsymbol{s}^{(\gamma)}\}^I)^T\right)^T. \tag{17}$$

Theorem B.1. *All the eigenvalues of the coefficient matrix $\boldsymbol{J}(\boldsymbol{s}^\gamma)$ of Model B satisfy the following condition:*

$$|\lambda_j(\boldsymbol{J}(\boldsymbol{s}^\gamma))| \leq 1 - 1/\eta < 1, \quad j = 1, 2, \cdots, 2N \tag{18}$$

Theorem B.2. *The matrix $\boldsymbol{J}(\boldsymbol{s}^{\gamma})$ of Model B has at least one eigenvalues $(\eta-1)/\eta$ and at least one eigenvalue 0. The corresponding eigenvector to the eigenvalue $(\eta-1)/\eta$ is $\boldsymbol{r}^{(\gamma)}$ and the corresponding eigenvector to the eigenvalue 0 is $\boldsymbol{p}^{(\gamma)}$.*

Theorem C.1. [2] *All the eigenvalues of the coefficient matrix $\boldsymbol{J}(\boldsymbol{s}^{\gamma})$ of Model C are real and satisfy the following condition.*

$$|\lambda_j(\boldsymbol{J}(\boldsymbol{q}^{\gamma}))| \leq 1, \quad j = 1, 2, \ldots, 2N \tag{19}$$

Theorem C.2. [2] *The matrix $J(\boldsymbol{q}^{(\gamma)})$ has at least one eigenvalue 1 and at least one eigenvalue $(\eta-1)/\eta$. It also has $2(N-m)$ eigenvalues 0. The corresponding eigenvector to the eigenvalue 1 is*

$$\boldsymbol{p}^{(\gamma)} = ((\{i\boldsymbol{q}^{(\gamma)}\}^R)^T, (\{i\boldsymbol{q}^{(\gamma)}\}^I)^T)^T \tag{20}$$

and that to the eigenvalue $(\eta-1)/\eta$ is

$$\boldsymbol{r}^{(\gamma)} = ((\boldsymbol{q}^{(\gamma)R})^T, (\boldsymbol{q}^{(\gamma)I})^T)^T. \tag{21}$$

Note that Theorems A.1 to C.2 hold for all the memory vectors $\boldsymbol{s}^{(\gamma)}$ $(\boldsymbol{q}^{(\gamma)})$, $\gamma = 1, 2, \cdots, m$. The proofs of Theorems A.1, A.2, B.1 and B.2 are omitted due to the space limitation.

3.3 Discussions

It is known that qualitative behavior of solutions near an equilibrium point of a nonlinear dynamical system is determined by its linearized model at the point if it is hyperbolic (in a discrete time system, the coefficient matrix of the linearized model has no eigenvalues of unit modulus) [9]. From the theorems obtained in the previous subsection, Models A and B are both hyperbolic, on the other hand, Model C is not hyperbolic.

For each model, it is required that all embedded memory vectors are at least asymptotically stable equilibrium points of the network for correct recalling as associative memories. From Theorems A.1 and B.1, each memory vector $\boldsymbol{s}^{(\gamma)}$ of Model A and B is an asymptotically stable equilibrium point because all the eigenvalues of the coefficient matrix $\boldsymbol{J}(\boldsymbol{s}^{\gamma})$ are less than one. Therefore every solution starting in the neighborhood of the vector $\boldsymbol{s}^{(\gamma)}$ tends to $\boldsymbol{s}^{(\gamma)}$ as $t \rightarrow \infty$. For model C, [2] investigates the structure of solution orbits near memory patterns $\Phi^{(\gamma)}$ based on the center manifold theory [9]. It is shown that, if $J(\boldsymbol{q}^{(\gamma)})$ has only one eigenvalue 1, each memory pattern $\Phi^{(\gamma)}$ is asymptotically stable, that is, every solution starting in the neighborhood of $\Phi^{(\gamma)}$ tend to $\Phi^{(\gamma)}$ as $t \rightarrow \infty$. Therefore it is concluded that in Model A and B all embedded memory vectors $\boldsymbol{s}^{(\gamma)}$, $\gamma = 1, 2, \cdots, m$ are asymptotically stable and they all could be recalled. On the other hand, in Model C, all the embedded memory vectors are not necessarily asymptotically stable and it is necessary to evaluate eigenvalues of the coefficient matrices $\boldsymbol{J}(\boldsymbol{q}^{\gamma})$ to check their stability.

As stated in Section 2, a set of the vectors to be stored $\boldsymbol{s}^{(\gamma)} \in \boldsymbol{C}^N$, $\gamma = 1, 2, \cdots, m$ are specified so as to satisfy the conditions (1) and (2), and in Models A and B each element of the memory vectors $\boldsymbol{s}^{(\gamma)}$ is only allowed to take restricted values in (9) for

Model A, and in (10) for Model B. There is a difference among Models A, B and C in choosing such a set of vectors. A question arises: how many vectors can we choose such vectors in N dimensional space for Model A, B and C ? For Model C, it can be shown that, for any N there always exist N vectors satisfying the conditions (1) and (2) in the N dimensional space. On the other hand, there not always exit N vectors satisfying the conditions (1) and (9) for Model A, and the conditions (1) and (10) for Model B in the N dimensional space. The problems of how many vector can we choose such vectors for Models A and B are equivalent to the existence problems of the complex Hadamard matrices and the generalized Hadamard matrices, respectively, which are partially solved but not completely solved.

4 Conclusions

In this paper we presented models of self-correlation type associative memories using complex-valued neural networks and studied their qualitative behaviors theoretically. One of the important factors to characterize behavior of a complex-valued neural network is its activation function. We presented three kinds of models with different activation functions and investigate the structures and asymptotic behaviors of solution orbits near each memory pattern. We compared three models in terms of asymptotic behaviors of solution orbits near each memory pattern and storage capabilities.

References

1. A. Hirose, “Dynamics of fully complex-valued neural networks,” Electronics Letters, vol.28, no.16, pp.1492-1494, 1992.
2. Y. Kuroe, N. Hashimoto and T. Mori “ Qualitative analysis of a self-correlation type complex-valued associative memories,” *Nonlinear Analysis*, vol.47, pp.5795-5806.
3. Y. Kuroe, N. Hashimoto and T. Mori, “ Qualitative Analysis of Continuous Complex-Valued Associative Memories,” *Artificial Neural Networks - ICANN 2001, George Dorffner et. al. (Eds.), Lecture Notes in Computer Science*, 2130, pp.843 -850, Springer.
4. I. Nemoto and T. Kono, “Complex-valued neural network,” The Trans. of IEICE, vol. J74-D-II, no. 9, pp. 1282-1288, 1991 (in Japanese).
5. A.J.Noest, “Phaser neural network,” Neural Informaion Processing Systems, D.Z.Anderson,ed., pp. 584–591, AIP, New York, 1988.
6. S.Jankowski, A.Lozowski, and J.M.Zurada, “Complex-valued multistate neural associative memory,” IEEE Trans. Neural Networks, vol. 7, no. 6, pp. 1491–1496, Nov. 1996.
7. G. M. Georgiou and C. Koutsougeras, “Complex Domain Backpropagation”, *IEEE Transactions on Circuits and Systems-II*, Vol. 39, No. 5, pp. 330–334, 1992.
8. Y. Kuroe, M. Yoshida and T. Mori: On Activation Functions for Complex-Valued Neural Networks - Existence of Energy Functions -; Artificial Neural Networks and Neural Information Processing - ICANN/ICONIP 2003, Okyay Kaynak et. al.(Eds.), Lecture Notes in Computer Science, 2714, pp.985-992, Springer, 2003
9. S. Sastry, Nonlinear Systems Analysis, Stability, and Control, Springer-Verlag, 1999.

Recovery of Performance in a Partially Connected Associative Memory Network Through Coding

Kit Longden

Institute for Adaptive and Neural Computation, School of Informatics,
University of Edinburgh, 5 Forrest Hill, Edinburgh, EH1 2QL, Scotland, U.K
kit@anc.ed.ac.uk

Abstract. Introducing partial connectivity to an associative memory network increases the variance of the dendritic sum distributions, reducing the performance. A coding scheme to compensate for this effect is considered, in which output patterns are self-organised by the network. It is shown using signal-to-noise ratio analysis that when the output patterns are self-organised the performance is greater than in a network with a higher connectivity and random patterns, in the regime of low connectivity and a high memory load. This analysis is supported by simulations. The self-organising network also outperforms the random network with input activity-dependent thresholding mechanisms in simulations.

1 Introduction

Low levels of connectivity are typical in brain structures that can be described as feedforward associative memory networks. When random partial connectivity is introduced into a binary associative memory network with a constant level of activity in the input layer [1], the number of active inputs to the output cells moves from a constant value to a binomial distribution. Variations in this input activity to each cell result in an increase in the variance of the dendritic sum distribution. The capacity of the network is then significantly lower than is otherwise predicted when this effect is not taken into account [2].

Marr [3] proposed a thresholding mechanism to improve the performance of such a network. The dendritic sum is divided by a cell specific inhibitory term proportional to the cell's input activity, reducing input activity-dependent variations in the dendritic sum. Subsequent related thresholding schemes have been developed to improve the performance of partially connected associative memory networks [4, 5]. None have satisfactorily explained how the input activity to each cell can be measured in a biological network during recall. It is also difficult to see how sufficiently accurate inhibitory thresholds can be set for each cell given the low proportion of neurons that are inhibitory interneurons, roughly 10% in the hippocampus [6] for example.

Marr [3] also proposed and implemented another solution to this problem: an algorithm for creating output representations. During the storage phase, the network self-organises the output pattern by choosing to be active the output cells

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 193–198, 2005.

with the highest number of active inputs. Marr asserts that his algorithm chooses the ‘best suited’ cells for representing the output. This paper provides support for this claim, through signal-to-noise ratio (SNR) analysis and simulations.

2 Methods

Analysis. The network consists of N_2 binary output units each connected to a fraction Z of N_1 input units. The synapses are binary and modified using an all-or-nothing Hebbian rule [1]. The dendritic sum of cell i, d_i and input activity, a_i, are given by

$$d_i = \textstyle\sum_j w_{ij} c_{ij} s_j \quad \text{and} \quad a_i = \textstyle\sum_j c_{ij} s_j$$

where c_{ij} denotes a synapse from cell j, w_{ij} is the synaptic weight and s_j is the state of cell j ($c_{ij}, w_{ij}, s_j \in \{0, 1\}$). In each pattern pair stored in the network, there are M_1 units active in the input layer, and M_2 units active in the output layer. In the input activity-dependent thresholding scheme proposed by [3] considered later and used by [4, 5], neuron i is active only if

$$d_i/a_i \geq f \quad \text{and} \quad d_i \geq t$$

where t is the subtractive threshold, and f is an additional divisive threshold.

In the algorithm proposed by Marr [3], the input pattern is associated with the M_2 output units with the highest input activity. Superscripts are used to refer to the network when the output patterns are chosen in this way: in the random network the dendritic sum signal is denoted by d_s^{rnd}, and the dendritic sum noise by d_n^{so} in the self-organised network. Buckingham [2] established the following accurate approximations for the dendritic sum signal and noise distributions:

$$P(d_s^{rnd} = x | a, r) = \mathcal{B}(x; M_1, Z^{rnd}) \tag{1}$$

$$P(d_s^{so} = x | a, r) = \begin{cases} \frac{1}{\alpha_2} \mathcal{B}(x; M_1, Z^{so}) & \text{if } a \geq T \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

$$P(d_n^{rnd} = x | a, r) = \mathcal{B}(x; a, \rho^{rnd}(r)) \tag{3}$$

$$P(d_n^{so} = x | a, r) = \begin{cases} \frac{1}{1-\alpha_2} \mathcal{B}(x; a, \rho^{so}(r)) & \text{if } a < T \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where r is the number of times a neuron is active in a pattern, α_2 is the activity level in the second layer(M_2/N_2), T the input activity threshold used in self-organising the patterns, $\rho(r)$ the probability that a synapse has been modified in the storage of r patterns, and $\mathcal{B}$ denotes a binomial distribution. In the pattern self-organisation algorithm, the neurons with the highest connectivity are chosen to be associated with the input pattern. This is incorporated in the analysis by assuming the signal neurons have a higher effective connectivity, Z_s^{so}. The neurons not chosen have a lower average connectivity, Z_n^{so}, such that

$$Z^{so} = \alpha_2 Z_s^{so} + (1 - \alpha_2) Z_n^{so}. \tag{5}$$

The modification of their synapses results in a higher proportion of modified synapses, $\rho(r)$:

$$\rho^{rnd}(r) = 1 - (1 - \alpha_1)^r \quad \text{and} \quad \rho^{so}(r) = 1 - (1 - \tfrac{Z_s}{Z}\alpha_1)^r$$

where α_1 is the activity level in the first layer (M_1/N_1).

Simulations. In the simulations, the parameters are as follows: $N_1 = N_2 = 4000$, $\alpha_1 = \alpha_2 = 0.03$, and 1000 patterns pairs are stored.

3 Analysis

We consider two networks, one with random and one with self-organised output patterns. The parameters are identical except that the connectivity in the random network is lower, so that $Z_s^{so} = Z^{rnd}$. For a noiseless, full-sized recall cue, the means of the dendritic sum distributions are well approximated by

$$\mu_s^{rnd} = M_1 Z^{rnd} \tag{6}$$

$$\mu_s^{so} = M_1 Z_s^{so} \tag{7}$$

$$\mu_n^{rnd}(r) = M_1 Z^{rnd} \rho^{rnd}(r) \tag{8}$$

$$\mu_n^{so}(r) = M_1 Z_n^{so} \rho^{so}(r). \tag{9}$$

Since r is distributed identically for both the networks, $\mu_n^{rnd} > \mu_n^{so}$. The relative magnitude of μ_n^{so} and μ_n^{rnd} is not obvious since $Z_n^{so} < Z^{rnd}$, but $\rho^{so}(r) > \rho^{rnd}(r)$. By induction, it will be shown that $\mu_n^{so} < \mu_n^{rnd}$. When $r = 1$,

$$\mu_n^{rnd}(1) = M_1 Z^{rnd} \alpha_1 \quad \text{and} \quad \mu_n^{so}(1) = M_1 \tfrac{Z_n^{so} Z_s^{so}}{Z^{so}} \alpha_1.$$

By design, $Z_s^{so} = Z^{rnd}$, and also $Z_n^{so} < Z^{so}$ so $\mu_n^{so}(1) < \mu_n^{rnd}(1)$. For any given r,

$$\mu_n^{rnd}(r) = \mu_n^{rnd}(r-1)(1 - \alpha_1) + M_1 Z^{rnd} \alpha_1 \tag{10}$$

$$\mu_n^{so}(r) = \mu_n^{so}(r-1)(1 - \frac{Z_s^{so}}{Z^{so}}\alpha_1) + M_1 \frac{Z_n^{so} Z_s^{so}}{Z^{so}} \alpha_1. \tag{11}$$

Now $Z_s^{so} > Z^{so}$, so $(1 - \frac{Z_s^{so}}{Z^{so}}\alpha_1) < (1 - \alpha_1)$. As was the case for calculating the inequality of $\mu_n(1)$, $Z_s^{so} = Z^{rnd}$ and $\frac{Z_n^{so}}{Z^{so}} < 1$, so $M_1 Z^{rnd} \alpha_1 > M_1 \frac{Z_n^{so} Z_s^{so}}{Z^{so}} \alpha_1$. Therefore if $\mu_n^{rnd}(r-1) > \mu_n^{so}(r-1)$, then $\mu_n^{rnd}(r) > \mu_n^{so}(r)$. Since $\mu_n^{rnd}(1) > \mu_n^{so}(1)$, by induction $\mu_n^{rnd}(r) > \mu_n^{so}(r)$.

The variance of the dendritic sum signal in the self-organised network is intuitively small, as the variance of the upper tail of the input activity distribution must be less than across the whole distribution (figure 1a). The dendritic sum signal variances are

$$(\sigma_s^{rnd})^2 = M_1 Z^{rnd}(1 - Z^{rnd}) \tag{12}$$

$$(\sigma_s^{so})^2 = \frac{1}{\alpha_2} \sum_{a=T} \mathcal{B}(a; M_1, Z^{so})(a - M_1 Z_s^{so})^2. \tag{13}$$

We next consider the parameter ranges of $\rho(r)$ and Z that constrain the relative magnitudes of σ_n^{rnd} and σ_n^{so}. The variance of the self-organised dendritic sum noise is well approximated by $a\,\rho^{so}(r)(1-\rho^{so}(r))$, since $\alpha_2 \ll 1$. The random dendritic sum noise variance is $a\,\rho^{rnd}(r)(1-\rho^{rnd}(r))$. When $\rho^{rnd}(r) \geq 0.5$, $\rho^{rnd}(r) > \rho^{so}(r)$. Thus $\sigma_n^{rnd}(r|\,a) > \sigma_n^{so}(r|\,a)$ when $\rho^{rnd}(r) \geq 0.5$.

The input activity variances are $\sigma_a^{so}{=}M_1 Z_n^{so}(1{-}Z_n^{so})$ and $\sigma_a^{rnd} = M_1 Z^{rnd}(1-Z^{rnd})$. Now $Z_n^{so} < Z^{rnd}$, when $Z^{rnd} \leq 0.5$. Thus $\mu_a^{rnd} > \mu_a^{so}$ and $\sigma_a^{rnd} > \sigma_a^{so}$ when $Z^{rnd} \leq 0.5$. If σ_a is increased, it can only increase $\sigma_n(r|\,a)$:

$$\sigma_n^2(r) = \int\!\!\int\!\!\int ((d_n^2 - E_{a,\rho}^2[d_n] + E_{a,\rho}^2[d_n])P(d_n|a,\rho)P(a)P(\rho)da\,d\rho\,d(d_n)$$

$$- \left[\int\!\!\int\!\!\int d_n\,P(d_n|a,\rho)P(a)P(\rho)da\,d\rho\,d(d_n)\right]^2 \tag{14}$$

$$= \sigma_\rho^2(\mu_a^2 - \mu_a) + \mu_a(\mu_\rho - \mu_\rho^2) + \sigma_a^2(\sigma_\rho^2 + \mu_\rho^2) \tag{15}$$

where $E_{a,\rho}^2[d_n]$ is the expected value of the dendritic sum noise, d_n, conditioned on a and ρ. From equation 15, it is clear that σ_n must increase when σ_a is increased. In addition, increasing μ_a must also increase σ_n, because $\mu_a > 1$. Therefore $\sigma_n^{rnd}(r) > \sigma_n^{so}(r)$ when $\rho^{rnd}(r) \geq 0.5$ and $Z^{rnd} \leq 0.5$.

Putting all the inequalities together, $\mu_s^{rnd} = \mu_s^{so}$, $\mu_n^{rnd} > \mu_n^{so}$, $\sigma_s^{rnd} > \sigma_s^{so}$ and $\sigma_n^{rnd}(r) > \sigma_n^{so}(r)$ if $Z^{rnd} \leq 0.5$ and $\rho^{rnd}(r) \geq 0.5$. Therefore, $SNR^{so} > SNR^{rnd}$ if $Z^{rnd} \leq 0.5$ and $\rho^{rnd}(r) \geq 0.5$.

It should be noted that the dispersion has not been included in the calculation of the SNR [7]. Due to the large number of output neurons used during simulations, the correlations in signal output activity between patterns are very small and have been neglected. It should also be noted that the inequalities are derived from accurate but ultimately approximate expressions. For instance, the connectivities of the signal and noise units are treated as averages rather than distributions.

4 Simulations

From the analysis, the performance of a self-organising network is better than an equivalent random network with a greater connectivity of $Z^{rnd} = Z_s^{so}$ for high loads $\rho^{rnd}(r) \geq 0.5$ and low connectivity $Z^{rnd} \leq 0.5$ when dendritic sum thresholding is used. This result is supported for one parameter set with $\rho^{rnd} = 0.60$ in figure 1b. For low Z, the SNR is higher as predicted. As Z increases, a performance advantage remains until as $Z \rightarrow 1$, the random and self-organising network SNRs tend to the same values. The value of ρ^{rnd} can be decreased by decreasing the activity levels or the number of patterns stored. In simulations, these changes increased the SNR for both networks, but did not significantly change their qualitative relationship (data not shown).

Implementing input activity-dependent thresholding greatly improves the SNR of the random network, as expected (figure 1b). It also decreases the performance of the self-organised network: the reduction in the difference between

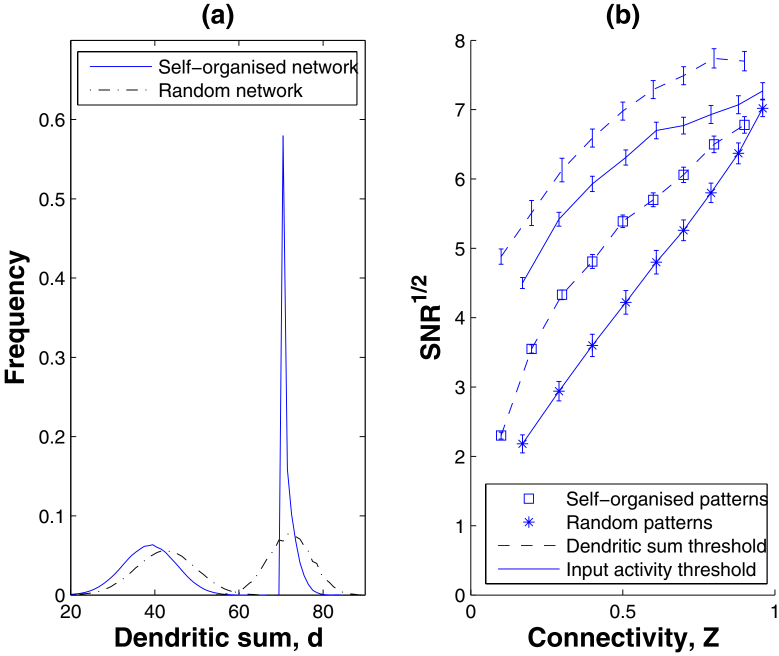


Fig. 1. **(a)** Dendritic sum distributions from a simulation, when $Z_s^{so} = Z^{rnd}$, requiring $Z^{so} = 0.50$ and $Z^{rnd} = 0.61$. **(b)** Network performances as a function of Z, with and without input activity-dependent thresholding, using noiseless recall cues. All data points are means and error bars are the mean standard deviation over 100 trials.

the means is greater than the reduction in the dendritic sum noise variance for this parameter set. The SNR of the random network with input activity thresholding is lower over the full range of connectivity values. In simulations, this performance advantage is maintained for lower values of ρ^{rnd} and for noisy recall cues (data not shown).

5 Discussion

The results demonstrate that coding schemes provide a novel way to recover performance when random partial connectivity introduced. When the output patterns are self-organised according to the algorithm proposed by Marr [3], the signal variance is extremely low, as the active units are taken from the upper tail of the connectivity distribution. The higher effective connectivity of the chosen output neurons, and the correspondingly lower average connectivity of the other neurons, ensures that the mean signal and the mean noise components of the dendritic sum are further separated than in the random network.

Using input activity-thresholding in the self-organised network reduces the noise variance, but also eliminates this increased separation of the means. Within the parameter ranges explored in simulations, the self-organised network using just dendritic sum thresholding outperformed the random network using input activity thresholding, as judged by the signal-to-noise ratio.

How output representations are formed is a relevant problem for biological examples of feedforward associative memory networks. The projection from CA3 to CA1 in the hippocampus has been argued to be an associative memory network [8]. How patterns of activity are formed in CA1 of the intact hippocampus remain unknown [9]. The implementation of the algorithm for self-organising output patterns [3] remains an open question.

References

[1] Willshaw, D.J., Buneman, O.P., Longuet-Higgins, H.C.: Non-holographic associative memory. Nature 222:197 (1969) 960–962

[2] Buckingham, J.: Delicate nets, faint recollections: a study of partially connected associative network memories. PhD thesis, Univ. Edinburgh (1991)

[3] Marr, D.: Simple memory: a theory for archicortex. Phil. Trans. R. Soc. B. 262:841 (1971) 24–81

[4] Buckingham, J., Willshaw, D.J.: On setting unit thresholds in an incompletely connected associative net. Network 4 (1993) 441–459

[5] Graham, B., Willshaw, D.J.: Improving recall from an associative memory. Biol. Cybernetics 72 (1995) 337–346

[6] Freund, T.F., Buzsaki, G.: Interneurons of the Hippocampus. Hippocampus 6 (1996) 347–470

[7] Dayan, P., Willshaw, D.J.: Optimizing synaptic learning rules in linear associative memories. Biol. Cybernet. 65:4 (1991) 253-265

[8] Treves, A.: Quantitative estimate of the information relayed by the Schaffer collaterals. J. Comp. Neuro. 2:3 (1995) 259–72

[9] Brun, V.H., Otnass, M.K., Molden, S., Steffenach, H.A., Witter, M.P., Moser, M.B., Moser, E.I.: Place cells and place recognition maintained by direct entorhinal-hippocampal circuitry. Science 296:5576 (2002) 2243–6

Optimal Triangle Stripifications as Minimum Energy States in Hopfield Nets

Jiří Šíma[*]

Institute of Computer Science, Academy of Sciences of the Czech Republic,
P. O. Box 5, 18207 Prague 8, Czech Republic
sima@cs.cas.cz

Abstract. The important task of generating the minimum number of sequential triangle strips for a given triangulated surface model is motived by applications in computer graphics. This hard combinatorial optimization problem is reduced to the minimum energy problem in Hopfield nets by a linear-size construction. First practical experiments have confirmed that computing the semi-optimal stripifications by using Hopfield nets is a promising approach. In this work we provide a theoretical justification of this method by proving that the classes of equivalent optimal stripifications are mapped one to one to the minimum energy states.

1 Sequential Triangular Strips and Hopfield Nets

Piecewise-linear surfaces defined by sets of triangles (triangulation) are widely used representations for geometric models. Computing a succinct encoding of a triangulated surface model represents an important problem in graphics and visualization. Current 3D graphics rendering hardware often faces a memory bus bandwidth bottleneck in the processor-to-graphics pipeline. Apart from reducing the number of triangles that must be transmitted it is also important to encode the triangulated surface efficiently. A common encoding scheme is based on sequential triangle strips which avoid repeating the vertex coordinates of shared triangle edges. Triangle strips are supported by several graphics libraries (e.g. IGL, PHIGS, Inventor, OpenGL). A *sequential triangle strip* (*tristrip*) of length $m-2$ is an ordered sequence of $m \geq 3$ vertices $\sigma = (v_1, \ldots, v_m)$ which encodes the set of $n(\sigma) = m-2$ different triangles $T_\sigma = \{\{v_p, v_{p+1}, v_{p+2}\}\,;\, 1 \leq p \leq m-2\}$ so that their shared edges follow alternating left and right turns as indicated in Figure 1.a (*dashed line*). Thus a triangulated surface model T with n triangles that is decomposed into k tristrips $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$ requires only $n + 2k$ (rather than $3n$) vertices to be transmitted. A crucial problem is to decompose a triangulation into the fewest tristrips which is NP-complete [2].

A new method of generating tristrips Σ for a given triangulation T with n triangles has recently been proposed [7] which is based on a linear-time reduction to the minimum energy problem in Hopfield network $\mathcal{H}_T$ having $O(n)$ units and connections. First practical experiments [5,7] have confirmed that $\mathcal{H}_T$ powered

[*] Research partially supported by the "Information Society" project 1ET100300517 and the Institutional Research Plan AV0Z10300504.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 199–204, 2005.

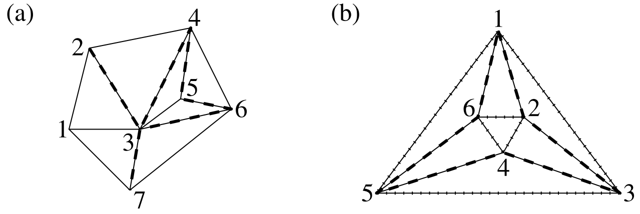


Fig. 1. (a) Tristrip (1,2,3,4,5,6,3,7,1) (b) Sequential cycle (1,2,3,4,5,6,1,2)

by simulated annealing (i.e. Boltzmann machine) can be used for computing the semi-optimal stripifications offline. For example, for "grid" models composed of 2312 triangles this method produced 297 tristrips on the average within 1–2 minutes as compared to 373 tristrips generated on-line by a leading stripification program FTSG [7]. In this paper the correctness of this reduction is formally shown providing a theoretical justification of the method. We prove a one-to-one correspondence between the classes of equivalent optimal stripifications of T and the minimum energy states reached by $\mathcal{H}_T$ during any sequential computation starting at the zero state (or $\mathcal{H}_T$ can be initialized arbitrarily if one asymmetric weight is introduced [7]). This provides another NP-completeness proof for the minimum energy problem in Hopfield nets [1,8]. A possible direction of further research is a generalization of the method for tristrips with zero-area triangles [2].

Hopfield networks [3] represent a very influential associative memory model which is connected to the much-studied Ising spin glass model in statistical physics [1]. Part of the appeal of Hopfield nets also stems from their natural hardware implementations using electrical networks or optical computers, and their application in combinatorial optimization [4]. A Hopfield network is composed of s *units* (*neurons*), indexed as $1, \ldots, s$, that are connected into undirected graph, in which each connection between unit i and j is labeled with an integer *symmetric weight* $w(i,j) = w(j,i)$. The absence of a connection indicates a zero weight, and vice versa. Further assume $w(j,j) = 0$ for $j = 1, \ldots, s$. The network *state* $\mathbf{y}^{(t)} = (y_1^{(t)}, \ldots, y_s^{(t)}) \in \{0,1\}^s$ is updated at time instants $t = 0, 1, \ldots$. The initial state $\mathbf{y}^{(0)}$ may be chosen arbitrarily, e.g. $\mathbf{y}^{(0)} = (0, \ldots, 0)$. At discrete time $t \geq 0$, the *excitation* of any neuron j is defined as $\xi_j^{(t)} = \sum_{i=1}^{s} w(i,j) y_i^{(t)} - h(j)$ including an integer *threshold* $h(j)$ local to unit j. At the next instant $t+1$, one (e.g. randomly) selected neuron j (*sequential* mode) computes its new output $y_j^{(t+1)} = H(\xi_j^{(t)})$ by applying the Heaviside activation function H, that is, j is *active* when $H(\xi) = 1$ for $\xi \geq 0$ while j is *passive* when $H(\xi) = 0$ for $\xi < 0$. For the remaining units $y_i^{(t+1)} = y_i^{(t)}$, $i \neq j$. A Hopfield net reaches a *stable state* $\mathbf{y}^{(t^*)}$ at time $t^* \geq 0$ if $\mathbf{y}^{(t^*)} = \mathbf{y}^{(t^*+1)}$ provided that any unit in the network is updated. The fundamental property of a symmetric Hopfield net is that its dynamics is constrained by *energy* function $E(\mathbf{y}) = -\frac{1}{2} \sum_{j=1}^{s} \sum_{i=1}^{s} w(i,j) y_i y_j + \sum_{j=1}^{s} h(j) y_j$ which is a bounded function defined on its state space whose value decreases along any nonconstant computation path ($\xi_j^{(t)} \neq 0$ is assumed without loss of generality). It follows that starting from any state the network converges towards some stable state corresponding to a local minimum of E [3]. Thus the cost function of a hard combinatorial optimization problem can be encoded into the energy function of a Hopfield net which is then minimized in the course of computation. Hence, the *minimum energy problem* of finding a network state with minimum energy is of special interest which is NP-complete [1,8].

2 The Reduction

Let T be a set of n triangles that represents a triangulated surface model homeomorphic to a sphere in which each edge is incident to at most two triangles. An edge is said to be *internal* if it is shared by exactly two triangles; otherwise it is a *boundary* edge. Denote by I and B the sets of internal and boundary edges, respectively. A *sequential cycle* is a “cycled tristrip” which is an ordered sequence of vertices $C = (v_1, \ldots, v_m)$ where $m \geq 4$ is even, encoding the set of $m-2$ different triangles $T_C = \{\{v_p, v_{p+1}, v_{p+2}\}\,;\, 1 \leq p \leq m-2\}$ so that $v_{m-1} = v_1$ and $v_m = v_2$. Also denote by I_C and B_C the sets of internal and boundary edges of C, respectively, that is, $I_C = \{\{v_p, v_{p+1}\}\,;\, 1 \leq p \leq m-2\}$ and $B_C = \{\{v_p, v_{p+2}\}\,;\, 1 \leq p \leq m-2\}$. An example of the sequential cycle is depicted in Figure 1.b where its internal (*dashed line*) and boundary (*dotted line*) edges are indicated. Let $\mathcal{C}$ be the set of all sequential cycles in T. For each $C \in \mathcal{C}$ one unique *representative* internal edge $e_C \in I_C$ can be chosen as follows. Start with any $C \in \mathcal{C}$ and choose any edge from I_C to be its representative edge e_C. Observe that for a fixed orientation of triangulated surface any internal edge follows either left or right turn corresponding to at most two sequential cycles. Denote by C' the sequential cycle having no representative edge so far which shares its edge $e_C \in I_C \cap I_{C'}$ with C if such C' exists; otherwise let C' be any sequential cycle with no representative internal edge or stop if all the sequential cycles do have their representative edges. Further choose any edge from $I_{C'} \setminus \{e_C\}$ to be the representative edge $e_{C'}$ of C' and repeat the previous step with C replaced by C'. Clearly, each edge represents at most one cycle because set $I_{C'} \setminus \{e_C\} \neq \emptyset$ always contains only edges that do not represent any cycle so far. If it were not the case then another sequential cycle C'' different from C would obtain its representative edge $e_{C''}$ from $I_{C'} \cap I_{C''}$ and hence a representative edge would already be assigned to C' before C is considered.

Hopfield network $\mathcal{H}_T$ corresponding to T will now be constructed. With each internal edge $e = \{v_1, v_2\} \in I$ two neurons ℓ_e and r_e are associated whose states either $y_{\ell_e} = 1$ or $y_{r_e} = 1$ indicate that e follows the left or right turn, respectively, along a tristrip. Let $L_e = \{e, e_1, e_2, e_3, e_4\}$ with $e_1 = \{v_1, v_3\}$, $e_2 = \{v_2, v_3\}$, $e_3 = \{v_2, v_4\}$, and $e_4 = \{v_1, v_4\}$ be the set of edges of the two triangles $\{v_1, v_2, v_3\}$, $\{v_1, v_2, v_4\}$ that share edge e. Denote by $J_e = \{\ell_f, r_f\,;\, f \in L_e \cap I\}$ the set of corresponding neurons. Unit ℓ_e is connected with all neurons from J_e via weights $w(i, \ell_e) = -7$ for $i \in J_{\ell_e} = J_e \setminus \{r_{e_2}, \ell_e, r_{e_4}\}$ except for units r_{e_2} (if $e_2 \in I$), ℓ_e, and r_{e_4} (if $e_4 \in I$) whose states may encode a tristrip that traverses edge e by the left turn. Such a situation (for $L_e \subseteq I$) is depicted in Figure 2.a where the tristrip together with associated active neurons r_{e_2}, ℓ_e, r_{e_4} are marked. Similarly, unit r_e is connected with $i \in J_{r_e} = J_e \setminus \{\ell_{e_1}, r_e, \ell_{e_3}\}$ via

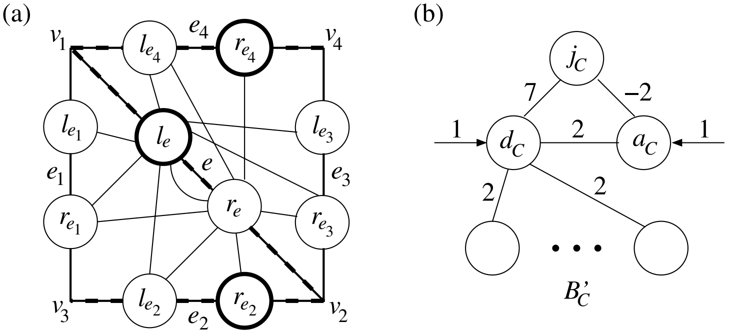


Fig. 2. The construction of Hopfield network $\mathcal{H}_T$

$w(i, r_e) = -7$ corresponding to the right turn. Hence, the states of $\mathcal{H}_T$ with these negative weights that enforce locally the alternation of left and right turns encode tristrips. For each e_C, $C \in \mathcal{C}$, define $j_C = \ell_{e_C}$ if e_C follows the left turn along C or $j_C = r_{e_C}$ if e_C follows the right turn along C. Denote $J = \{j_C\,;\, C \in \mathcal{C}\}$ and $J' = \{\ell_e, r_e \notin J\,;\, e \in I\}$. Define the thresholds of underlying neurons as $h(j) = -5 + 2b_{e(j)}$ for $j \in J'$ and $h(j) = 1 + 2b_{e(j)}$ for $j \in J$ where $e(j) = e$ for $j \in \{\ell_e, r_e\}$ and $b_e = |\{C \in \mathcal{C}\,;\, e \in B'_C\}| \leq 2$ for $B'_C = B_C \setminus L_{e_C}$.

Hopfield network $\mathcal{H}_T$ must avoid the states encoding cycled strips of triangles around sequential cycles [2] which would have less energy E than those encoding the optimal stripifications. Thus two auxiliary neurons d_C, a_C are introduced for each $C \in \mathcal{C}$. Unit d_C computes the disjunction of outputs from all neurons associated with boundary edges B'_C of C which, being active, enables the activation of unit j_C associated with e_C. Hence, any tristrip may pass through e_C along the direction of C only if a boundary edge of C is a part of another tristrip crossing C. This ensures that the states of $\mathcal{H}_T$ do not encode sequential cycles. Unit a_C balances the contribution of d_C to energy E when j_C is passive. Figure 2.b defines the weights and thresholds of d_C, a_C for $C \in \mathcal{C}$. This completes the construction of $\mathcal{H}_T$. Observe that the number of units $s = 2|I| + 2|\mathcal{C}|$ (similarly the number of connections) in $\mathcal{H}_T$ is linear in terms of $n = |T|$ because $|\mathcal{C}| \leq 2|I| = O(n)$ since each internal edge can belong to at most two cycles.

3 The Correctness

Let $\mathcal{S}_T$ be the set of optimal stripifications with the minimum number of tristrips for T. Define $\Sigma \in \mathcal{S}_T$ is *equivalent* with $\Sigma' \in \mathcal{S}_T$, i.e. $\Sigma \sim \Sigma'$ iff $\{T_\sigma\,;\, \sigma \in \Sigma\} = \{T_{\sigma'}\,;\, \sigma' \in \Sigma'\}$. Two equivalent optimal stripifications may differ in a tristrip σ encoding triangles $T_\sigma = T_C$ of sequential cycle C that is split at two different positions. Let $[\Sigma]_\sim = \{\Sigma' \in \mathcal{S}_T\,;\, \Sigma' \sim \Sigma\}$ and denote by $\mathcal{S}_T/_\sim = \{[\Sigma]_\sim\,;\, \Sigma \in \mathcal{S}_T\}$ the partition of $\mathcal{S}_T$ into equivalence classes.

Theorem 1. *Let $\mathcal{H}_T$ be a Hopfield network corresponding to triangulation T with n triangles and denote by $Y^* \subseteq \{0,1\}^s$ the set of stable states that can* *be reached during any sequential computation by $\mathcal{H}_T$ starting at the zero state. Then each state $\mathbf{y} \in Y^*$ encodes a correct stripification $\Sigma_{\mathbf{y}}$ of T into k tristrips and has energy*

$$E(\mathbf{y}) = 5(k - n)\,. \tag{1}$$

In addition, there is a one-to-one correspondence between the classes of equivalent optimal stripifications $[\Sigma]_\sim \in \mathcal{S}_T/_\sim$ having the minimum number of tristrips for T and the states in Y^* with minimum energy $\min_{\mathbf{y}\in Y^*} E(\mathbf{y})$.

Proof. Stripification $\Sigma_{\mathbf{y}}$ is decoded from $\mathbf{y} \in Y^*$ as follows. Denote $I_0 = \{e \in I\,;\, y_{\ell_e} = y_{r_e} = 0\}$ and let $I_1 = I \setminus I_0$. Set $\Sigma_{\mathbf{y}}$ contains each ordered sequence $\sigma = (v_1, \ldots, v_m)$ of $m \geq 3$ vertices that encodes $n(\sigma) = m - 2$ different triangles $\{v_p, v_{p+1}, v_{p+2}\} \in T$ for $1 \leq p \leq m-2$ such that their edges $e_0 = \{v_1, v_3\}$, $e_m = \{v_{m-2}, v_m\}$, and $e_p = \{v_p, v_{p+1}\}$ for $1 \leq p \leq m-1$ satisfy $e_0, e_1, e_{m-1}, e_m \in I_0 \cup B$ and $e_2, \ldots, e_{m-2} \in I_1$. We will prove that $\Sigma_{\mathbf{y}}$ is a correct stripification of T. Observe first that every $j \in J \cup J'$ is passive if there is active $i \in J_j$. This ensures that each $\sigma \in \Sigma_{\mathbf{y}}$ encodes a set T_σ of different triangles whose shared edges follow alternating left and right turns and that sets T_σ, $\sigma \in \Sigma_{\mathbf{y}}$, are pairwise disjoint. In particular, for each $j \in J \cup J'$ the number of weights 2 from some d_C contributing to ξ_j is at most $b_{e(j)} \leq 2$ which are subtracted within threshold $h(j)$. Hence, if all $i \in J_j$ are passive then $\xi_j \leq 5$ for $j \in J'$, and $\xi_j \leq 6$ for $j \in J$ which may include weight 7 from d_C. Thus, active $i \in J_j$ contributing to ξ_j via weight -7 makes j passive due to $\mathbf{y}$ is a stable state. Further, we check $\bigcup_{\sigma\in\Sigma_{\mathbf{y}}} T_\sigma = T$. It suffice to prove that there is no $C = (v_1, \ldots, v_m) \in \mathcal{C}$ such that $e_p = \{v_p, v_{p+1}\} \in I_1$ for all $p = 1, \ldots, m-2$. Suppose that such C exists which implies $B_C \cap I \subseteq I_0$. Then $j_C \in J$ associated with e_C could not be activated during sequential computation of $\mathcal{H}_T$ starting at the zero state since $h(j_C)$ can be reached only by weight 7 from d_C which computes the disjunction of outputs from units i for $e(i) \in B'_C \cap I \subseteq I_0$. Hence, $e_C \in I_0$ which is a contradiction. This completes the argument for $\Sigma_{\mathbf{y}}$ to be a correct stripification of T.

Let $\Sigma_{\mathbf{y}}$ contain k tristrips. Each $\sigma \in \Sigma_{\mathbf{y}}$ is encoded using $n(\sigma) - 1$ edges from I_1. The number of active units in $J' \cup J$ equals $|I_1| = \sum_{\sigma\in\Sigma_{\mathbf{y}}} (n(\sigma) - 1) = n - k$. We will prove that each active $j \in J' \cup J$ is accompanied with a contribution of -5 to energy E which gives (1). Active $j \in J' \cup J$ makes all $i \in J_j$ passive. Neuron j is also connected to $b_{e(j)}$ active units d_C for $e(j) \in B'_C$ computing the disjunctions that include active j. Thus active $j \in J'$ produces contribution $-\frac{1}{2}b_{e(j)}w(d_C, j) - \frac{1}{2}b_{e(j)}w(j, d_C) + h(j) = -b_{e(j)}w(d_C, j) + h(j) = -5$ to E. Similarly, active $j_C \in J$ assumes active d_C and makes a_C passive, which contributes $-b_{e(j_C)}w(j_C, d_C) - w(d_C, j_C) + h(j_C) + h(d_C) = -5$ to E. Unit a_C balances the contribution of active d_C to E when j_C is passive, that is, $-w(a_C, d_C) + h(d_C) + h(a_C) = 0$.

Optimal stripification $\Sigma \in \mathcal{S}_T$ is encoded by state $\mathbf{y}$ of $\mathcal{H}_T$ so that $\Sigma \in [\Sigma_{\mathbf{y}}]_\sim$. Equivalent $\Sigma' \sim \Sigma$ is used to determine $\mathbf{y}$ such that $\Sigma_{\mathbf{y}} = \Sigma'$. Each $\sigma \in \Sigma$ encoding triangles $T_\sigma = T_C$ of some $C \in \mathcal{C}$ is replaced with $\sigma' = (v_1, \ldots, v_m) \in \Sigma'$ having $T_{\sigma'} = T_\sigma$ so that σ' starts with representative $e_C = \{v_1, v_2\}$. Let ℓ_e or r_e from $J' \cup J$ be active iff there is $\sigma = (v_1, \ldots, v_m) \in \Sigma'$ such that its edge

$e = \{v_p, v_{p+1}\}$ for some $2 \le p \le m-2$ follows the left or right turn, respectively. Further, unit d_C for $C \in \mathcal{C}$ is active iff there is active $i \in J' \cup J$ for $e(i) \in B'_C$ while a_C is active iff d_C is active and j_C is passive. It follows that $\mathbf{y}$ is a stable state of $\mathcal{H}_T$. It must still be proven that $\mathbf{y} \in Y^*$.

Define a directed graph $\mathcal{G} = (\mathcal{C}, \mathcal{A})$ where $(C_1, C_2) \in \mathcal{A}$ is an edge of $\mathcal{G}$ iff $e_{C_1} \in B'_{C_2}$. Let $\mathcal{C}'$ be a subset of all vertices $C \in \mathcal{C}$ with $y_{j_C} = 1$ that create directed cycles in $\mathcal{G}$. Suppose that units i are passive for all $e(i) \in \bigcup_{C \in \mathcal{C}'} B'_C \setminus E_{\mathcal{C}'}$ where $E_{\mathcal{C}'} = \{e_C\, ;\, C \in \mathcal{C}'\}$. For each $C \in \mathcal{C}'$ units i for $e(i) \in B_C \cap L_{e_C}$ are also passive due to active j_C. It seems that such a stable state $\mathbf{y}$ could not be reached during any sequential computation by $\mathcal{H}_T$ starting at the zero state since j_C, $C \in \mathcal{C}'$, can be activated only by d_C, $C \in \mathcal{C}'$, which can be activated only by active j_C, $C \in \mathcal{C}'$. Since $\varSigma_{\mathbf{y}} \in \mathcal{S}_T$ the underlying tristrips follow internal edges of $C \in \mathcal{C}'$ as much as possible being interrupted only by edges from $\bigcup_{C \in \mathcal{C}'} B_C \setminus E_{\mathcal{C}'}$. Any $\sigma \in \varSigma_{\mathbf{y}}$ crossing some $C_1 \in \mathcal{C}'$ (i.e., $\emptyset \neq T_\sigma \cap T_{C_1} \neq T_{C_1}$) has one its end within C_1 because σ may enter C_1 only through $e_{C_2} \in B_{C_1}$ (i.e., $y_{j_{C_2}} = 1$) which is the only representative edge of $C_2 \in \mathcal{C}'$ that σ follows. It can be proven [6] that there exists $C \in \mathcal{C}'$ containing two $\sigma_1, \sigma_2 \in \varSigma_{\mathbf{y}}$ such that $T_{\sigma_1} \subseteq T_C$ and $T_{\sigma_2} \subseteq T_C$. Hence, $\varSigma'_{\mathbf{y}}$ with fewer tristrips can be constructed from $\varSigma_{\mathbf{y}}$ by introducing only one $\sigma^* \in \varSigma'_{\mathbf{y}}$ such that $T_{\sigma^*} = T_C$ (e.g. $y_{j_C} = 0$) instead of the two $\sigma_1, \sigma_2 \in \varSigma_{\mathbf{y}}$ while any $\sigma \in \varSigma_{\mathbf{y}}$ crossing and thus ending within C is shortened to $\sigma' \in \varSigma'_{\mathbf{y}}$ so that $T_{\sigma'} \cap T_C = \emptyset$ which does not increase the number tristrips. This is a contradiction with $\varSigma_{\mathbf{y}} \in \mathcal{S}_T$, and hence $\mathbf{y} \in Y^*$. Obviously, $[\varSigma_{\mathbf{y}}]_\sim$ with the minimum number of tristrips corresponds uniquely to $\mathbf{y} \in Y^*$ having minimum energy $\min_{\mathbf{y} \in Y^*} E(\mathbf{y})$ according to equation (1). □

References

1. Barahona, F.: On the computational complexity of Ising spin glass models. Journal of Physics A: Mathematical and General **15** (10) (1982) 3241–3253
2. Estkowski, R., Mitchell, J.S.B., Xiang, X.: Optimal decomposition of polygonal models into triangle strips. In: Proceedings of the 18th Annual Symposium on Computational Geometry. ACM Press, New York (2002) 254–263
3. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences USA **79** (8) (1982) 2554–2558
4. Hopfield, J.J., Tank, D.W.: "Neural" computation of decision in optimization problems. Biological Cybernetics **52** (3) (1985) 141–152
5. Pospíšil, D.: Generating triangle strips by Hopfield network. Student's project (in Czech), Faculty of Informatics, Masaryk University, Czech Republic (2002)
6. Šíma, J.: Tristrips on Hopfield networks. Technical report V-908, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague (2004)
7. Šíma, J.: Generating sequential triangle strips by using Hopfield nets. In: Proceedings of the ICANNGA'2005 7th International Conference on Adaptive and Natural Computing Algorithms. Springer-Verlag, Vienna (2005) 25–28
8. Šíma, J., Orponen, P.: General-purpose computation with neural networks: A survey of complexity theoretic results. Neural Computation **15** (12) (2003) 2727–2778

A Biophysical Model of Decision Making in an Antisaccade Task Through Variable Climbing Activity

Vassilis Cutsuridis[1,*], Ioannis Kahramanoglou[1], Stavros Perantonis[1],
Ioannis Evdokimidis[2], and Nikolaos Smyrnis[2,3]

[1] Computational Intelligence Laboratory, Institute of Informatics and Telecommunications,
National Center for Scientific Research "Demokritos", Agia Paraskevi, Athens GR-15310
{vcut, ikahra, sper}@iit.demokritos.gr
[2] Cognition and Action Group, Neurology Department, National University of Athens,
Aeginition Hospital, 72 Vas Sofias Ave, Athens GR-11528
[3] Psychiatry Department, National University of Athens, Aeginition Hospital,
72 Vas Sofias Ave, Athens GR-11528
smyrnis@med.uoa.gr

Abstract. We present a biophysical model of saccade initiation based on competitive integration of planned and reactive cortical saccade decision signals in the intermediate layer of the superior colliculus. In the model, the variable slopes of the climbing activities of the input cortical decision signals are produced from variability in the conductances of Na^+, K^+, Ca^{2+} activated K^+, NMDA and GABA currents. These cortical decision signals are integrated in the activities of buildup neurons in the intermediate layer of the superior colliculus, whose activities grow nonlinearly towards a preset criterion level. When the level is crossed, a movement is initiated. The resultant model reproduces the unimodal distributions of saccade reaction times (SRTs) for correct antisaccades and erroneous prosaccades as well as the variability of SRTs (ranging from 80ms to 600ms) and the overall 25% of erroneous prosaccade responses in a large sample of 2006 young men performing an antisaccade task.

1 Introduction

In the brain, climbing activity is a prominent profile of neuronal activity observed in the thalamus, superior colliculus, primary motor cortex, prefrontal cortex and other brain areas and it is found to be related to the anticipation of forthcoming events and to the generation of movements. Climbing activity spans from hundreds of milliseconds up to tens of seconds [1]. In the frontal eye fields of monkeys there are populations of visuomotor neurons that begin to fire in advance of saccades, with their activity rising linearly upon presentation of a suitable target stimulus [5]. Buildup cells in the monkey's superior colliculus (SC) begin to linearly build up their activity after the signal to make a saccade is presented [8]. The rate of rise varies randomly from trial to trial and the saccade is initiated when this activity reaches a fixed threshold [5], [6].

[*] Corresponding author.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 205 – 210, 2005.

The model presented in this paper is an attempt to model the biophysical mechanisms underlying the generation of slowly varying climbing, temporal integrator-like activity of the reactive and planned input decision signals of a SC model in an antisaccade task [9]. This work combines and extends previous biophysical models [1], [9], [10].

2 Materials and Methods

2.1 Basis of the Model

In a modeling attempt of the antisaccade task [4], Cutsuridis and colleagues [9] hypothesized that the preparation of an antisaccadic eye movement consisted of two cortically independent and spatially separated decision signals representing the reactive and planned saccade signals, whose linearly rising phases are derived from two normal distributions with different means and standard deviations. These two cortical decision signals were then integrated at opposite colliculi locations, where they competed against each other via lateral excitation and remote inhibition. A saccade was initiated when these decision processes, represented by the neuronal activity of SC buildup neurons with nonlinear growth rates varying randomly from a normal distribution, gradually build up their activity until reaching a preset criterion level. The crossing of the preset criterion level in turn released the “brake” from the SC burst neurons and allowed them to discharge resulting in the initiation of an eye movement. The model’s main prediction was that there is no need of a top-down inhibitory signal that prevents the error prosaccade from being expressed, thus allowing the correct antisaccade to be released. Moreover, the model offered a functional rationale at the SC neuronal population level of why the antisaccadic reaction times are so long and variable and simulated accurately the correct and error antisaccade latencies, the shape distributions and the error probabilities.

Our intention in this study is to model the biophysically plausible mechanisms that can produce climbing activity with adjustable slope. We extend the SC model by adding two cortical modules that will generate the planned and reactive decision signals. The decision signals will be derived from the population activities of networks of pyramidal neurons and inhibitory interneurons. We will use Hodgkin-Huxley mathematical formulations to explore the biophysical mechanisms that give rise to the randomly varying climbing activities of the cortical decision signals. These decision signals will then drive the SC model and generate correct antisaccade and error prosaccade reaction time (RT) distributions as well as response probabilities. These simulated RT distributions and error probabilities will be compared to psychophysically derived latency distributions and error probabilities [3], [7].

2.2 Architecture

Standard Hodgkin-Huxley modeling techniques were used to simulate networks of single compartmental models of cortical pyramidal neurons and cortical inhibitory interneurons (IN). Pyramidal neuron membrane potential obeyed

$$CdV / dt = -(I_{leak} + I_{Na} + I_{NaP} + I_{KS} + I_{HVA} + I_{C} + I_{DR} + I_{AHP} + I_{AMPA} + I_{NMDA} + I_{GABA}) + I_{inj} \quad (1)$$

with C_m = 1 μF cm^{-2}. GABAergic inhibitory interneuron membrane potential obeyed

$$CdV / dt = -(I_{leak} + I_{Na} + I_{DR} + I_{AMPA} + I_{NMDA} + I_{GABA}) + I_{inj} \quad (2)$$

with C_m = 1 μF cm^{-2}. Ionic currents I_{Na}, I_{NaP}, I_{Ks}, I_C, I_{DR}, and I_{HVA} were modeled as in [10], whereas I_{AHP} was modeled as in [1]. Table 2 of [10] provided a summary of the gating variables and their respective powers for all ionic conductances used in this study. The synaptic currents (I_{AMPA}, I_{NMDA}, and I_{GABA}) were given by double exponential functions exactly as in [1]. Synaptic short term dynamics were determined by the available synaptic efficacy (R) and a utilization variable (u) exactly as in [1]. We simulated low spontaneous background activity in the network, by delivering random noise to all pyramidal and GABAergic cells, generated from Poisson processes convolved with the AMPA, NMDA and GABA synaptic conductances. Because very little is known about the detailed connectivity of neurons and the associated synaptic strengths in the frontal cortices, we intentionally kept the network model as general as possible. Two networks of 10 pyramidal cells and 5 GABAergic interneurons each were simulated. In each network, we assumed that all pyramidal cells and GABAergic interneurons were fully connected [10]. The output of each network was the average population activity of a homogenous population of neurons with identical connections. These outputs were then used as the input drives of the superior colliculus (SC) model [9].

2.3 Implementation

The simulations were performed on a Pentium IV 3.2 GHz PC with MATLAB's version R13 installed. The whole system of differential and algebraic equations was implemented in MATLAB (The MathWorks, Inc, Natick, MA). The differential equations of the cortical neural integrator model were integrated numerically using one of the MATLAB ordinary differential equation solvers (mainly ode23s, a one step solver based on modified Rosenbrock formula of order 2 [2]) with time step *Δt = 0.001 ms*. The differential equations of the SC model were integrated numerically using one of the MATLAB ordinary differential equations solvers (ode45, an implicit solver based on the Dormand-Prince pair method [2]) with time step *Δt = 0.001 ms*). Relative (error) tolerance was set to 10^{-6}.

3 Experiments and Results

3.1 Experimental Setup

The data used in this study were collected in an antisaccade task [3], [7]. Details of the experimental procedure used for the collection of these data are described therein [3], [7]. Briefly, 2006 conscripts of the Greek Air Force were instructed to perform eye movements in the opposite direction from the location of a stimulus that appears in their right or left peripheral visual field while they are fixating on a central stimulus. The correct or error saccade reaction time (SRT) was measured in each trial for every subject. Trials with reaction times < 80 ms were excluded as anticipations and trials with reaction times > 600 ms were excluded as no response trials. The median RT and the inter-quartile range for antisaccades and error prosaccades of all 2006 conscripts were grouped into ten virtual groups after performing clustering analysis using the STATISTICA software version 5.5 (StatSoft, Inc, Tulsa, OK). The purpose of the cluster analysis was to partition the observations into groups ("clusters") so that the pairwise dissimilarities between those assigned to the same cluster tend to be smaller than those in a different cluster. We demonstrate below the results from all ten clusters.

3.2 Results

The observed variability in the rising phase (slope) of the average firing rates of the pyramidal neurons was found to be due to noise in the conductances of the I_{NaP} and I_{NMDA} currents. Noise in the conductances of I_{Ks}, I_{DR}, I_C and I_{HVA} currents didn't produce any variability in the rising phase of the average firing rate. The slope of the climbing activity was carefully adjusted so that the simulated correct and error RT distributions and the error probabilities to approximate the experimental ones in an antisaccade task (see Table 1). We estimated the slopes of the rising phases of the average firing rates of two cortical networks of neurons in each trial by fitting to them a straight line. We used these slope values as values of the slopes of the rising phases of the planned and reactive inputs of [9]. The slope values of the reactive and planned inputs were sorted in ascending order, so that the slope of the reactive input was always greater than the slope of the planned input. The threshold was adjusted, so that the simulated error rate closely matched the observed. Its value was set to a different value for each group, but it was kept fixed across trials for each group [9].

Table 1. (*Columns 2-4*) Simulated correct median, error median, and error rate for average and all ten groups. Values in parentheses stand for experimental values. Units: correct SRT (ms), error SRT (ms). (*Columns 5-6*) Values of χ^2 test of homogeneity between correct and error experimental and simulated percent density distributions for antisaccades and error prosaccades. χ^2 values marked with an asterisk indicate a significant difference between the simulated and the observed RT distributions. Rejection region: $\chi^2 \geq \chi^2_{0.05}$ (37.65). The degrees of freedom were 25.

	Median RT of antisaccades	Median RT of error prosaccades	% antisaccade error rate	antisaccade χ^2 value	prosaccade χ^2 value
G 1	254.80 (242.40)	212.99 (216.66)	24.27 (17.02)	35.21	24.18
G 2	282.38 (288.44)	188.10 (193.66)	23.93 (28.86)	31.82	27.97
G 3	263.10 (251.79)	180.63 (175.53)	20.87 (24.79)	30.34	21.82
G 4	365.69 (349.42)	218.99 (221.36)	37.00 (34.58)	36.46	35.67
G 5	218.20 (213.58)	177.85 (172.77)	27.36 (24.92)	35.21	24.18
G 6	294.174 (288.16)	279.541(265.20)	13.04 (16.15)	36.15	34.92
G 7	276.50 (279.21)	202.97 (201.96)	38.62 (39.07)	90.5^*	33.56
G 8	281.89 (280.91)	212.54 (201.92)	20.15 (23.73)	32.16	32.89
G 9	251.30 (249.27)	209.90 (211.65)	12.41 (12.02)	56.06^*	96.24^*
G 10	327.56 (307.5)	331.07 (326.99)	20.05 (21.81)	33.88	83.57^*

The SC model was allowed to run for 1000 trials in each group. We recorded the simulated median antisaccade and error prosaccade RT values and the error rates for each group (see Table 1). In order for each group to compare the SRT distributions of the real experimental data with the simulated SRT distributions, we normalized the SRT distribution of each subject data and then added the normalized distributions for all subjects belonging to the same group. More specifically, the time interval between the 80 ms and 600 ms was divided into twenty-six categories, each lasting 20 ms (e.g. category 1 was between 80 ms and 100 ms, category 2 between 100 ms and 120 ms, and so forth). For each category we calculated its percent relative frequency of response times. The mean frequency for all subjects in a group was then calculated. The discrepancy in each category between the simulated and experimental correct and error distributions was measured by the squared difference between the observed (simulated) and the expected (experimental) frequencies divided by the expected frequency ((Observed – Expected) ^{2} / Expected). The χ^2 value was the sum of these quantities for all categories. The rejection region was set at $\chi^2 \geq \chi^2_{0.05}$. The χ^2 test of homogeneity tested the null hypothesis of whether the simulated and experimental normalized distributions of SRTs for antisaccades and error prosaccades differ between them and showed a significant difference in 2 of the 10 comparisons for antisaccade RT distributions and 2 of the 10 comparisons for the error prosaccade RT distributions (see Table 1).

4 Conclusion

The simulations of the model presented here show that the randomly varying climbing activities of the input decision signals of a SC model in an antisaccade task are due to the interplay of K^+, Na^+, and Ca^{2+} activated K^+ currents as well as due to variability of NMDA synaptic currents. The model is successful at predicting the correct antisaccade and error prosaccade RT distributions as well as the response probabilities from a population of 2006 subjects. There are further paradigms and architectures to which this model can be extended to. For instance, in this study we assumed that the internal properties of all pyramidal neurons in the network were the same (homogenous) and that the connectivity was symmetric. However, real populations of neurons will always have a certain degree of heterogeneity in their internal parameters and in their connectivity patterns. For this reason, we are in the process of examining other more realistic cases of neuronal connectivity in our network. Finally, we are investigating in a more systematic way which ionic conductances have the strongest effects on the rising phase of the average firing rate of the pyramidal neurons and what are the mechanisms that cause the variability in the climbing activities.

References

1. Durstewitz, D.: Self Organizing Neural Integrator Predicts Interval Times Through Climbing Activity. J. Neurosci. 23(12) (2003) 5342-5353
2. Shampine, L.F., Reichelt, M.W.: The MATLAB ODE Suite. SIAM Journal on Scientific Computing. 18 (1997) 1-22

3. Evdokimidis, I., Smyrnis, N., Constantinidis, T.S., Stefanis, N.C., Avramopoulos, D., Paximadis, C., Theleritis, C., Efstratiadis, C., Kastrinakis, G., Stefanis, C.N.: The Antisaccade Task in a Sample of 2006 Young Men I. Normal Population Characteristics. Exp Brain Res. 147 (2002) 45-52
4. Hallett, P.R.: Primary and Secondary Saccades to Goals Defined by Instructions. Vis. Res. 18 (1978) 1279-1296
5. Hanes, D.P., Schall, J.D.: Neural Control of Voluntary Movement Initiation. Science. 274 (1996) 427-430
6. Reddi, B.A.J., Carpenter, R.H.S.: The Influence of Urgency on Decision Time. Nat. Neurosci. 3 (2000) 827-831
7. Smyrnis, N., Evdokimidis, I., Stefanis, N.C., Constantinidis, T.S., Avramopoulos, D., Theleritis, C., Paximadis, C., Efstratiadis, C., Kastrinakis, G., Stefanis, C.N.: The Antisaccade Task in a Sample of 2006 Young Males II. Effects of Task Parameters. Exp. Brain Res. 147 (2002) 53-63
8. Everling, S., Dorris, M.C., Klein, R.M., Munoz, D.P.: Role of Primate Superior Colliculus in Preparation and Execution of Anti-saccades and Prosaccades. J. Neurosci. 19(7) (1998) 2740-2754
9. Cutsuridis, V., Smyrnis, N., Evdokimidis, I., Perantonis, S.: A Neural Model of Decision Making by the Superior Colliculus in an Antisaccade Task. submitted to Biol. Cybernetics
10. Durstewitz, D., Seamans, J.K., Sejnowski, T.J.: Dopamine-Mediated Stabilization of Delay-Period Activity in a Network Model of Prefrontal Cortex. J. Neurophys. 83 (2000) 1733-1750

Can Dynamic Neural Filters Produce Pseudo-Random Sequences?

Yishai M. Elyada[1] and David Horn[2]

[1] Max-Planck Institute of Neurobiology,
Department of Systems and Computational Neurobiology,
Am Klopferspitz 18, D-82152 Martinsried, Germany
elyada@neuro.mpg.de
[2] School of Physics and Astronomy,
Raymond & Beverly Sackler Faculty of Exact Sciences,
Tel-Aviv University, Tel-Aviv 69978, Israel
horn@tau.ac.il

Abstract. Dynamic neural filters (DNFs) are recurrent networks of binary neurons. Under proper conditions of their synaptic matrix they are known to generate exponentially large cycles. We show that choosing the synaptic matrix to be a random orthogonal one, the average cycle length becomes close to that of a random map. We then proceed to investigate the inversion problem and argue that such a DNF could be used to construct a pseudo-random generator. Subjecting this generator's output to a battery of tests we demonstrate that the sequences it generates may indeed be regarded as pseudo-random.

1 Introduction

Dynamic Neural Filters (DNFs) [1] are recurrent networks that transform input space into spatiotemporal behavior. Their dynamics are defined by

$$n_i(t) = H(\sum_j w_{ij} n_j(t-1) - \theta_i) \tag{1}$$

where $n_i(t)$ is the i-th neuron's activation state at time t, H is the Heaviside step function, w_{ij} is the synaptic coupling matrix and θ_i are the neuronal thresholds. This one-step dynamics defines a relation between a state of the system $(n_i)_{i=1\ldots N}$ at time $t-1$ and the state of the i-th neuron at time t.

It is well known that symmetric synaptic matrices lead only to fixed points or two-cycles [2] whereas anti-symmetric ones can lead up to four cycles [3]. The largest cycles may be expected from asymmetric matrices, i.e. ones whose asymmetry $\alpha = \frac{\sum w_{ij} w_{ji}}{\sum w_{ij} w_{ij}}$ is close to zero, and indeed these matrices have been shown to result in cycles that are exponentially long in N, the size of the system [4,5]. Furthermore, it has been shown that the choice

$$\theta_i = \frac{1}{2} \sum_j w_{ij} \tag{2}$$

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 211–216, 2005.

guarantees that the output of this network will have the largest cycle that a given DNF may possess [1].

In this paper, we show that choosing random orthogonal weight matrices results in significantly longer cycles, the average length of which approaches that of a random mapping. Motivated by this result, we next consider these networks as pseudo-random generators (PRGs) and test the quality of the sequences they generate. Previously known PRGs based on simpler, Hopfield-like networks have been subjected only to partial testing with short generated bit sequences [6]. Other PRG candidates are Cellular Neural Networks [7] however they necessitate a fairly complicated emulation of cellular automata.

2 Optimal Choice of the Weight Matrix

Given the optimal threshold (2) let us prove that, for each neuron, the space of states is naturally divided into two halves: the number of activating states (i.e. the ones whose occurrence at $t-1$ implies $n_i(t) = 1$) is equal to the number of inactivating ones. Assume s to be an activating state. Its complement $\bar{s}$, where $\bar{s}_i = 1 - s_i$ will then be inactivating. This follows from the fact that if $\sum_j w_{ij}s_j > \theta_i = \frac{1}{2}\sum_j w_{ij}$ then $\sum_j w_{ij}(1-s_j) < \sum_j w_{ij} - \frac{1}{2}\sum_j w_{ij} = \theta_i$. Obviously the complement of every inactivating state will be an activating one[1] thus completing the proof.

This leads us to suggest imposing orthogonality on the weight matrix by applying the Gram-Schmidt orthogonalization process to N initial row vectors drawn randomly from a Gaussian distribution. The intuition behind this is that it serves to guarantee maximum lack of correlation between outputs of different neurons at any given time. Moreover, since each neuron's hyperplane divides the state-space into two halves in an independent manner, the partitions defined over all the neurons in the system are optimally small; this reduces the probability of closing a cycle in each step, allowing for the production of longer sequences. Numerical testing shows that the largest partitions created by random orthogonal matrices are never larger than about $\frac{2}{3}N$ states.

Figure 1 compares the average length L of cycles as function of N, the total number of neurons, for three different choices of weight matrices: random Gaussian matrices, asymmetrized matrices and random orthogonal matrices. All show exponential increase of the cycle length L with N. Using the parametrization $L = e^{\beta N + \beta_0}$ (choosing $N = 20, 25, 30$ to avoid the finite size effect reported in [5]), we find β values of .215 for the random matrices, .243 for the asymmetrized matrices (compare with [5]), and .320 for the random orthogonal weight matrices. The latter is closer to the exponential coefficient for completely random maps $\frac{\log 2}{2} = .347$. This result is very encouraging given the fact that the DNF is after all a deterministic system.

[1] These statements of complementarity are true for all but a negligible set of matrices where an equality, rather than inequality, can be obtained for some neuron i and some state s. When dealing with integer matrices this can be explicitly avoided by requiring the sum of each row in the weight matrix to be odd.

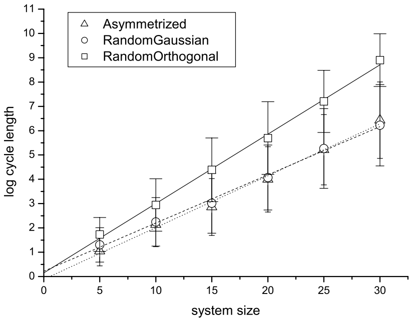


Fig. 1. Dependence of DNF sequence length on N for 3 different methods of generating weight matrices: Averages and standard errors over 100 trials for each system size are shown, with regression lines.

3 Can DNFs Generate Pseudo-Random Sequences?

Pseudo-random generators are functions that expand short, random bit sequences ('seeds') into long pseudo-random bit sequences [8,9] that are computationally indistinguishable from truly random sequences; in other words, the next bit should not be predictable with probability significantly different from $\frac{1}{2}$. PRGs are important primitives in cryptology, and they can be used as a source of effective randomness in the so-called one-time pad cryptosystem (see, e.g., [10]).

PRGs can be constructed using one-way functions (OWFs), by using a hardcore predicate of the OWF (see, e.g., [11] section 3.3.3). Such a construction can be implemented by iteratively applying the function to its own output, and outputting the hardcore predicate at each iteration. For one-to-one length preserving OWFs (bijections of the state-space onto itself), the parity of a certain subset of the bits of the input comprises a hardcore predicate of these functions ([12]).

Is the DNF-step function an OWF? Computing one step of a given DNF dynamics is a simple task, amounting to polynomial time complexity given the digital encoding of a particular DNF. On the other hand, *inverting* one step of the dynamics, i.e. finding a source state for a given target state, is equivalent to the $\mathcal{NP}$-complete problem Integer Programming (*IP*), finding $x \in \{0,1\}^n$ such that $Wx \geq b$ for some matrix W and vector b. Clearly, the fact that a function is $\mathcal{NP}$-complete does not imply that it is also a OWF, but we will, nevertheless, try to provide evidence for the validity of our conjecture, i.e. that the DNF dynamics function is hard to invert in most cases, qualifying it as a good candidate for a OWF.

To do so, we rely on the well developed field of IP algorithms (see, e.g., [13], [14]). Since these methods were developed to solve general IP problems we use them as a benchmark for the hardness of our problem. Fig. 2 shows the results

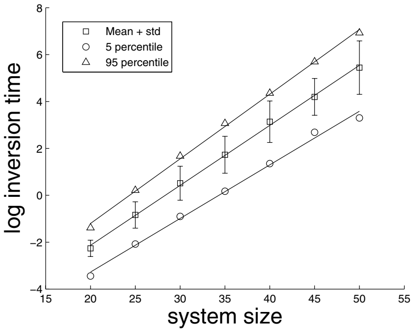


Fig. 2. Dependence of the log inversion time of DNF-step on the size of the system (N). For each N, 100 initial states were randomly chosen for a single random orthogonal DNF and one step was calculated. The mean log time taken by GLPK to find an initial state given the second state is presented as a function of N.

of applying the linear programming package GLPK [15] to inverting the step function of DNFs with randomly generated orthogonal weight matrices. The dependence of average solution time on the number of neurons in the system is clearly exponential, testifying to the hardness of solution by this algorithm. We can implement a construct that outputs the parity of a certain subset of bits within the bounds of the DNF model by a DNF 'parity-gadget', which can be added to a given DNF in order to compute the parity of the chosen set of k bits in two steps. The parity gadget consists of a block of k neurons, the i-th of which calculates after one step whether the previous state of the original DNF contained more than i activated neurons, and a parity neuron with a weight vector of $(1, -1, 1, -1...)$ and threshold $\theta = 1$, which uses only inputs from the k-neuron block to calculate the parity of the original DNF two steps previously.

We stress that until we find a weight-matrix construction algorithm that generates preimages of size 1, making the DNF-step function a 1-1 length preserving function, the construct does not necessarily constitute a PRG, even if the DNF-step function is one-way. Nevertheless, we will show some evidence that this construct might still be able to generate pseudo-random sequences, despite this shortcoming.

4 Performance of the DNF Generator

Although our random orthogonal DNFs are not 1-1 functions of the state-space, the DNF generator construct can generate sequences that pass statistical pseudo-randomness tests. We tested this with a comprehensive battery of such tests, available from NIST (http://csrc.nist.gov/rng/) [16,17] as a means of detecting non-randomness in binary bit sequences generated by pseudo-random generators for cryptographic applications. Generally speaking, consistent failures in any of the 189 listed tests (including multiple variants of several tests) immediately suggests an attack scheme, either explicitly outlined by the test itself, or by the theorem relating unpredictability to indistinguishability from a truly random sequence [8,9]. For each test, the suite generates randomness p-values for all the tested sequences and then uses two final tests to check for deviation from randomness over the set of all 70 bit sequences. The first is a goodness-of-fit test for uniformity of the p-values, and the second is the proportion of sequences that failed each particular test with a 0.01 rejection rate (i.e. $p < 0.01$ for that test).

We used the NIST test suite on 70 1Mbit DNF generator sequences with parity gadgets computing the parity of 5 arbitrary bits. Out of a total of 189 uniformity of p-values and 189 proportion of failure tests, only 2 of the former and 3 of the latter failed. This result is similar to results for other well-tested PRGs such as the Blum-Blum-Shub generator and the RSA generator.

An important point to be made here is that this test suite does not make any use of the information available in the weight matrix. There might still exist, therefore, other ways to 'break' the DNF generator based on this, despite the results shown in Fig. 2. On the other hand, this might also suggest that the DNF generator construct, *without* the weight-matrix being publicly available, is strong enough to generate cryptographically safe pseudo-random sequences.

5 Discussion

Unpredictability is a desired feature of animal behavior in many pursuit-evasion scenarios, with countless examples for such behavior found in nearly all animal groups [18]. Apparently, generating seemingly random sequences of behavior is an innate ability of humans when placed in a competitive situation where unpredictability is an optimal strategy [19], and this ability can also be enhanced by feedback [20,21].

In a presumably deterministic brain, the use of unpredictability as a strategy for survival or evasion must assume a deterministic mechanism of generating unpredictable behavior, in other words, a pseudo-random behavior generator. The model we present suggests that this can be implemented by a simplified neural network without any stochasticity assumptions.

Besides its obvious and straightforward applicability in encryption schemes, the conjectured ability of DNFs to generate pseudo-randomness also holds implications for current-day paradigms of experimental neural research. Namely, partial knowledge of the dynamics of a simple neural system is not always sufficient for effective extrapolation beyond the given data. This is still possible even if the dynamics are fully known for a subset of the neurons and the connectivity of the system is given. Put in another way, recording activity from a few neurons in a large neural system does not guarantee any effective generalization of these data.

References

1. Quenet, B., Horn, D.: The dynamic neural filter: a binary model of spatiotemporal coding. Neural Comput. **15** (2003) 309–329
2. Hertz, J., Krogh, A., Palmer, R.G.: Introduction to the Theory of Neural Computation. Addison-Wesley Longman Publishing (1991)
3. Peretto, P.: An Introduction to the Modeling of Neural Networks. Cambridge University Press (1992)
4. Gutfreund, H., Reger, D.J., Young, A.P.: The nature of attractors in an asymmetric spin glass with deterministic dynamics. J. Phys. A: Math. Gen. **21** (1988) 2775–2797
5. Bastolla, U., Parisi, G.: Attractors in Fully Asymmetric Neural Networks. J. Phys. A: Math. Gen. **30** (1997) 5613–5631
6. Karrasa, D.A., Zorkadis, V.: On neural network techniques in the secure management of communication systems through improving and quality assessing pseudorandom stream generators. Neural Networks **16** (2003) 899–905
7. Crounse, K., Yang, T., Chua, L.O.: Pseudo-random sequence generation using the cnn universal machine with applications to cryptography. Proc. IVth IEEE International Workshop on Cellular Neural Networks and Their Applications (1996) 433–438
8. Yao, A.: Theory and applications of trapdoor functions. Proc. 23th FOCS (1982) 464–479
9. Blum, M., Micali, S.: How to generate cryptographically strong sequences of pseudo-random bits. SIAM J. Comput. **13** (1984) 850–864
10. Massey, J.L.: An introduction to contemporary cryptology. Proc. of the IEEE **76** (1988) 533–549
11. Goldreich, O.: Foundations of Cryptography: Basic Tools. Cambridge University Press (2001)
12. Goldreich, O., Levin, L.: Hard-core predicates for any one-way function. Proc. of the 21st ACM STOC (1989) 25–32
13. Nemhauser, G., Wolsey, L.: Integer and Combinatorial Optimization. John Wiley and Sons (1988)
14. Johnson, E., Nemhauser, G., Savelsbergh, M.: Progress in linear programming based branch-and-bound algorithms: An exposition. INFORMS J. Comp. **12** (2000) 2–23
15. Makhorin, A.: Gnu linear programming kit version 4.4. Free Software Foundation (2004)
16. Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J., Vo, S.: A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST (2001) http://csrc.nist.gov/rng/SP800–22b.pdf
17. Soto, J.: Statistical testing of random number generators. Proc. 22nd NISSC (1999)
18. Driver, P.M., Humphries, N.: Protean Behavior: The Biology of Unpredictability. Oxford University Press (1988)
19. Rapoport, A., Budescu, D.V.: Generation of random series in two-person strictly competitive games. J. Exp Psych: Gen. **121** (1992) 352–363
20. Neuringer, A.: Can people behave randomly? the role of feedback. J. Exp Psych: Gen. **115** (1986) 62–75
21. Neuringer, A., Voss, C.: Approximating chaotic behavior. Psych. Sci. **4** (1993) 113–119

Making Competition in Neural Fields Suitable for Computational Architectures

Hervé Frezza-Buet[1] and Olivier Ménard[1,2]

[1] Supélec, 2 rue Edouard Belin, F-57070 Metz, France
Herve.Frezza-Buet@supelec.fr
[2] Loria, BP 239 - 54506 Vandoeuvre-lès-Nancy Cedex, France
Olivier.Menard@supelec.fr

Abstract. In this paper, a new competition mechanism for neural fields is proposed, as well as first experimental studies of its robustness. The computational properties of this algorithm are discussed, arguing that such properties are suitable for neural architectures, where some restrictions of the usual neural fields competition methods are not acceptable.

1 The Role of Competition

From a functional and computational point of view, neural fields are bi-dimensional fields, tiled with identical elementary computational units. Some *lateral* connections, linking neurons to their *neighbors* in the field, gives the global computation an intrinsically bi-dimensional nature. This structure is clearly inspired by the anatomy of the cerebral cortex, that has been described as a sheet, tiled with elementary neural circuits, the cortical columns, as reviewed in [1]. The very nature of cortical computation seems to be grounded on that bi-dimensional topology. A well known example is orientation selectivity in V1 [2]. These observations have lead to self-organizing computational architectures, whose central point is the setting of a competition over the neural field. That competition can be viewed as a distributed decision process, contrasting activities in the field [1].

Self-organization is a computational property that has been stressed by early studies in artificial neural networks. This property is in most cases the result of a competitive learning among computational units that owns a preferred value. The activation of a unit depends on the fitting of the input it receives to its preferred value (*matching*), and this activation is used to perform competition among units. The basic idea is that units learn by adapting their preferred input according to the actual one they receive, but the key point is that such a learning is modulated by competition. The result is the setting of preferred inputs so that units represent at best the actual (and a priori unknown) input distribution. These general terms apply to all unsupervised learning techniques in the field of neural networks, but also to statistical methods as k-means.

Recently, we have proposed a multi-map architecture [3,4], where such a local competitive processes is crucial, since it is responsible for the global coordination of activities in the different maps. Details of this work are out of the scope of the

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 217–222, 2005.

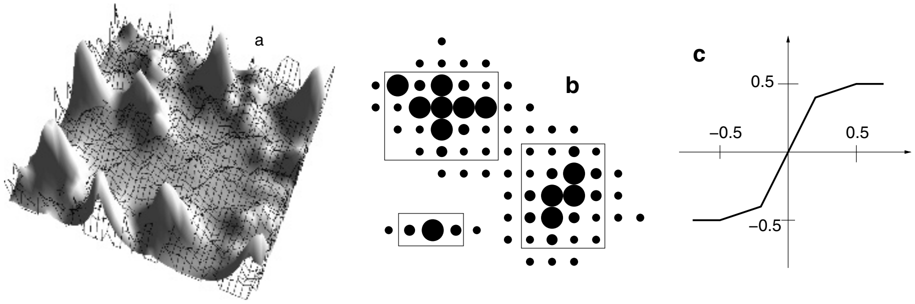


Fig. 1. (a) Black grid is the $i(t)$ values, and gray surface is the resulting $u(t)$ bubbles. (b) The frame of a bubble is the smallest frame including a contiguous set of u values over the threshold θ. The frames that contain u values that are all under threshold τ are ignored in experimental measurements. (c) Approximation of a sigmoid function.

present paper, but the design of such an architecture has been the motivation to propose a new competition algorithm that overcomes the weaknesses of existing techniques in that context.

2 Setting Up Competition

In this section, the most commonly used competition mechanisms are presented, stressing their computational advantages and limits. Their plausibility as a biological model, that sometimes motivates these mechanisms, is not discussed since this paper rather focuses on computational properties.

The fastest competition mechanism is the winner-takes-all mechanism used in the Self Organizing Maps (SOMs) [5] by Kohonen. Once the winner is found, a Gaussian-shaped learning rate kernel is applied around it, so that the winner and its close neighbors learn. In this approach, competition consists of an explicitly search for the best matching unit. The drawbacks are mainly twofolds. First, this search breaks intrinsic neural parallelism, that some implementation would take advantage of [6]. Second, it chooses only one winner in thefield, and is thus not suitable for large fields where many learnings should stand in parallel, at separated places in the neural surface.

Another way to perform competition over a bi-dimensional neural field is the setting up of two sets of lateral connections. The first one contain short range excitatory connections, whereas the second one contains inhibitory longer range connections. This view is close to biology, and has lead to early modeling of self-organizing processes [7]. Let us denote $i_k(t)$ the matching value of a unit k at time t. The value $i_k(t)$ is high when the current input at unit k is close to its preferred one. We will also call $u_k(t)$ the activation of unit k at time t, and ω^+_{kl} (resp. ω^-_{kl}) the excitatory (resp. inhibitory) connection weights to unit k from neighboring units l. Each of the ω^+_{kl} and ω^-_{kl} usually form a Gaussian-shaped positive distribution of weights over the neural field, centered around unit k. Let

$\omega_{kl} = \omega^+_{kl} - \omega^-_{kl}$ be the resulting lateral influence, that commonly has a Mexican hat-shaped distribution. This defines a so-called on-center off-surround lateral influence. Last, let $f\,[.]$ be a sigmoid-type non linearity. The purpose of such an architecture is to make the distribution of u_k at equilibrium form patches of activities around the best matching units. Thus, the learning process, that is modulated by u, consists of a learning at best matching units and their close neighborhood. These patches are also referred as "bubbles" of activity over the neural field.

Kohonen has proposed a competition mechanism based on a differential equation, with a convergence proof. This equation leads to the selection of a single winner unit in the field, and a Gaussian learning kernel around it has then to be applied. Moreover, some extra mechanism has to be added for allowing the setting of new bubbles, as current bubbles are very stable and must be destroyed when new input comes.

Another area of the study of neural field, and certainly one of the most significant, is the continuum neural field theory (CNFT) introduced by Amari [8] and generalized later to the case of bi-dimensional neural fields [9]. These approaches are based on the analysis of differential equations over a continuous 2D field, where units k are the points in the field. The weights are given by a Maxican hat-shaped weight kernel $\omega(|k-l|)$. The dynamics of the neural field are given by equation 1.

$$\frac{du_k}{dt}(t) = -u_k(t) + \int_l \omega(|k-l|) f\,[u_l(t)]\,dl + i_k(t) + h \tag{1}$$

The behavior of the neural field, i.e. its ability to form bubbles and the ability of a bubble to be self-sustained when matchings are reset, depends on the bounds of the primitive $W(r) = \int_{|k|<r} \omega(|k|)dk$, and h. Proofs have been found mainly or the cases of an uniform distribution of the $i_k(t)$.

All these on-center off-surround methods have the advantage of relying on the very distributed nature of the competition processes in neural fields. Thus, they are more suitable for a parallel approach, that is mandatory for large distributed architectures. Moreover, such methods compute local competitions in the neural field, allowing the rising of more than one bubble, which overcomes the winner-takes-all limitation.

The drawbacks of these methods are mainly twofolds. First, the computation of the lateral influence as well as the relaxation process for the $u_k(t)$ is time consuming, and this is why we work on efficient parallel implementations [6]. Second, as equations involve the sum of lateral influences, the dynamics of the competition mechanism are strongly dependent on that sum, as reveals the convergence requirements of the CNFT. Thus, side effects may have dramatic consequences, on the border of the maps for example, where the weight kernel is modified. It can be observed that bubbles rise on corners of the neural field whatever the entries, since those places are much less inhibited than "regular" ones. This is usually avoided in simulations by making the topology of the field torus-like.

3 Designing Competition from "Top" Operator

Current work on multi-map self organization [4,3] has stressed the need of competition algorithms that have the smartness of the CNFT, but that are tractable from a computational point of view, without side effects, since the neural fields may have any shape, and the number of lateral connection may change from a unit to another. Moreover, the bubble formation process has to be robust to the input noise. As our approach is based on the maximum function and non linearities, we are not able to give formal stabilization proofs, but an empirical study of the properties of that competition process is provided here.

3.1 Principle and Properties

One main drawback of the usual neural field competition techniques is the sum operator (or integral) that is sensitive to the actual distribution of lateral connections. In order to free from this sensibility, the mechanism proposed in this paper is rather based on the computation of maxima, whose values are not dependent on the number of elements in the collection they are computed from.

Let us use previously defined notations, and note $\{.\}^{\mathrm{top}\,k}$ the sum of the k highest values of a set. Let us build a neural field of $n \times n$ units, providing separate excitatory and inhibitory connections. The weights are Gaussian, $\omega^+_{ij} = \exp(-(i-j)^2/\sigma^2_+)$ and $\omega^-_{ij} = \exp(-(i-j)^2/\sigma^2_-)$, and weight values under threshold ρ are ignored to save connection resources. At each time step, each unit in the neural field is updated according to equation 2.

$$\Delta u_k(t) \leftarrow \lambda . f\left[\alpha \left\{\omega^+_{kl}.u_l(t)\right\}_l^{\mathrm{top}\,n^+} - \beta \left\{\omega^-_{kl}.u_l(t)\right\}_l^{\mathrm{top}\,n^-} + \gamma i_k(t) - \delta u_k(t)\right] \quad (2)$$

The u values are kept in $[0,1]$ by a supplementary saturation mechanism, and matching rules are supposed to provide values of $i(t)$ in $[0,1]$ as well. This mechanism allows the rising of bubbles at locally best matching places (see. fig. 1-a) as the CNFT does. The sigmoid function $f[x]$ is approximated by a linear functions (see. fig. 1-c) to save computation time. Last, and this is the key point of our approach, the $\{.\}^{\mathrm{top}\,k}$ operator isn't sensitive to the number of connections, as opposed to the sum operator, and the lateral influences do not change the dynamics at the border of the neural field, avoiding the dramatic side effects observed with the CNFT.

3.2 Experiments

First empirical studies are presented here, with the following framework. At each computation cycle, equation 2 is applied to each unit in a random order (asynchronous evaluation). Every 100 cycles, a new distribution $i(t)$ of inputs is forced in the neural field, and kept constant during the 100 next cycles. Measurements are made before changing the $i(t)$ distribution, and 200 measurements are made for each experiments (20000 cycles). The given $i(t)$ distributions are composed of a Gaussian $\mu \exp(-x^2/\sigma^2)$, centered at a random place in the neural field,

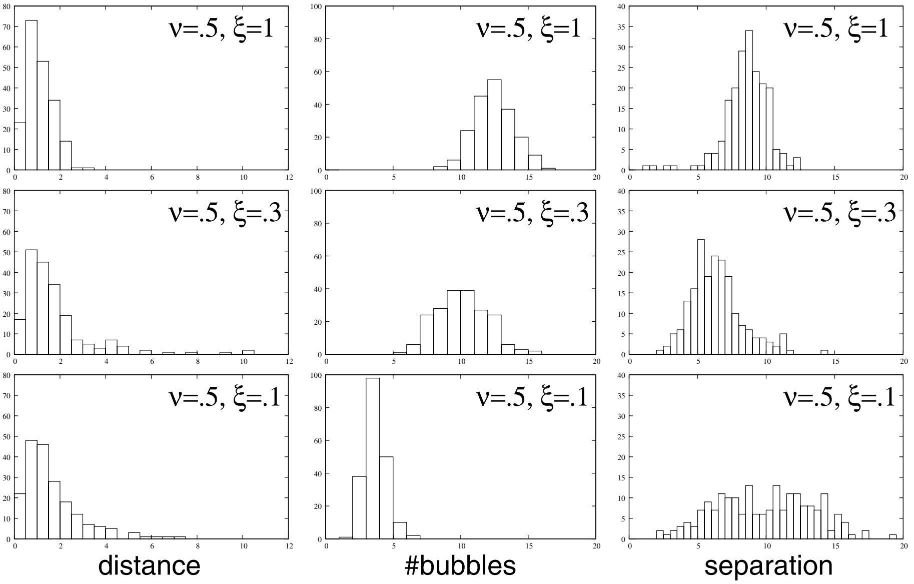


Fig. 2. Experimental results. ν is the noise value, and ξ the connection probability. See text for detail.

with a random amplitude $\mu \in [.5, 1]$ and $\sigma = 2$. A noise is added to this input, by adding *at each place* in the field a random value taken in $[-\nu, \nu]$. Last, robustness to damage in lateral connectivity is tested, since we use a probability ξ to actually create a lateral connection, for both excitatory and inhibitory cases. When damaging connections, bubbles are formed by dense group of peaks, that would be considered as many small close bubbles by the process described in figure 1-b. That is why a diffusion process is added in order to blur $u(t)$ so that group of close peaks are seen as a continuous bubble in our experiments. So the $u'_i(t)$ used for experiments is $u'_i(t) = \max(u_i(t), \rho \max_j u'_j(t))$, where index j describes the four neighbors of unit i in the grid.

Histograms of the following values are given on figure 2. First the distance between the center of the Gaussian and the center of gravity of the closest bubble is measured. This shows how well the bubble represents the significant Gaussian component of the input. Second, the number of bubbles is measured, and third the shortest distance (separation) between center of gravity of bubbles in the neural field. This latter shows the effect of inhibition, that scatters bubbles over the field. It can be seen that several bubbles are allowed, because of noise, but that there is always a patch of activity around the input (distance column). Lateral connection can be damaged severely, thus saving computation time, without preventing bubbles from being well separated. With $\xi = .1$, many units have no excitatory links at all, and fail to raise bubbles from noisy inputs. This is the reason why the number of bubbles decreases, as well as the separation between them increases. All these measurements are made by using the frames defined by figure 1-b to distinguish bubbles in the u' distribution. Experimental values are $\rho = .1, \sigma_+ = 2, \sigma_- = 10, n^+ = 2, n^- = 5, \alpha = .2, \beta = .1, \gamma = .2, \delta = .2, \lambda = .5$, $\theta = .2, \tau = .3, \rho = .6$, and the size of the neural field is 41×41.

4 Discussion

We can see on figure 2 that the mechanism is robust to both high levels of noise in the input activity and intensive connection damage. Robustness can be seen by the results on the "distance" column on figure 2, since in all cases, there is a bubble near the center of the noisy Gaussian. Moreover, the neural field is able to provide several bubbles, allowing the learning at different separate places. Robustness to lateral damage is the cue point, since it allows to manage few connection, which saves most of the computation time. Moreover, this computation of lateral influences is the one that alterates most parallel performances. Last, as the mechanism is insensitive to the actual sum of lateral weights, it can be used in a context where lateral weights are adaptive, as in some recent models of V1 [10], that are to use computational shortcuts to overcome side effects. Many supplementary experimental studies are currently at work to test sensitivity to all parameters, since these first results are encouraging, and promote the use of multi-max operator $\{.\}^{\mathrm{top}\,k}$ for lateral influence in actual computational architectures.

The authors would like to thank the Lorraine region, the Robea program of the CNRS and the European MirrorBot project for their support.

References

1. Burnod, Y.: An adaptive neural network : the cerebral cortex. Masson (1989)
2. Hubel, D.H., Wiesel, T.N.: Functional architecture of macaque monkey visual cortex. Ferrier Lecture Proc. Roy. Soc. London (1977)
3. Ménard, O., Frezza-Buet, H.: Multi-map self-organization for sensorimotor learning: a cortical approach. In: Proc. IJCNN. (2003)
4. Ménard, O., Frezza-Buet, H.: Rewarded multi-modal neuronal self-organization: Example of the arm reaching movement. In: Proc. AISTA. (2004)
5. Kohonen, T.: Self-Organizing Maps. Springer (1997)
6. Vialle, S., Ménard, O., Frezza-Buet, H.: Making cortically-inspired sensorimotor control realistic for robotics: Design of an extended parallel cellular programming model. In: Proc. AISTA. (2004)
7. Willshaw, D., von der Malsburg, C.: How patterned neural connections can be set up by self-organization. Proc. Roy. Soc. London **B 194** (1976)
8. Amari, S.I.: Dynamical study of formation of cortical maps. Biol. Cyb. **27** (1977)
9. Taylor, J.G.: Neural networks for consciousness. Neural Networks **10** (1997)
10. Miikkulainen, R., A.Bednar, J., Choe, T., Sirosh, J.: Self-organization, plasticity, and low-level visual phenomena in a laterally connected map model of the primary visual cortex. Psychology of Learning and Motivation (1996)

Neural Network Computations with Negative Triggering Thresholds

Petro Gopych

V.N. Karazin Kharkiv National University, 4 Svoboda sq., 61077 Kharkiv, Ukraine
pmg@kharkov.com

Abstract. Recent binary signal detection theory and neural network assembly memory model's optimal data-decoding/memory-retrieval algorithm exists simultaneously in functionally equivalent neural network (NN), convolutional, and Hamming distance forms. In present paper this NN algorithm has been specified to provide decoding/retrieval probabilities at both positive and negative neuron triggering thresholds needed, in particular, for ROC curve computations. Examples of intact and damaged NNs are considered, model neuron receptive fields are introduced, a comparison between NN and analytic computations of decoding/retrieval probabilities is also performed.

1 Introduction

Using the energy (or Lyapunov) function approach, J.J.Hopfield [1] has introduced fully interconnected single-layer recurrent neural networks (NN) where autoassociative content addressable memories may exist. They correspond to minima in the network's energy landscape and their main virtue is their ability to restore the previously learned memory vector (trace) from an initial, corrupted or incomplete, binary vector but corresponding NNs with zero-diagonal weight matrices do not ensure that the nearest memory trace is associated to a distorted initial pattern. In a similar way, B.Kosko [2] has demonstrated that Hopfield feedforward associative memory is a special case of more general feedback bidirectional associative memories (BAM) and that every real matrix is a BAM. A bidirectional memory is a two-layer NN and it may correspond to a kind of S.Grossberg's adaptive resonance [3]. As Hopfield NNs and BAMs allow spurious memories, they are not the 'ideal' associative memories.

On the basis of the data coding/decoding algorithm introduced in [4], an optimal binary signal detection theory, BSDT [5], and neural network assembly memory model, NNAMM [6,7], were developed without any optimization procedure. Their data-decoding/memory-retrieval algorithm exists simultaneously in functionally equivalent NN, convolutional, and Hamming distance forms and provides the best decoding/retrieval performance [5-7]. The price paid for the NNAMM optimality is that it places each memory trace in its own assembly memory unit. Such an intact autoassociative NN has no spurious memories and might be considered as the 'ideal' autoassociative memory. The NN algorithm mentioned will be specified and studied for the case of negative neuron triggering thresholds in more details below.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 223 – 228, 2005.

2 Positive Threshold NN Computations

First, let us remind BSDT/NNAMM basic definitions [5-7]: x, N-dimensional vectors with components $x^i = \pm 1$ (the third, zero, component may also exist but it manifests itself only when damaged NNs are considered); x_0, reference vector that represents information stored or that should be stored in an NN; x_r, random vector or binary noise (the signs of its components are randomly chosen with uniform probability, ½); $x(d)$, damaged reference vector with components

$$x_i(d) = \begin{cases} x_0^i, & if \quad u_i = 0, \\ x_r^i, & if \quad u_i = 1 \end{cases} \qquad d = \sum u_i / N, \qquad i = 1,...,N, \tag{1}$$

where d is the damage degree of x_0 and in d magnitudes of marks u_i (0 or 1) may randomly be chosen with uniform probability, ½. For particular $x(d)$, m is the number of marks $u_i = 1$, $d = m/N$, $0 \le d \le 1$; $x(0) = x_0$ and $x(1) = x_\mathrm{r}$; $q = 1 - d$ is a fraction of intact components of x_0 in $x(d)$ or *intensity of cue*, $0 \le q \le 1$; $q + d = 1$, d and q are proper fractions. For a given $d = m/N$, the number of different vectors $x(d)$ is $2^m C^N{}_m$, $C^N{}_m = N!/(N - m)!/m!$; for d ranged $0 \le d \le 1$, complete finite set of all vectors $x(d)$ consists of $\sum 2^m C^N{}_m = 3^N$ items ($m = 0,1,\ldots,N$).

Equation 1 defines the data coding algorithm used [4]. For the decoding of data coded as described, we use a two-layer NN with N McCalloch-Pitts model neurons in its entrance and exit layers; these neurons are linked as in Fig. 1*a*, 'all-inputs-to-all-outputs.' Such NNs are served by vectors $x = x(d)$ [as the set of vectors $x(d)$, consisting of 3^N items, is complete, each vector x may be written as $x = x(d)$].

For a learned NN, its synapse matrix elements, w_{ij}, are defined as

$$w_{ij} = \xi x_0^i x_0^j, \tag{2}$$

where $\xi > 0$ is a parameter ($\xi = 1$ below), $x^i{}_0$ and $x^j{}_0$ are the ith and the jth components of x_0, respectively. w, which is not a zero-diagonal matrix, is defined by vector x_0 and Equation 2 unambiguously. We refer to w as the perfectly learned or the 'ideal' NN and it is of crucial importance that it remembers only *one* pattern x_0. It is also postulated that the NN's input vector $x_\mathrm{in} = x(d)$ is decoded successfully if the learned NN transforms x_in into the NN's output $x_\mathrm{out} = x_0$; an additional 'grandmother' neuron, GN, checks this fact (GN is an integrate-and-fire coincidence neuron responding only to a previously defined precise combination of its inputs, x_0).

The weighted sum, h_j, of all input signals v_i received by the jth exit-layer neuron is

$$h_j = \sum w_{ij} v_i, \quad i = 1,...,N, \tag{3}$$

where $v_i = x^i{}_\mathrm{in}$ is an input/output signal of the ith entrance-layer neuron which plays the role of a fan-out that conveys its input, x_in, to all exit-layer neurons. The sum h_j may be interpreted as a current value of the jth exit-layer neuron's membrane potential or the jth component of the GN receptive field, RF (see Fig. 1).

For the jth exit-layer neuron, its output, $x^j{}_\mathrm{out}$, is produced by the binarization of h_j, using a rectangular response function with the neuron's triggering threshold θ (as one can see in Fig. 1, such a binarization, or spike generation in other words, could take place either in soma of exit-layer neurons or in dendrites of the GN):

$$x^j_{out} = \begin{cases} +1, & if \quad h_j > \theta \\ -1, & if \quad h_j \le \theta, \end{cases} \tag{4}$$

where for $h_j = \theta$ the value $x^j_{\text{out}} = -1$ is arbitrary assigned (cf. Table 1).

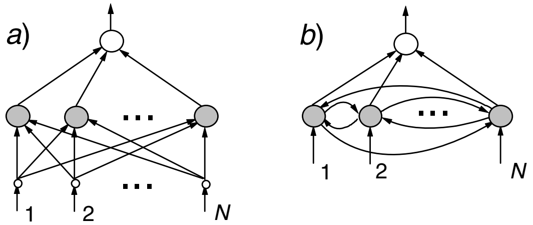


Fig. 1. The NN architecture in the considered (*a*) and in a Hopfield-like (*b*) forms. Designations: small open circles, entrance-layer neurons (fan-outs); large open circle, GN learned to respond to x_0; filled circles, exit-layer neurons; arrows, destinations of spike propagation; *N*, the number of neurons in a layer. Connections (the set of arrows) between entrance- and exit-layer neurons are a milieu where x_0 is stored in the form of a balanced arrangement of weights, $w_{ij} = \pm 1$. If in panel *a* the lengths of vertical arrows go to zero (conserving the original arrangement of all w_{ij}) then a fully interconnected Hopfield-like single-layer architecture will be obtained (panel *b*). That is why, roughly speaking, the NN in panel *a* may be called a 'two-layer Hopfield NN' (as in [7]) and it may be assumed that its entrance-layer neurons specify processes in dendrite arbors of exit-layer neurons in part.

For the *j*th exit-layer neuron, Equations 2 and 3 give: $h_j = \sum w_{ij} x^i_{\text{in}} = x^j_0 \sum x^i_0 x^i_{\text{in}} = x^j_0 Q$ where $Q = \sum x^i_0 x^i_{\text{in}}$ is a convolution of x_0 and x_{in}, $-N \le Q \le N$. Of $h_j = Qx^j_0$ follows that $h = Qx_0$ or $h_j = \pm Q$. Thus, the patterns *h* have either *original* (as the pattern x_0, $h = Qx_0$, $Q \ge 0$) or *reverse* (reverse to x_0, $h = -|Q|x_0$, $Q < 0$) form. For patterns of the original form, the substitution of $h_j = Qx^j_0$ in the first row of Equation 4 gives $Qx^j_0 > \theta$. Consequently, $x^j_{\text{out}} = x^j_0$ if $Q > \theta$ and $\theta > -Q$ (cf. Fig. 2*a*) but any original *h*-pattern (appearing for each possible *Q*, odd and even *N*, or intact and damaged NNs) will only be identified, if $\theta \ge 0$. That means that $x_{\text{out}} = x_0$ and an input x_{in} is decoded (x_0 is extracted) successfully, if $Q(d) > \theta$; $\theta \ge 0$ is Q_0, a threshold of the discrete value of *Q*, and, simultaneously, the neuron's triggering threshold. Hence, for perfectly learned intact NNs, NN and convolutional decoding algorithms are equivalent.

Since $Q(d) > \theta$ and $D = (N - Q)/2$ where $D(d)$ is Hamming distance between x_0 and $x(d)$, the inequality $D < (N - \theta)/2$ is also valid and NN, convolutional, and Hamming decoding algorithms mentioned are equivalent. As Hamming distance decoding algorithm is the best (optimal) in the sense of statistical pattern recognition quality (that is, no other algorithm can better recognize x_0 in x_{in}), NN and convolutional algorithms described are also the best.

The network in Fig. 1*a* is a kernel of an NN/convolutional local feature discrimination (peak identification) algorithm [8]; it is also a part of the NNAMM's assembly memory unit, AMU [7, Fig. 2], where a two-layer NN (a counterpart to real cortical networks) is repeatedly tested by different x_{in} and the result of testing, x_{out}, is verified by the corresponding GN located in additional reference memory (it is, probably, a counterpart to a neuron in the hippocampal regions).

3 Negative Threshold NN Computations

If $\theta = 0$, then for a false alarm (identification of x_0 in noise, x_r) or a free recall (recollection of x_0 initiated by noise, x_r) probability, the NN decoding/retrieval algorithm gives either $F = ½$ (N is odd) or $F = ½ - \Delta F(N)$ (N is even, $\Delta F(N) \to 0$ if $N \to \infty$) and the more the θ the less the F is [6]. As ROCs are functions of F, $0 \le F \le 1$, from above results only a half of an ROC may be plotted. To complete the ROC, let us use the fact that the set of vectors analyzed, $x_{\mathrm{in}} = x(d)$, is finite and contains $n(d) = 2^m C^N{}_m$ items among which there are those producing original as well as reverse patterns h (RFs). For example, in case of determining F ($m = N$, a pure noise analysis), $n(1) = 2^N$. Thus, a complete ROC includes points reflecting recall/recognition probabilities obtained as a result of analysis of those x_{in} that produce both original and reverse RFs and the latter are identified, using negative neuron thresholds. As many cortex neurons may reverse the polarity of their RFs [9], this is a biologically plausible situation (from our consideration follows that for a given cell the form of its RF depends on a reference pattern x_0, the number and arrangement of NN damages, and x_{in}; original and reverse RF patterns are, in general, equally probable).

Table 1. Rules for the binarization of components, h_j, of neuron RFs according to the NN recall/recognition algorithm considered[1]

Index	$\theta \ge 0$	$\theta < 0$	Index
	1	2	
A	$x_{out}^j = \begin{cases} +1, & if \quad h_j > \theta, \\ -1, & if \quad h_j \le \theta \end{cases}$ $Q(d) > \theta,$ $D(d) < (N - \theta)/2$	$x_{out}^j = \begin{cases} +1, & if \quad h_j > 0 \quad or \quad h_j \le \theta \\ -1, & if \quad h_j \le 0 \quad and \quad h_j > \theta \end{cases}$	*a*
		$x_{out}^j = \begin{cases} +1, & if \quad h_j > 0 \quad or \quad h_j < \theta \\ -1, & if \quad h_j \le 0 \quad and \quad h_j \ge \theta \end{cases}$	*b*
B	$x_{out}^j = \begin{cases} +1, & if \quad h_j \ge \theta, \\ -1, & if \quad h_j < \theta \end{cases}$ $Q(d) \ge \theta,$ $D(d) \le (N - \theta)/2$	$x_{out}^j = \begin{cases} +1, & if \quad h_j \ge 0 \quad or \quad h_j \le \theta \\ -1, & if \quad h_j < 0 \quad and \quad h_j > \theta \end{cases}$	*c*
		$x_{out}^j = \begin{cases} +1, & if \quad h_j \ge 0 \quad or \quad h_j < \theta \\ -1, & if \quad h_j < 0 \quad and \quad h_j \ge \theta \end{cases}$	*d*

Table 1 specifies NN binarization rules needed to identify original and reverse RFs, h. $P(\theta)$, the correct decoding probability of inputs x_{in} which is valid for positive as well as negative θ, is defined by the equation $P(\theta) = P_{\mathrm{Or}}(\theta) + P_{\mathrm{Rv}}(\theta)$; $P_{\mathrm{Or}}(\theta)$ and $P_{\mathrm{Rv}}(\theta)$ are decoding probabilities of vectors x_{in} producing original and reverse RFs, respectively. If $\theta \ge 0$, then $P_{\mathrm{Rv}}(\theta) = 0$; if $\theta < 0$, then $P_{\mathrm{Or}}(\theta) = P_{\mathrm{Or}}(0)$. These assertions are illustrated in Fig. 2.

[1] Equation 4, concerning the decoding/retrieval algorithm based on an intact NN, and corresponding convolutional and Hamming algorithms (Section 2) are placed in cell *A*; for other versions of these algorithms see *B*; for damaged NNs, convolutional and Hamming algorithms should separately be derived for each specific arrangement of NN damages; if $h_j = 0$, then Equations *A*, *a*, and *b* may give $x^j{}_{\mathrm{out}} = -1$ while Equations *B*, *c*, and *d* may give $x^j{}_{\mathrm{out}} = 1$.

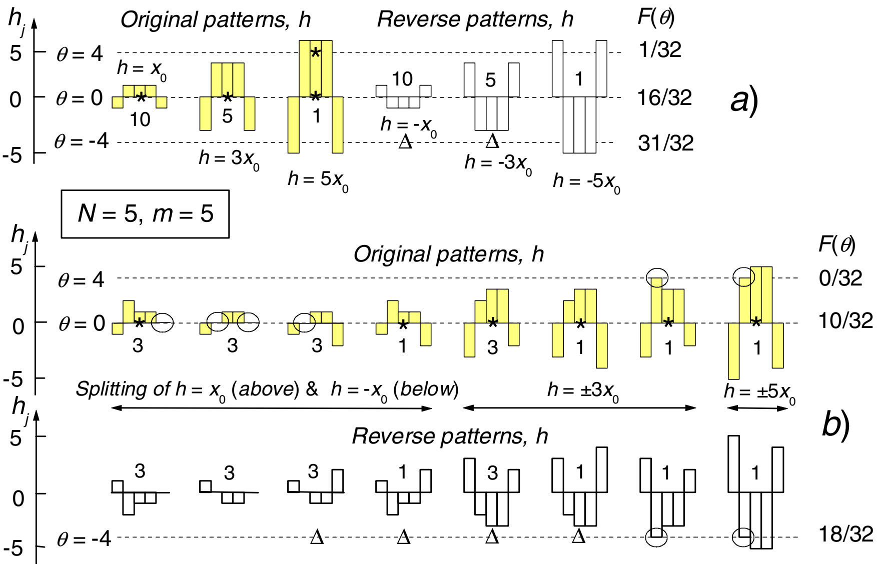


Fig. 2. All possible types of 1D profiles (patterns) of neuron RFs, h, appearing when an intact NN with $N = 5$ (a) or the same one damaged (b) analyzes a complete set of $2^N = 32$ different noise vectors $x_{in} = x_r$ ($m = N = 5$). For the damaged NN, two, (1,2) and (4,5), of its $N^2 = 25$ connections (entrance-layer neuron, exit-layer neuron) are disrupted: $w_{12} = w_{45} = 0$; for the intact NN (a) and the damaged NN (b) their original and reverse RFs are respectively shown as gray and white histograms, $h = (h_1,h_2,h_3,h_4,h_5)$; the NNs and their GNs (Fig. 1) are learned to remember the pattern $x_0 = (–1,1,1,1,–1)$, left-most histogram in panel a; arrows in panel b denote sets of patterns obtained by means of the splitting (due to NN damages) of corresponding h-patterns related to the intact NN (panel a); open circles denote the cases where $h_j = \theta$, the values of θ considered as an example are shown as dashed lines; asterisks and triangles point out patterns identified by Equations A ($\theta \geq 0$) and a ($\theta < 0$) adopted from corresponding cells of Table 1; the digit near each pattern is the number of such patterns in a complete set of them; $F(\theta) = P_{\mathrm{Or}}(\theta) + P_{\mathrm{Rv}}(\theta)$, false-alarm (or free-recall) probability at different values of θ.

4 Comparison of NN and Analytic Computations

NN (Sections 2 and 3) and analytic ([6], Equations 7 and 8) computations of decoding probabilities are compared in Table 2 where names of columns coincide with indices of NN binarization rules in Table 1. Equation A of Table 1 is used at $\theta \geq 0$ and $\theta < 0$; at $\theta < 0$ for intact NNs Equation A together with any of the equations in column 2 of Table 1 gives equivalent results, for damaged NNs it produces results depending on the choice of binarization rules (a, b, c, or d). There are essential distinctions between ROCs found for intact and damaged NNs: the former have a given form not depending on x_0 [5]; for the latter even the number of values of F depends on the NN damage arrangements, reference pattern x_0, and NN binarization rules (e.g., in Table 2 there are columns containing 9 and 10 different values of F, $0 \leq F \leq 20/32$).

Table 2. $F(\theta)$ calculated analytically and by NN algorithms defined in the legend to Fig. 2 ($\theta = Q_0$, neuron thresholds; values of $F(\theta)$ found in the figure are shown in bold face)

Q_0 or θ	Intact NN, $N = 5$		Damaged NN, $N = 5$, $w_{12} = w_{45} = 0$, x_0			
	Analytic, [6]	*A*	*a*	*b*	*c*	*d*
-6	32/32	32/32	20/32	20/32	20/32	20/32
-5	31/32	31/32	19/32	20/32	19/32	20/32
-4	31/32	**31/32**	**18/32**	19/32	18/32	19/32
-3	26/32	26/32	14/32	18/32	14/32	18/32
-2	26/32	26/32	13/32	14/32	10/32	14/32
-1	16/32	16/32	10/32	13/32	10/32	10/32
0	16/32	**16/32**	**10/32**	10/32	10/32	10/32
1	6/32	6/32	6/32	6/32	6/32	6/32
2	6/32	6/32	2/32	2/32	2/32	2/32
3	1/32	1/32	1/32	1/32	1/32	1/32
4	1/32	**1/32**	**0/32**	0/32	0/32	0/32
5	0/32	0/32	0/32	0/32	0/32	0/32

5 Conclusion

For intact and damaged NNs, NN computations of decoding/retrieval probabilities are specified and studied at $\theta < 0$. Using different binarization rules, conditions are found under which such NNs produce ambiguous and unambiguous computation results.

Acknowledgement. I am grateful to HINARI for free on-line accesses to recent full-text journals and to my family and my friends for their help and support.

References

1. Hopfield, J.J.: Neural Networks and Physical Systems with Emergent Collective Computational Abilities. Proc. Nat. Acad. Sci., USA 79 (1982) 2554-2558
2. Kosko, B.: Bidirectional Associative Memories. IEEE Trans. Syst., Man, Cyber. 18 (1988) 49-60
3. Grossberg, S.: How does a Brain Build a Cognitive Code? Psych. Rev. 1 (1980) 1-51
4. Gopych, P.M.: Determination of Memory Performance. JINR Rapid Communications 4[96]-99 (1999) 61-68 (in Russian)
5. Gopych, P.M.: Sensitivity and Bias within the Binary Signal Detection Theory, BSDT. Int. J. Inf. Theor. & Appl. 11 (2004) 318-328
6. Gopych, P.M.: ROC Curves within the Framework of Neural Network Assembly Memory Model: Some Analytic Results. Int. J. Inf. Theor. & Appl. 10 (2003) 189-197
7. Gopych, P.M.: A Neural Network Assembly Memory Model Based on an Optimal Binary Signal Detection Theory. Programming Problems (Kyiv, Ukraine) no.2-3 (2004) 473-479
8. Gopych, P.M.: Identification of Peaks in Line Spectra Using the Algorithm Imitating the Neural Network Operation. Instr. & Exper. Techn. 41 (1998) 341-346
9. DeAngelis, G.C., Ohzava, I., Freeman, R.D.: Receptive-Field Dynamics in the Central Visual Pathways. Trends in Neurosci. 18 (1995) 451-458

A Model for Delay Activity Without Recurrent Excitation

Marc de Kamps

Chair for Robotics and Embedded Systems, Institut für Informatik, TU München,
Boltzmannstrasse 3, D-85748 Garching bei München, Germany
kamps@in.tum.de
http://www6.in.tum.de/~kamps

Abstract. Delay activity (DA) is the increased firing rate of a cortical population, which persists when the stimulus that induced it is removed. It is believed to be the neural substrate for working memory, and as such highly relevant for theories of cognition. The cortex is highly recurrent, mainly excitatory, and finding stable attractors for DA at low firing rates for realistic neuronal parameters has proven to be hard. Most models for DA use recurrent excitation. Here a model with recurrent disinhibition is presented, which is manifestly stable. This model requires a cortical circuit that is slightly more complex than circuits in models using recurrent excitation, but circuits of comparable complexity have been found in cortex. Since delay attractors can not be observed directly, it is important to consider all theoretical possibilities.

1 Introduction

Delay activity (DA) is the increased firing rate with respect to baseline of a population of neurons, which is caused by a stimulus and which persists once the stimulus is removed. DA is believed to be the neural substrate of working memory and therefore a good model of DA is of prime importance for models of higher cognition. Experimental results indicate that DA is in the order of 10-20 Hz, whereas the normal cortical background rate is in the order of 1-10 Hz. Since the cortex is a highly recurrent network, which consists primarily of excitatory neurons, this is remarkable and it turns out to be challenging to create realistic models of cortical dynamics of DA that remain stable at rates which are far below maximum firing rates. The problem was clearly defined by Amit and Brunel [1] and their model produced stable rates for a small stimulus sensitive excitatory population, which was embedded in a larger local pool of excitatory neurons, and which where both controlled by a local inhibitory pool. Delay activity was sustained by a higher potentiation of efficacies between neurons in the stimulus sensitive population, together with the contribution from the non-stimulus specific excitatory background, i.e. by recurrent excitatory feedback. Recently Latham and Nirenberg [2] have shown that the model of Amit and Brunel (and others based on similar modeling assumptions) produces reasonable rates, for biologically plausible neuronal and network parameters, but that these rates are extremely sensitive to the choice

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 229–234, 2005.

of these parameters. To solve this problem Latham and Nirenberg extended the original analyis from [1] considerably, and they went beyond the sparse coding limit. In the sparse coding limit, the stimulus sensitive group of neurons is assumed to make out such a small fraction of the local excitatory background, that it does not influence the background populations significantly. Latham and Nirenberg found that low rate, stable DA is possible beyond the sparse coding limit, i.e. when DA influences the local cortical background rates significantly. Network parameters were chosen such that the local background state is effectively inhibitory and an increase in DA is matched by an increase in the inhibitory background, which ensures the systems stability.

This is a strong assumption: an attractor that corresponds to a specific working memory state may involve many DA populations, and the implication of the model of Latham and Nirenberg is that the local background is increased significantly (and is effectively inhibitory) in the entire cortical area that sustains the attractor. This may actually be what happens, but as Latham and Nirenberg point out, attractors cannot be observed directly and inferences must be made about their existence by comparing experimental data with model predictions. It is therefore important to consider possible alternatives.

In this paper, I will argue that it is possible to implement delay activity be recurrent disinhibition, rather than recurrent excitation and that the only excitatory activity necessary in this model is feedfoward. A cortical circuit that is implemented in this way is manifestly stable at low and plausible firing rates, but is slightly more complex than the ones considered in [1] and [2]. In the next section, I will introduce the model, and in the last section I will discuss the differences between the various models for DA and their implications for experimental data.

2 The Model

The main modeling assumptions are the same as in [1] and [2]: cortical neurons receive a large and unspecific background from remote cortical areas, a local excitatory and a local inhibitory population. The stationary population firing rate of population i is given by:

$$\nu_i = \phi_i(\mu_i, \sigma_i), \tag{1}$$

where:

$$\phi_i(\mu_i, \sigma_i) \equiv \left\{ \tau_{ref,i} + \sqrt{\pi}\tau_i \int_{\frac{V_{reset,i}-\mu_i}{\sigma_i}}^{\frac{\mu_i-\theta_i}{\sigma_i}} du \, [1 + \mathrm{erf}(u)] \, e^{u^2} \right\}^{-1} \tag{2}$$

$$\mu_i = \tau_i \sum_j J_{ij} N_{ij} \nu_j, \tag{3}$$

$$\sigma_i = \sqrt{\sum_j \tau_i J_{ij}^2 N_{ij}}. \tag{4}$$

τ_i, $\tau_{ref,i}$ are the membrane time constant and the absolute refractory period, respectively, in s, θ_i and $V_{reset,i}$ the threshold potential and the reset potential, respectively, in V, all for neurons in population i. N_{ij} is the effective number of neurons from population j seen by a neuron in population i and J_{ij} the effective efficacy from a spike in population j on a neuron in population i in V. These equations form a closed system which can be solved in ν_i. In practice, one does this by introducing a pseudo-dynamics:

$$\tau_i \frac{d\nu_i}{dt} = -\nu_i + \phi(\mu_i, \sigma_i), \tag{5}$$

and selecting initial values $\nu_i(0)$.

The circuit has a structure as shown in Fig. 1. The neuronal, network and connectivity parameters are given in Table 1. Although the number of parameters is quite large, many are already familiar from [1]: g, x, J_{EE}, J_{IE}, J_{EI} and J_{II} are chosen such that an unspecific cortical background rate ν_{ext}, which is input to all populations, is replicated in the local excitatory pool **E**, while the the local inhibitory pool **I** fires at a slightly higher rate. About half of the input of any given neuron comes from the cortical background ($x = 0.5$). **E** and **I** are stimulus insensitive, to a first approximation. A small number of neurons are distinguished by the fact that they receive slightly more potentiated input from the other neurons in **E** and they constitute a subset of **E**, denoted by **DA**. Normally, they would fire at a higher rate than neurons in **E**, but it is assumed that these neurons receive more potentiated input from a subset of **I**, denoted by **SUP** and therefore they will typically fire at the same rate as neurons in **E**. Neurons in **SUP** can be inhibited by neurons in **DIS**, which are typically inhibited rather strongly by **I** and therefore do not influence **SUP** under non-delay conditions. Hence, **SUP** will fire at the same rate as **I**, and **DA** will fire at the same rate as **E**.

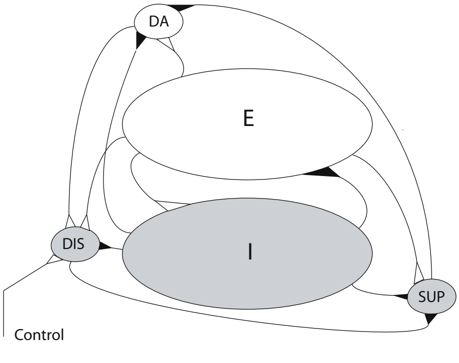


Fig. 1. Local circuit for delay activity. Excitatory populations are white, inhibitory populations are grey. White (black) triangles indicate excitatory (inhibitory efficacies). The relative sizes give a rough indication of the number of neurons involved.

Table 1. The circuit parameters. τ_{exc} is the membrane time constant for the excitatory parameters: E and DA, τ_{inf} for the inhibitory populations: **I**, **SUP** and **DIS**. All populations receive an extra cortical background rate ν_{ext}, with connection parameters $\{xC_E, J_{EE}\}$.

Neuronal parameters						
τ_{exc} = 20 ms, τ_{inh} = 10 ms, τ_{ref} = 2 ms, V_{reset} = 0 mV, θ= 20 mV						
Network parameters						
C_E = 20000, C_I = 2000, x = 0.5, g = 5, ν_{ext} = 3 Hz, ν_E = 3 Hz, ν_I = 5.1 Hz						
x_{DA} = 0.02, γ_{DA} = 1.25, x_{SUP} = 0.02, γ_{SUP} = 3, $x_{SUP,DA}$ = 0.07, $\gamma_{SUP,DA}$ = 3 x_{DIS} = 0.1, γ_{DIS} =2.5, x_C = 0.03, γ_C = 1.9, $x_{DA,DIS}$ = 0.1, $\gamma_{2.2}$, $\gamma_{I,DIS}$ = 2.2						
Connectivity table						
i	**E**	**I**	**DA**	**SUP**	**DIS**	**CONT**
N_{Ei}	$x(1-x_{DA})C_E$	$(1-x_{SUP})C_I$	$xx_{DA}C_E$	$x_{SUP}C_I$	0	0
J_{Ei}	$\theta/193.4$	gJ_{EE}	J_{EE}	J_{EI}	0	0
N_{Ii}	$x(1-x_{DA})C_E$	$(1-x_{SUP})C_I$	$xx_{DA}C_E$	$x_{SUP}C_I$	0	0
J_{Ii}	$\theta/120$	gJ_{IE}	J_{IE}	J_{II}	0	0
N_{DAi}	xC_E	$(1-x_{SUP})C_I$	0	$x_{SUP,DA}C_I$	0	0
J_{DAi}	$\gamma_{DA}J_{EE}$	J_{EI}	0	$\gamma_{SUP,DA}J_{EI}$	0	0
N_{SUPi}	$x(1-x_{DA})C_E$	C_I	$xx_{DA}C_E$	0	$x_{DIS}C_I$	0
J_{SUPi}	J_{IE}	J_{II}	J_{IE}	0	$\gamma_{DIS}J_{II}$	0
N_{DISi}	$C_Ex(1-x_C$ $-x_{DA,DIS})$	C_I	$xx_{DA,DIS}C_E$	0	0	x_CC_E
J_{DISi}	J_{IE}	$\gamma_{I,DIS}J_{II}$	$\gamma_{DA,DIS}J_{IE}$	0	0	γ_CJ_{IE}

Now consider the situation where **DIS** is stimulated rather strongly by an external control signal and therefore will inhibit **SUP**, which results in turn in the disinhibition of **DA**, which responds by a higher firing rate. Moreover, **DA** excites **DIS**. Under no-delay conditions **DA**'s firing rate, which is equal to that of **E**, is not able to overcome the hard inhibition of **I** on **DIS**. But, if it its higher, disinhibited, firing rate is able to keep **DIS** active, even when the control stimulus is removed, then the elevated firing rate of **DA** will persist and is delay activity. This state will only return to the original one if **DIS** will be inhibited again, for instance due to another control signal, or if one if the populations that fire at an elevated rate (**DA, DIS**) is affected by adaptation.

3 Results

In Fig. 2 we see this happen: at t = 0.40 s, the rate of **CONT** is raised by an external stimulus, which lasts for 0.05 s. The result is that for a brief while **DIS** is firing at a high rate (approximately 70 Hz), which inhibits **SUP**. The inhibition of **SUP**, allows the higher potentiated **DA** population to fire at a higher rate of approximately 18 Hz. Its excitation of **DIS** is strong enough to keep it firing at approximately 7 Hz, which is strong enough to keep **SUP** inhibited.

For this mechanism to function, the **SUP** neurons must inhibit the **DA** neurons rather specifically. This is reflected in the relative high values of $x_{SUP,DA}$

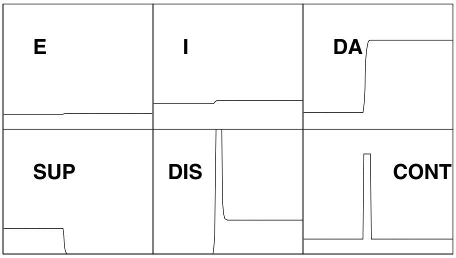


Fig. 2. For each histogram, the horizontal axis represents time t: $0 < t < 1$ s. The vertical axis represents firing rate f, $0 < f < 25$ Hz. The peak in **DIS** briefly extends to 70 Hz.

and γ_{DA}. The values for $x_{SUP,DA}$ and $\gamma_{SUP,DA}$ are chosen such that **DA** fires at rate ν_{ext} in the non-delay condition, i.e. at the same rate as the rest of module **E**. It is assumed that there is not much interaction between the **SUP** and the **I** module. Other than the specific connections from **SUP** to **E** and from **DIS** to **SUP**, there is no distinction between these neurons and neurons of **I**, and **SUP** is assumed to fire at the same background rate as **I**, ν_I in the non-delay condition. **DIS** neurons must inhibit the **SUP** neurons rather specifically, shown by the relatively high values of x_{DIS} and γ_{DIS}. Importantly, the **DA** neurons must not be able to overcome the inhibition on **DIS** at the normal firing rates, but must inhibit **DIS** when firing at delay rates. The values of the other x and γ parameters is uncritical, and might have been taken zero instead. They have been chosen to demonstrate that small interactions between other populations than the ones described above, which are to be expected, do not disrupt the mechanism.

It turns out that parameter space is large and other reasonable values for firing rates can easily be found: for example if one choses $\gamma_{DA} = 1.13$, $\gamma_{SUP,DA} = 1.6$ and $\gamma_{DA,DIS} = 3.5$ one finds delay rate at 10 Hz, rather than at 18 Hz, with the other rates close to the ones shown in Fig. 2. The explanation is simple: the lower value for γ_{DA} leads to a lower firing rate in case **SUP** is inhibited. γ_{SUP} must be decreased, so as to keep **DA** firing at the background rate ν_E in the non-delay condition, and the lower delay firing rate of **DA** must be compensated by an increased potentiation $\gamma_{DA,DIS}$

The parameter space is substantially enlarged by inhibiting **DIS**. This basically decouples **DA** from **DIS** and **SUP** in the non-delay condition. If **DIS** were firing at background rate ν_I, changes in the rate of **DA** will be fed back to **DA** via **DIS** and **SUP** and it becomes harder to find parameters that give a desired delay rate.

4 Discussion

This model uses almost the same modeling assumptions as [1]. Particularly important is the idea that every neuron receives a large number of input spikes, even if it is not directly stimulated and only receives baseline rates from other neurons. I use this idea as well, but in a different way: the values in Table 1 show that a moderately higher potentiation of a relatively small fraction of its input synapses can lead to a firing rate which is significantly higher than baseline activity. Hence, it is possible that spontaneous firing rates significantly above baseline could emerge in such a population, if there is no compensating extra potentiation of its inhibitory inputs. This possibility is crucial for the model described here. A large number of parameters is necessary to describe local pools **E**, **I**, which are firing at stable and low firing rates, but this part of the model is the same as in [1] and [2].

The crucial departure from these models consists in the assumption that there is structure in the **I** population, which functions as a disinhibition circuit. Such disinhibition circuits have been shown to exist in cortex [3]. Although at first sight this model involves a more complicated structure than [1] and [2], it is simpler in dynamical terms. *This is because there is only feedfoward excitation in delay conditions*, and in non-delay conditions the **DA** and **SUP** are integral parts of the **E** and **I** populations, respectively. Moreover, it is not necessary for **DA** neurons to couple to themselves. Finally, the inhibition of **DIS** under non-delay conditions prevents even indirect feedback of **DA** onto itself.

To play a part in cognitive processes, working memory must be controlled: it must be selected, information must be stored into it and retrieved from it, and it is likely that the control of such operations is performed by gating circuits which are very much like the ones described in this model. It is also well known that inhibition plays a role in working memory and that disruptions of its function can result in substantial cognitive impairment. This suggests that inhibition plays a more important role than just rate control.

Earlier experience with large scale cortical modeling [4] has shown that stability of dynamics in local circuits is essential to ensure stability in a large network. This has been the prime motivation for this model, but in the end experiment decides. The predictions of this model for experiments that involve DA are clearly different from [2]: in our model only specific subsets of neurons fire at elevated rates in the delay condition, whereas in [2] the entire area that sustains the attractor is involved. In the former case it should be easy to find neurons that fire at baseline activity, whereas this should be more difficult in the latter.

References

1. Amit, D., Brunel, N. Model of Global Spontaneous Activity and Local Structred Activity During Delay Periods in the Cerebral Cortex. Cerebral Cortex **7** (1997) 237-252
2. Latham, P. E., Nirenberg, S. Computing and Stability in Cortical Networks Neural Computation **16** (2004) 1385-1412
3. Gonchar, Y. and Burkhalter, A. Connectivity of GABAergic Calretinin-immunoreactive Neurons in Rat Primary Visual Cortex. Cerebral Cortex **9** (1999) 683-696
4. van der Velde, F., de Kamps, M. Neural Blackboard Architectures of Combinatorial Structures in Cognition. To appear in: Behavioral and Brain Sciences.

Neuronal Coding Strategies for Two-Alternative Forced Choice Tasks

Erich L. Schulzke and Christian W. Eurich

Universität Bremen, Institut für Theoretische Physik, Otto-Hahn-Allee 1,
D-28359 Bremen, Germany
{erich, eurich}@neuro.uni-bremen.de
http://www.neuro.uni-bremen.de/

Abstract. We investigated the connection between electrophysiological properties of neural populations and their ability to discriminate between the presence of one and two stimuli in a two-alternative forced choice task. The model is based on maximum likelihood estimation in a stimulus space that allows for the presence of multiple stimuli. Repetitive presentation of virtual stimuli yields receiver–operator–characteristics (ROC) curves and psychometric functions from noisy neural responses. For the case of one-dimensional stimuli like the movement direction of a random dot cloud we tested two coding strategies for discriminative ability. It turns out that narrow tuning curves and a variability of tuning widths within the neural population yields a high percentage of correct responses in the simulated psychophysical discrimination task. These results are similar to findings about the localization of single stimuli by neural populations: The examined encoding strategies lead to both an improvement of single stimulus estimation and discrimination between one and two stimuli.

1 Introduction

How does the brain efficiently represent information about the world? It is nowadays accepted that the brain processes information in a distributed way by neural populations, which encode stimulus features such as direction, spatial frequency, or contrast. Much literature has been devoted to determine the representational accuracy of a single stimulus value by a neural population. For example, an estimation-theoretic framework employing Bayes' theorem or Fisher information has proven to be effective for stimulus estimation [7] and investigation of coding strategies that lead to increased representational accuracy. Aspects of these strategies include trial-to-trial variability in the stimulus encoding procedure (e.g. additive, proportional, or multiplicative noise, background activity) [11], correlations in neural firing variability [1,11], width of tuning curves and dimensionality of the stimulus that is being encoded [13]. Furthermore, research has been performed to investigate hyperacuity phenomena in vision [8].

However, stimuli in the real world usually do not consist of only one direction and the analysis in above-mentioned work is tailored for the case that only one stimulus is presented at some time. Multiple stimuli may occur in the receptive field of a neuron, especially in higher processing areas like visual area MT,

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 235–240, 2005.

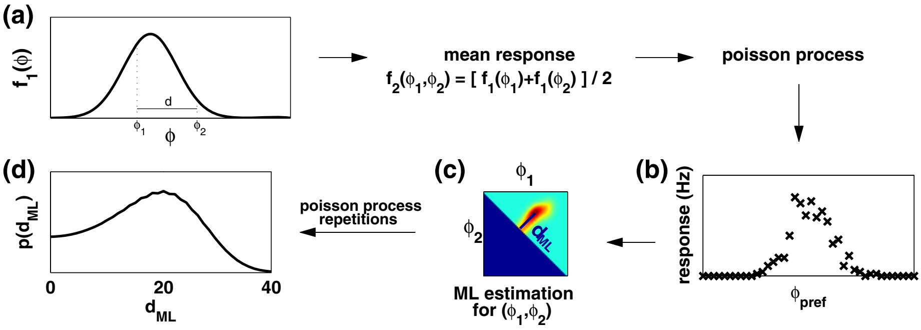


Fig. 1. Encoding and decoding scheme applied in the simulations. The tuning curve of a neuron is depicted in Figure (a). The mean response to two stimuli ϕ_1 and ϕ_2 is given by the mean of the respective tuning curve values. The mean response $f_2(\phi_1, \phi_2)$ drives a poisson process. Figure (b) in the lower right corner shows the response of a neural population to the presentation of (ϕ_1,ϕ_2). The x–axis denotes the preferred direction of the respective neuron. The likelihood for all possible stimuli in a compound stimulus space is calculated (c) and the distance d_{ML} of the maximum to the main diagonal is determined. Repetitions of the poisson process lead to the probability distribution $p(d_{ML})$ in Figure (d).

where receptive field sizes allow for the existence of more than one stimulus at the same time. For example, responses of neurons in macaque visual area MT and psychophysical discrimination performance of humans to moving random dot clouds were measured, where the dots were divided in two subpopulations with different movement directions [9]. In a further experiment Dinse et al. [4] investigated the tactile discrimination performance of humans to stimulation of finger tips for stimulation by one or two needles. How can physiological and psychophysical data be described theoretically, and which physiological constraints influence perception?

To our knowledge, only few theoretical works tried to extend the framework for the estimation of single stimulus values to the presentation of multiple stimuli. Zemel et al. [12] showed that it is possible to adapt the encoding scheme of the Bayesian approach to include multiple stimulus values at the same time and estimate a whole stimulus distribution. Eurich [5] introduced a framework based on Fisher-information in a compound space, where neural responses are described by tuning curves which allow for the presence of two stimuli.

Here we perform a numerical Maximum Likelihood analysis in a compound stimulus space [5] to determine receiver–operator–characteristics (ROC) curves and psychometric functions for the discrimination between the presentation of one and two stimuli.

2 Maximum Likelihood Estimation in a Compound Space

The coding and decoding scheme of our model is depicted in Figure 1. Let us consider a population of N independent neurons that respond selectively to the direction of moving random dot clouds. Every neuron of the population is characterized by its tuning curve $f_1(\phi)$. The subscript denotes that $f_1(\phi)$ is the average response of the neuron to a single stimulus presentation ϕ. Tuning curves are defined as

$$f_1(\phi) = F_{max}\zeta + F_{max}(1-\zeta)\exp\left(\frac{(\phi-\phi_{opt})^2}{2\,\sigma^2}\right), \tag{1}$$

and are functions of the favorite direction ϕ_{opt}, the width of the gaussian σ, the maximal firing rate F_{max} and the baseline ζ $(0 \leq \zeta \leq 1)$. In order to model the mean response of a neuron to the presentation of a stimulus pair (ϕ_1, ϕ_2) we assumed that the mean responses to the single stimulus values are averaged: $f_2(\phi_1, \phi_2) = \frac{1}{2}\left(f_1(\phi_1) + f_1(\phi_2)\right)$. This assumption is taken over from experimental findings in visual areas MT and MST of macaque monkeys [6] and in cat area 17 [10]. The subscript of f_2 now denotes that f_2 is the average neural response to the simultaneous presentation of two stimuli ϕ_1 and ϕ_2. The mean firing rates of the neurons drive poisson processes which generate randomized responses $\boldsymbol{n} = (n_1, \ldots, n_N)$ of a neural population upon the presentation of (ϕ_1, ϕ_2) (see Figure 1b). The probability of observing the response $\boldsymbol{n}$ in the time window T is given by

$$P(\boldsymbol{n}|\phi_1, \phi_2) = \prod_{i_1}^{N} \frac{\left(f_{2,i}(\phi_1, \phi_2)\,T\right)^{n_i}}{n_i!}\, e^{-f_{2,i}(\phi_1,\phi_2)\,T} \tag{2}$$

Now we can determine the stimulus pair $(\phi_{1,ML}, \phi_{2,ML})$, which is most likely to have elicited the response $\boldsymbol{n}$ by using the knowledge about how our model generated the responses to the stimulus: We perform a Maximum Likelihood (ML) estimation in the space of all possible stimulus combinations (ϕ_1, ϕ_2). Note that the case $\phi_1 = \phi_2$ corresponds to the presentation of a single stimulus ϕ. For single stimulus presentations the distance $d_{ML} = \phi_{1,ML} - \phi_{2,ML}$ of the stimulus estimate from the main diagonal in the compound space is determined. By repeating the realization of the poisson process several times we estimate the probability distribution $p(d_{ML})$ for single stimulus presentations. We also determined $p(d_{ML})$ for stimulus pairs (ϕ_1, ϕ_2) with $d = |\phi_1 - \phi_2|$, which are centered around the value of the single stimulus. ROC curves [2] can be calculated from the comparison of the probability distributions for presentation of ϕ (single stimulus) and (ϕ_1, ϕ_2) as a function of d. The informational content of the ROC curves is summarized by the psychometric function, which denotes the fraction of correct responses in a two-alternative forced-choice task.

3 Results

Figure 2 shows likelihood distributions for a neural population of 50 neurons with equally distributed preferred directions ϕ_{opt}, tuning width $\sigma = 30$ deg, maximal firing rate $F_{max} = 40$ Hz, baseline $\zeta = 0$. The likelihood values are color–coded for stimuli (ϕ_1, ϕ_2) with $d = 0, 8, 16, 24$ deg from (a) to (d). The stimulus space is restricted to $\phi_1 \geq \phi_2$. Large likelihood values are shifted from the main diagonal for increasing d.

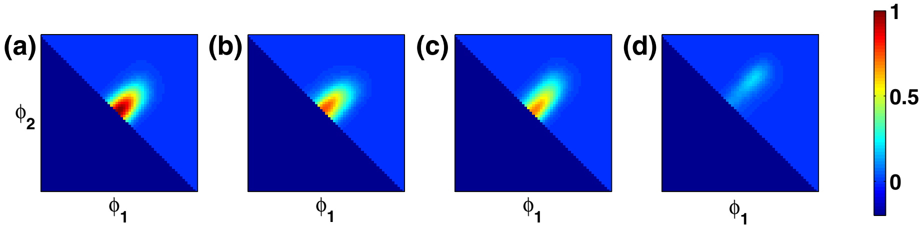


Fig. 2. Maximum likelihood estimation in the compound space spanned by (ϕ_1, ϕ_2). Likelihoods were averaged for 50 repetitions of the Poisson process and rescaled to fit the range [0,1]. The likelihood values are color–coded. In (a) the likelihood estimation is presented for a single stimulus presentation. Likelihood estimates for two stimuli with distance d=8,16,24 deg are shown in (b)–(d), respectively (neural population: N=40, σ=30 deg, ζ=0, F_{max}=40 Hz).

3.1 Discrimination Ability of a Neural Population

In Figure 3a ROC curves are shown for the neural population used for Figure 2. For a single ROC curve the ability to discriminate between a single stimulus ϕ and a stimulus (ϕ_1, ϕ_2) with a fixed d is examined. The two stimulus conditions are represented by their probability distributions $p_1(d_{ML})$ und $p_2(d_{ML})$ (see Fig. 1d). In the ROC curves hit rates are plotted as a function of false alarms, where hit rates are defined as the probability for correct classification of a single stimulus ϕ and false alarms are the probability of classifying a stimulus (ϕ_1, ϕ_2) as single stimulus (for details about the ROC analysis method see [3]). The area below an ROC curve corresponds to the fraction of correct decisions in a two-alternative forced choice task and is plotted in Figure 3b as a function of the distance d of the respective (ϕ_1, ϕ_2) pair. A fraction correct of 0.5 corresponds to random decision. The sample population exhibits a strong increase of classification quality as a function of d.

3.2 Coding Strategies for Discrimination

We used our framework to test different coding strategies. First we examined the impact of the tuning curve baseline ζ on psychometric functions. The parameters of the neural population were $N = 180$, $F_{max} = 40$ Hz, $\sigma = 30$ deg. ζ was varied in the range $0 - 0.5$. The result depicted in Figure 4a shows that the baseline has a negative impact on classification performance and that this influence is considerably large: The psychometric function for $\zeta = 0.5$ (solid grey line) is below the light grey dotted line, which denotes the function for a population with $N = 20$ and $\zeta = 0$, where other parameters have not been altered. Figure 4b shows the dependency of psychometric functions on the tuning width σ ($N = 180$, $F_{max} = 40$ Hz, $\zeta = 0$). Obviously, narrow tuning curves are better suited for the discrimination task. In a third examination we tested the influence of jittered tuning curve widths on classificaton performance. The widths σ of the tuning curves were varied randomly around a mean value σ_{mean} according to a gaussian distribution with standard deviation σ_j. Figure 4c shows that the stronger the jitter the better the discriminative performance.

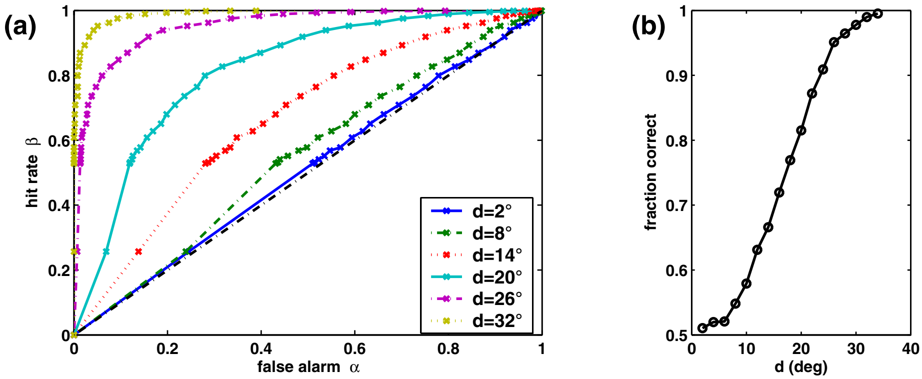


Fig. 3. (a) ROC curves for the population used in Fig.2. Each ROC curve plots hit rates as a function of false alarms for the discrimination between a single stimulus ϕ and two contemporaneously presented stimuli (ϕ_1, ϕ_2). Different curves result from different distances d. **(b)** Psychometric function resulting from the ROC curves in (a): Every point of the function denotes the area below an ROC curve.

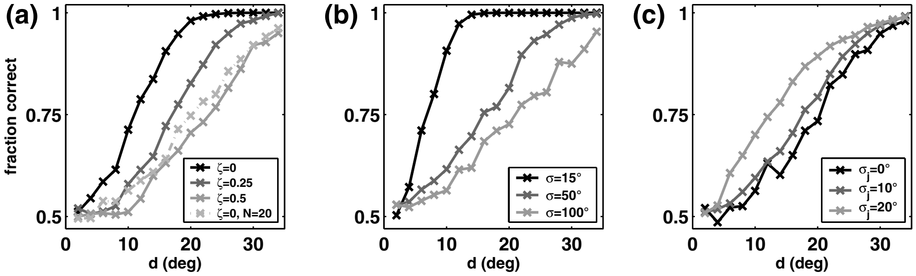


Fig. 4. Influence of different parameters in the encoding scheme on psychometric functions. **(a)** shows the influence of baseline firing rate of the neurons. The light–colored dotted line shows the psychometric function for a population of N=20 neurons, while the solid curves were calculated for N=180. Note that introducing a baseline of 20 Hz $(\zeta = 0.5)$ and reducing the neural population from N=180 to N=20 results in similar impairment of discriminative ability. Figure **(b)** shows the influence of tuning width on discrimination: Narrow tuning curves perform better than large ones. Figure **(c)** shows the influence of jitter of tuning width around a fixed mean value: Strong jitter performs better than uniform tuning width.

4 Discussion

We examined the discrimination between the presence of one or two stimuli at the same time by a neural population in a model which employs Maximum Likelihood estimation in a compound space (ϕ_1, ϕ_2). The model proved to be well–suited for the topic in combination with an ROC analysis. We found that discriminative ability is strongly impaired by an increase of the baseline ζ of the tuning curves: For $\zeta = 0.5$ the discrimination was worse than obtained by reducing the number of coding neurons from 180 to 20. We also found that narrow tuning curves are better suited for discrimination than large ones. Jitter of tuning width within the population also leads to better classification performance. These results are all in perfect accordance to the findings for the localization of single stimuli [11], indicating that the same coding strategies lead to an improvement for both stimulus estimation and discrimination between one and two stimuli. More coding strategies can be tested as the influence of stimulus dimension, noise correlations, and different noise models. It would also be interesting to find a link between our numerical analysis and results obtained with Fisher information for the same compound space [5]. Finally, our model extends the framework for single stimulus estimation, but is subjected to a similar constraint by assuming a fixed number of stimuli in the analysis. For the general case, where the number of stimuli is not fixed, reconstructions of a stimulus distribution can be performed as done in [12]. In that case the examination of coding strategies has to be adopted for the stimulus distribution estimation method resulting in other measures for discriminability.

References

1. Abbott, L.F., Dayan, P.: The effect of correlated variability on the accuracy of a population code. Neural Comp. **11** (1999) 91-101.
2. Britten, K.H., Shadlen, M., Newsome, W.T., Movshon, J.A.: The analysis of visual motion: A comparison of neuronal and psychophysical performance. J. Neurosci. **12** (1992) 4745-4765.
3. Dayan, P., Abbott, L.F.: Theoretical neuroscience: Computational and mathematical modeling of neural systems. MIT Press, MA (2001).
4. Dinse H.R., Kreikemeier, K., van der Berg, I., Böhmer, G., Wiemer, J.: Manipulating input correlations: complementary forms of cortical map reorganization and perceptual learning. Soc. Neurosci. Abstr. 28 (2002) 840.3.
5. Eurich, C.W.: An estimation-theoretic framework for the presentation of multiple stimuli. In: Adv. Neur.Inform.Process. **15**, Becker, S., Thrun, S., and Obermayer,K. (eds.) MIT Press, MA, (2003) 293-300.
6. Recanzone, G.H., Wurtz, R.H., Schwarz U.: Responses of MT and MST neurons to one and two moving objects in the receptive field. J. Neurophysiol. **78** (1997) 2904-2915.
7. Seung, H.S., Sompolinsky, H.: Simple models for reading neuronal population codes. PNAS **90** (1993) 10749-10753.
8. Snippe, H.P., Koenderink, J.J.: Discrimination thresholds for channel-coded systems. Biol. Cybern. **66** (1992) 543-551.
9. Treue, S., Hol, K., Rauber H.J.: Seeing multiple directions of motion–physiology and psychophysics. Nat. Neurosci. **3** (2000) 270-276.
10. van Wezel, R.J.A., Lankheet, M.J.M., Verstraten, F.A.J., Maree A.F.M., van de Grind, W.A.: Responses of complex cells in area 17 of the cat to bi-vectorial transparent motion. Vis. Res. **36** (1996) 2805-2813.
11. Wilke, S., Eurich, C.W.: Representational accuracy of stochastic neural populations. Neural Comp. **14** (2001) 155-189.
12. Zemel, R.S., Dayan, P., Pouget, A.: Probabilistic interpretation of population codes. Neural Comp. **10** (1998) 403-430.
13. Zhang, K., Sejnowski, T.J.: Neural tuning: To sharpen or broaden? Neural Comp. **11** (1999) 75-84.

Learning Features of Intermediate Complexity for the Recognition of Biological Motion

Rodrigo Sigala[1,2], Thomas Serre[2], Tomaso Poggio[2], and Martin Giese[1]

[1] Laboratory for Action Representation and Learning (ARL), Dept. of Cognitive Neurology, University Clinic Tübingen, Schaffhausenstr. 113, D-72072 Tübingen, Germany
Rodrigo.Sigala@tuebingen.mpg.de, Martin.Giese@uni-tuebingen.de
[2] McGovern Institute for Brain Research, Brain and Cognitive Sciences, Massachusetts Institute of Technology, Bldg.E25-201, 45 Carleton St., Cambridge, MA 02142, USA
serre@mit.edu, poggio@ai.mit.edu

Abstract. Humans can recognize biological motion from strongly impoverished stimuli, like point-light displays. Although the neural mechanism underlying this robust perceptual process have not yet been clarified, one possible explanation is that the visual system extracts specific motion features that are suitable for the robust recognition of both normal and degraded stimuli. We present a neural model for biological motion recognition that learns robust mid-level motion features in an unsupervised way using a neurally plausible memory-trace learning rule. Optimal mid-level features were learnt from image motion sequences containing a walker with, or without background motion clutter. After learning of the motion features, the detection performance of the model substantially increases, in particular in presence of clutter. The learned mid-level motion features are characterized by horizontal opponent motion, where this feature type arises more frequently for the training stimuli without motion clutter. The learned features are consistent with recent psychophysical data that indicates that opponent motion might be critical for the detection of point light walkers.

1 Introduction

Humans can recognize biological motion (e.g. human actions like walking and running) accurately and robustly; even from stimuli consisting only of a small number of illuminated dots that move like the joints of a human actor ("point light walkers") [6]. The neural mechanism that underlies the robust generalization from normal to point-light stimuli remains largely unclear. A possible explanation is that the brain extracts specific motion features that are shared by both stimuli classes. The nature of these features is unknown, and it has been discussed whether they are based predominantly on motion or form information [7]. In a recent study, combining methods from image statistics and psychophysical experiments, it was shown that robust recognition can be accomplished based on mid-level motion features [2].

Neurophysiological studies in monkeys and imaging studies in humans suggest that the perception of biological movements and actions involves both the ventral and the dorsal visual processing stream (see [5] for a review). A recent computational model

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 241 – 246, 2005.

for biological motion recognition tries to account for a variety of the existing experimental data using relatively simple physiologically-plausible mechanisms [5]. The model is based on a feed-forward architecture which has been derived by extending a "standard model" (SM) for the recognition of stationary objects in the visual cortex [8]. Like other models for object recognition in the cortex [4, 8], our model represents complex movements in terms of learned prototypical patterns that are encoded by model neurons that respond to complex body shapes.

We apply in this paper a new biologically inspired algorithm, the "Memory Trace" (MeT) learning rule, to optimize model mid-level features for motion recognition. Originally the MeT algorithm was devised for the learning of mid-level in the context of the SM [9]. It has been demonstrated that by application of this learning algorithm the detection performance of the model for real-world stimuli could be substantially improved, resulting in performance levels which exceed the ones of several state-of-the-art computer vision systems for object detection [9]. Here we use the MeT algorithm in the context of a model for the recognition of biological movements and actions in order to optimize mid-level motion features for the detection of walkers.

Our paper first describes the model and the learning algorithm. We then present the results for the detection of walkers and show that learning of optimized mid-level motion features improves the performance, in particular in presence of motion clutter.

2 Methods

2.1 Model for Biological Motion Recognition

Our model corresponds to the motion pathway of the model in [5]. It consists of a hierarchy of neural detectors that are selective for motion features with different levels of complexity (fig. 1a). The first level of the model is formed by local motion energy-detectors whose responses are derived from the optic-flow fields of the stimuli assuming physiologically plausible tuning characteristics (see [5]). The model contains detectors for 70 x 43 different spatial positions and for 4 different directions. It turned out that for the feature learning it is critical that the outputs of the motion energy units are temporally smooth. We assume a simple linear low-pass filter with a time constant of τ = 228 ms, corresponding to the differential equation $\tau \dot{u}(t) = r(t) - u(t)$, where $r(t)$ is the motion energy signal and $u(t)$ the detector output. In the second layer, motion simple (MS2) units encode prototypical motion features of intermediate complexity. They combine the responses of the motion energy detectors with different direction preferences on the previous layer within a limited spatial region. These neurons are modeled by Gaussian radial basis function (RBF) units. The centers $\mathbf{c}_k$ of these RBFs are determined by the MeT algorithm. The responses of these neural detectors depend on the similarity of the local motion energy patterns from the present stimulus, that is given by the vectors $\mathbf{e}_k$, and these learned centers through the relationship: $x_k = \exp(|\mathbf{e}_k - \mathbf{c}_k|^2 / 2\sigma^2)$. For modeling position-invariant recognition, each mid-level motion detector is realized multiple times centered at different random spatial locations. Motion complex (MC2) units pool the responses of all mid-level motion detectors of the same type within a limited spatial receptive field using a MAXIMUM operation. The responses of these units are partially position-invariant. They define the input of a classifier that detects the presence or absence of a walker in the stimulus sequence. We tested different types of classifiers (cf. section 2.4).

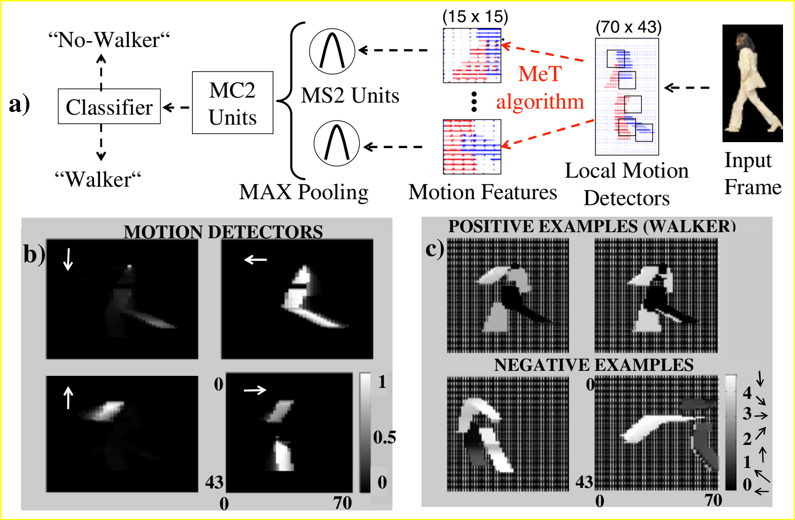


Fig. 1. a) Illustration of our model. See text for explanation. b) Activations of the local motion detectors tuned to 4 different directions (white arrows) for a walking frame shown as grey-coded maps. c) Optic-flows, with directions encoded as grey levels, for two positive (top) and two negative (bottom) examples. Zero motion energy is encoded by the dotted background (white dots on black).

2.2 "Walker-Detection" Task

The performance of our system was evaluated using a walker detection task. We used stimuli with uniform background, and with motion clutter. Stimuli were generated from five actors whose joint trajectories were tracked from videos (one gait cycle with 42 frames) [5]. The walking sequences of five different actors were used as positive examples, and other human actions (e.g. running, boxing, jumping) as negative examples. We selected randomly different sets of these sequences for training and testing the system. To introduce motion clutter for the same stimuli we added 100 moving squares (3x3) at random positions in each stimulus frame, defining random motion with uniform distribution of motion energy over the different directions.

2.3 Feature-Learning with the "Memory Trace" (MeT) Algorithm

Motion features with intermediate complexity were learnt using the MeT algorithm [9] (cf. Fig. 1a). The MeT algorithm is a biologically inspired mechanism for the unsupervised learning of frequently occurring features. The algorithm is inspired by previous work [3] that exploits a simple trace rule for the learning of shift invariance. Our trace rule assumes that the MS2 units keep record of their recent synaptic activity by an internal memory trace signal. In addition, it is assumed that the different features compete for the activations that a given stimulus produces. Successful activation of a feature results in an increase of its memory trace signal. Otherwise, the trace signal decays. Features whose memory trace falls below a fixed threshold are eliminated, and replaced by new features. New features are generated by choosing a randomly positioned local region in the visual field and taking the outputs of the motion energy detectors within this region for the present stimulus as feature vector. (See [10] for details). Learning is online since new features can be selected for each training step.

2.4 Classification Stage

To test the validity of the learned mid-level features for the detection of biological movements, we classified the outputs of the MC2 layer using different types of classifiers: 1) The "Maximally Activated Unit" (MAU) classifier that is biologically plausible. It corresponds to a radial basis function unit whose center is trained with the output signals from the MC2 level for the learned movements. If the activation of this unit is higher than a particular threshold the stimulus is classified as the particular action. Otherwise the classification result is negative. 2) k-Nearest Neighbor (k-NN), a standard technique for classification, was also implemented using RBF units whose centers were learned in the same way as the centers of the MAU classifier. During classification, the label of a test example is set to the label of the majority of the k nearest neighbors of the training set (we tested for $k = 1$ and $k = 5$). 3) Support Vector Machine (SVM) classifiers [13], as used in many recent machine vision systems (e.g. [9]). Although SVMs are not biologically plausible, they provide a typically well-performing classification back-end, which is useful to derive a measure for the quality of the learned features.

3 Results

Performances (Area Under the Receiver Operator Characteristic (ROC) curve) for all classifiers.are shown in Table 1 using the MeT algorithm (*MeT*) without and with motion clutter in the background (*Clutter*). For comparison we also show the results for stimuli in motion clutter when the mid-level features were defined by selecting randomly positioned regions from the stimuli (*Rand*)[1].

Fig. 2 (I) and (II) show the "best" features for the walker detection task for the simulations without and with motion clutter. An important observation is that many of these best features are characterized by horizontal opponent motion. The ROC curves for the three test conditions are shown in Fig. 2 (III). Performance after training with the MeT rule without clutter is almost perfect. This is not only true for state-of-the-art classifiers but also for simpler classifiers such as NN and MAU. Even in presence of clutter the MeT rule is significantly better than for randomly selected features. (The 5-NN outperforms SVM classifier, probably due to overfitting). This robust performance is consistent with recent results from the shape pathway [9]. It suggests a key

[1] Since we were interested mainly in recognition with cluttered background, we did not compare the MeT algorithm with a random selection of features for the other case.

role for plasticity in intermediate and higher visual areas of cortex for the realization of robust recognition.

Table 1. Performances (Area under the ROC) of the system for walker detection. Bold numbers indicate the classifier that gives the best performance for each experiment

Performances (Area Under the ROC)				
Mode	**MAU**	**SVM**	**k-NN (k=1)**	**k-NN (k=5)**
MeT	0.977	0.999	**0.981**	0.962
MeT + Clutter	0.869	0.912	0.957	**0.972**
Rand + Clutter	0.726	0.795	0.876	**0.890**

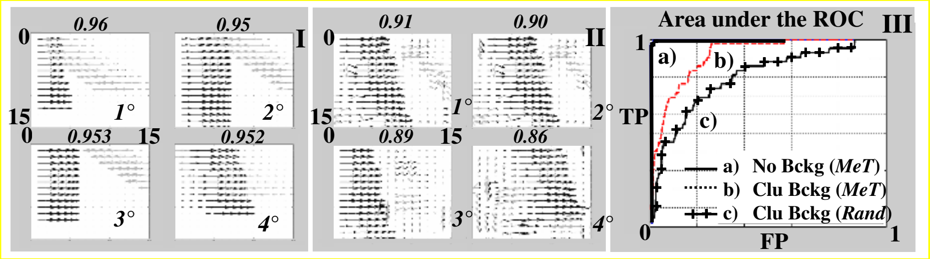


Fig. 2. I) Best four features for the stimuli without and with (II) clutter. Features were ranked according to the area under the ROC (numbers on top) computed separately for each individual feature. Features are plotted as optic-flow fields over the corresponding spatial windows. Black arrows indicate the values for local motion detectors that are selective for motion from left to right, and grey arrows indicate detectors selective for motion in opposite direction. The arrow length indicates the corresponding detector activation. III) ROC curves for the system with SVM classifier, for the MeT algorithm without (a) and with clutter (b), and for random selection of features (c).

4 Discussion

We have presented simulations using local learning rule for the optimization of mid-level motion features in a hierarchical model for the recognition of biological movements. The most important contribution of this rule compared to other approaches (e.g. [11]) relies in its neural plausibility. We found that learning of optimized mid-level features substantially improves the performance of the model, in particular in presence of motion clutter. Similar results have been obtained with a model for shape processing in the ventral pathway using the same learning rule. This suggests a key role of visual experience and plasticity throughout the whole visual cortex. Further work in this direction should implement neurally plausible mechanisms for the classification stage.

In addition, we found that for the detection of walkers, our algorithm learned optimized motion features that are characterized by horizontal opponent motion, for training with and without motion clutter. In principle, the same technique could be applied to optimize form features for the recognition of biological movements from body postures [2]. The importance of opponent motion features seems to be supported by psychophysical an imaging results that show that opponent horizontal motion might be a critical feature for the recognition of walkers, and degraded point light stimuli. Electrophysiological experiments indicate the existence of opponent motion-selective neurons, e.g. in monkey areas MT and MST [1, 12].

Acknowledgements. We are grateful to A. Casile for his support with the simulations. ARL is supported by the deutsche Volkswagenstiftung, DFG SFB 550, and HFSP. CBCL is sponsored by: Office of Naval Research (DARPA) Contract No. MDA972-04-1-0037, Office of Naval Research (DARPA) Contract No. N00014-02-1-0915, National Science Foundation (ITR/IM) Contract No. IIS-0085836, National Science Foundation (ITR/SYS) Contract No. IIS-0112991, National Science Foundation (ITR) Contract No. IIS-0209289, National Science Foundation-NIH (CRCNS) Contract No. EIA-0218693, National Science Foundation-NIH (CRCNS) Contract No. EIA-0218506, and National Institutes of Health (Conte) Contract No. 1 P20 MH66239-01A1.

References

1. Born, R. T. (2000). Center-surround interactions in the middle temporal visual area of the owl monkey. J. Neurophysiol. 84, 2658–2669.
2. Casile A, Giese M. (2005) Critical features for the recognition of biological motion, Journal of Vision, 5, 348-360.
3. Földiak, P. (1991). Learning invariance from transformation sequences, Neural Computation, vol. 3, pp. 194-200, 1991.
4. Fukushima, K.(1980) Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol.Cybern.36, 193–202.
5. Giese M A, Poggio T (2003): Neural mechanisms for the recognition of biological movements and action. Nature Reviews Neuroscience 4, 179-192.
6. Johansson, G. (1973) Visual perception of biological motion and a model for its analysis. Perc. Psychophys. 14, 201-211.
7. Mather, G., Radford, K., & West, S. (1992). Low-level visual processing of biological motion. Proc. R. Soc. Lon. B, 249(1325), 149-155.
8. Riesenhuber, M. & Poggio, T. (1999) Hierarchical models for object recognition in cortex. Nat. Neuroscience 2, 1019-1025.
9. Serre, T. & Poggio, T.(2005). Learning a vocabulary of shape-components in visual cortex, in prep.
10. Sigala, R. (2005) A Neural Mechanism to Learn Features of Intermediate Complexity in the Form and Motion Visual Pathway of Cortex. Thesis MSc. in Neural and Behavioural Sciences, MPI International Research School, Tuebingen, Germany.
11. Song, Y., Goncalves, L. & Perona, P. (2001) Unsupervised Learning of Human Motion Models. Advances in Neural Information Processing Systems 14, Vancouver, Cannada.
12. Tanaka, K., Fukuda, Y. & Saito, H. (1989). Analysis of motion of the visual field by direction, expansion/contraction, and rotation cells clustered in the dorsal part of the medial superior temporal area of the macaque monkey. J. Neurophysiol. 62, 626–641.
13. Vapnik, V. (1998) Statistical Learning Theory. John Wiley and Sons, New York, 1998.

Study of Nitric Oxide Effect in the Hebbian Learning: Towards a Diffusive Hebb's Law

C.P. Suárez Araujo[1], P. Fernández López[1],
P. García Báez[2], and J. Regidor García[1]

[1] Institute for Cybernetics, University of Las Palmas de Gran Canaria, Spain
{cpsuarez, pfernandez}@dis.ulpgc.es, jregidor@dmor.ulpgc.es
[2] Dept. of Statistics, Operations Research and Computation,
University of la Laguna, Spain

Abstract. The Computational Neuroscience has as main goal the understanding of the computational style of the brain and developing artificial systems with brain capabilities. Our paper belongs to this field. We will use an Hebbian neural ensemble which follow a non-linear differential equation system namely Hebbian System (HS), which represent the neurodynamics and the adaptation in accordance with the Hebb's postulate, to study the influence of the NO diffusion in the Hebbian learning. Considering that the postsynaptic neurons provide retrograde signals to the presynaptic neurons [1] we suggest the NO as a probable biological support to the Hebb's law propounding a new mathematical formulation of that learning law, the diffusive Hebb's law. We will present a study of the behavior of the diffusive Hebb's law using a Diffusive Hebbian System (DHS).

1 Introduction

The notion that underlying mechanisms of the learning processes are based in some modifications at cellular level, which depend on correlation activities, dates back to neural habit law [2]. Afterward, Tanzi [3] and Ramon y Cajal [4] were whom suggested that learning arises from synaptic change. This conception precedes both biological learning and the neural network models of learning. In both cases it is necessary to establish exact connections between the neurons which are participating in the process. Later, Hebb gave the first explicit statement of the physiological learning rule for synaptic modification in 1949 [5].

The main objective of our work is studying the influence of the NO diffusion in the Hebbian learning and propounding a new mathematical formulation of Hebb's law. We suggest the presence of the retrograde cellular messenger NO as a probable biological support to the Hebb learning law. NO is a free radical gas, which once synthesized, it freely diffuses through membranes affecting all neighbouring cells, including the pre-synaptic one and the cell that released it, and having an influence on the long-term potentation (LTP) [6] and in the formation of long-term memory (LTM) [8]. Because of this we must take into account a new conception of the meaning of the Hebb law; it is not a correlation's postulate any more.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 247–253, 2005.

We will work with Hebbian neural ensembles which follow a general mathematical scheme corresponding to non-linear differential equation systems namely *Hebbian Systems* (HS) and an extension, the *Diffusive Hebbian Systems* (DHS), which represent the neurodynamics and the adaptation in accordance with the Hebb's postulate [5]. We study the dynamics of these systems, in rest state and when they receive constant inputs, using bifurcation theory. Our developments belong to *Global Study Framework of Diffusion messenger NO* (GSFNO) specifically in the *Theoretical Framework* (TF) [6].

2 The Nitric Oxide Effect in the Hebbian Learning

2.1 Neural System Model

The study of the underlying mechanisms in the learning processes, in *Biological and Artificial Neural Network* (BNN/ANN) have been performed using a model of simplified *Neural System* (NS) which consists of monodimensional Hebbian neural ensembles, Fig. 1a. The pre-synaptic ensembles with activation states $\{\mathbf{S}^A\}$ and inputs $\{\mathbf{X}^A\}$, and the postsynaptic ensembles with activation states $\{\mathbf{S}^B\}$, $\{\mathbf{S}^C\}$, $\{\mathbf{S}^D\}$. The strengths of wire connections and lateral interactions are $\{\mathbf{W}^{AB}\}$ and $\{\mathbf{U}^{CB}\}$, respectively. The wireless connections are the virtual (diffusive) weights $\{\mathbf{V}^{DB}\}$, which establish the *Diffuse Neighbourhood* (DNB) of neurons [6], which synthesize NO, in the post-synaptic side. We are considering a 2-nearest neighbours DNB.

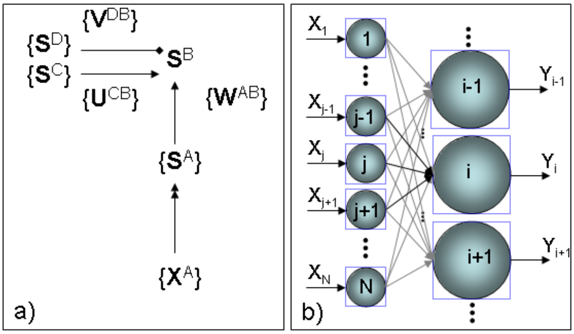


Fig. 1. a) Neural System Model (NS). b) Neural Micro-surround used in the study.

The neurodynamics and the learning processing of this system, without lateral interactions, are mathematically represented by *Hebbian System*, Eq. (1), in accordance with the Hebb's postulate.

$$\begin{aligned} \dot{S}^B &= f(S^B, S^A, W^{AB}) \\ \dot{W}^{AB} &= h(W^{AB}, S^A, S^B) \end{aligned} \tag{1}$$

We propose an extension of this HS, the *Difussive Hebbian System* (DHS), introducing the NO effect, Eq. (2).

$$\begin{gathered}\dot{S}^B = f(S^B, S^A, W^{AB}) + g(S^B, S^C, V^{DB}) \\ \dot{W}^{AB} = h(W^{AB}, S^A, S^B, \varDelta C^{BA}, \varDelta C^{BD}) \\ \dot{V}^{DB} = \theta(V^{DB}, S^A, S^B, C^B) \\ \dot{C}^B = -\lambda_{C^B} C^B - D_B \sum_i \varDelta C_i^{BD} - D_{BA} \varDelta C_i^{BA} + F\end{gathered} \tag{2}$$

In this paper we will dedicate our efforts to study the Hebbian learning processing in the DHS, postponing the analysis of a complete DHS for future works.

2.2 Analysis of the Diffusive Hebb's Law

The dynamics of weights for the Hebbian systems responds to the Hebb's law [5]. There is not direct evidence of the cause of the postulated change of the efficiency between two connected cells, A and B. We propose as a probable biological support to the Hebb's law the presence of a retrograde cellular messenger NO. Our developments try to show that the *Volume Transmission* (VT), by means of NO diffusion, is capable to transport the appropiate information from post-synaptic to pre-synaptic neuron cooperating in the emergence of Hebbian learning. Therefore, the NO may underlie a form of non-local, non-synapse-specific learning and a new non-correlation character of Hebb's law. This new conception of the Hebb's rule implies a reformulation for its general mathematical expression, where the gradient of NO concentration between post- and pre-synaptic sides must be considered as a new variable, with an important role in the learning process. We started from the generalized Hebbian algorithm activity product rule with a quadratic weight dynamics [2], [6]. We will study the NO effect not only as a neuromodulatory influence [7], [9], but embodiments the information from post-synaptic side as a function of the gradient of NO concentration between post- and pre-synaptic side, Eq. (3). and considering that it is not a correlation expression and it would represent a non-synapse-specific learning.

$$\dot{W}^{AB} = c[\varDelta C^{BA}]\{\alpha\phi[S^A] - W^{AB}\} \tag{3}$$

Where α is the learning rate and $\phi[S^A]$ is a function of the pre-synaptic neuron activation. We have empirically obtained a simple and effective $\varDelta \mathbf{C}^{BA}$ dependence for the function $c[\varDelta C^{BA}]$, Eq. (4). In this way the final expression for diffusive Hebb's law is given in Eq. (5).

$$c[\varDelta C^{BA}] = \varDelta C^{BA} / \varDelta C^{BA}_{max} \tag{4}$$

$$\dot{W}^{AB} = \{\varDelta C^{BA} / \varDelta C^{BA}_{max}\}\{\alpha\phi[S^A] - W^{AB}\} \tag{5}$$

Where $\varDelta \mathbf{C}^{BA}_{max}$ is a constant with the maximum value which is reached for the gradient during the whole process of weight modification.

For studying the behavior of the diffusive Hebb's law we define our model of NS containing the pre- and post-synaptic neuron ensembles with five neurons

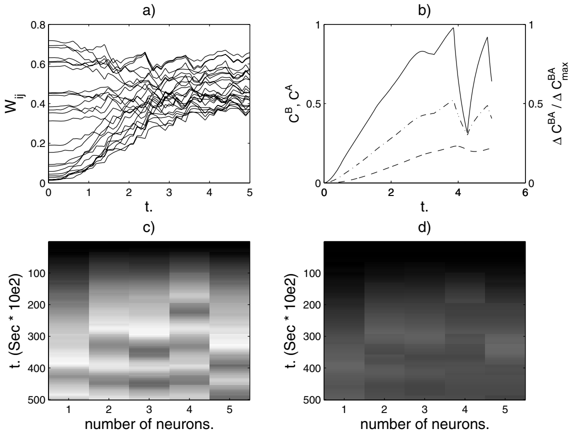


Fig. 2. Results of the Mode A. a) dynamics of weights. b) $\Delta C^{BA}/\Delta C^{BA}_{max}$ in the neural micro-surround (1-1) (solid line), and profiles of NO in the post-synaptic (dash-dot line) and pre-synaptic (dashed line) compartment of the neural micro-surround (1-1). c) NO concentration in post-synaptic neural ensembles. d) NO concentration in pre-synaptic neural ensembles.

each one and with a constraint, only the post-synaptic neural ensemble consists of NO neurons, that is to say, neurons which can synthesize NO. In our developments we will use the compartmental model of NO diffusion [10], where a theoretical abstraction denominated *compartment* is the principal element for the diffusion process and it has its biological equivalent in a neuron. Furthermore, we can define a framework for studying the evolution of hebbian weights, namely neural micro-surround *i-j*, Fig. 1b, where *i* belongs to the post-synaptic area and *j* to the pre-synaptic area. The neural micro-surround is the minimal set of compartments containing all neurons and synapses implicated in a diffusion process. The concentration of NO and the different modes for NO spreading are associated to each compartment and neural micro-surround, respectively.

We will consider three different modes for spreading the NO from post-synaptic to pre-synaptic neural ensembles in the NS: *Mode A, B and C.* The information environment of our experiments consist of ten different input data, which are randomly presented to the NS, five times during five seconds. A biological constraint in the NO neurons is the existence of a refractory time for the synthesis processing. In our study all experiments have been performed considering NO saturation in time.

Mode A.- The NS is considered as an isotropic environment for NO diffusion, where the DNB and *Diffusive Hybrid Neuromodulation* (DHN) [6], effects are present. In this mode the NO concentration reached by the pre-synaptic neurons is the same. The NO diffusion is represented by the Eq. (6).

$$\begin{gathered}\dot{C}_i^B = D_B\{C_{i+1}^B + C_{i-1}^B - 2C_i^B\} - D_{BA}\{C_i^B - C_j^A\} - \lambda_{C^B} C_i^B + F_i \\ \dot{C}_j^A = D_{BA}\{C_i^B - C_j^A\} - \lambda_{C^A} C_j^A\end{gathered} \tag{6}$$

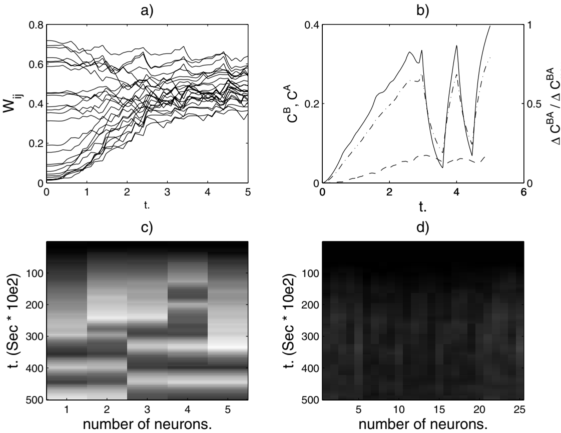


Fig. 3. Results of the Mode B. a) dynamics of weights. b) $\Delta C^{BA}/\Delta C^{BA}_{max}$ in neural micro-surround (1-1) (solid line), and profiles of NO in the post-synaptic (dash-dot line) and pre-synaptic (dashed line) compartment of the neural micro-surround (1-1). c) NO concentration in post-synaptic neural ensembles. d) NO concentration in pre-synaptic neural ensembles.

Where the C are the NO concentrations in the compartments, D^B and D^{BA} are the diffusion constants of the NS, the terms with λ_{C^B} and λ_{C^A} represent the self-regulation of NO production and F_i is the NO synthesis function which depends on activation state of the NO neuron with a maximun value of one.

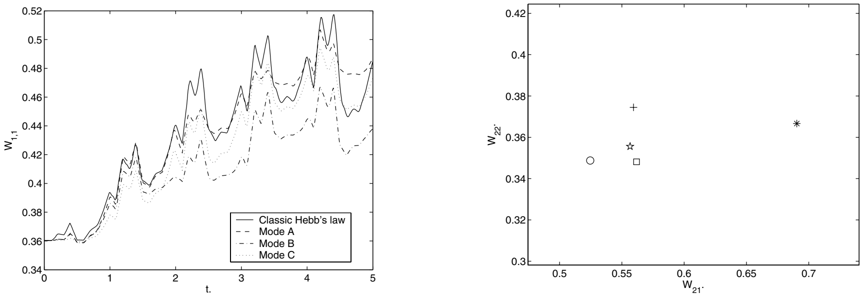
The dynamics for this mode A are given in the Fig. 2. We can observe the information transported by the NO from post-synaptic to pre-synaptic side providing a slighter evolution of weight than the classical Hebb's law. When the NO saturation in time is considered, complex structures emerge.

Mode B.- The NS is considered as an anisotropic environment for NO diffusion. This anisotropy is present only in the pre-and post-synaptic gap of the neural micro-surrounds and it will be represented by the diffusion constant which is in terms of the correlation of neural activities. The DNB and the DHN effects are present and the NO diffusion is represented by the Eq. (7).

$$\begin{gathered}\dot{C}_i^B = D_B\{C_{i+1}^B + C_{i-1}^B - 2C_i^B\} - N_i \sum_j g[S_i^B]\phi[S_j^A]\{C_i^B - C_j^A\} - \lambda_{C^B} C_i^B + F_i \\ \dot{C}_j^A = N_i g[S_i^B]\phi[S_j^A]\{C_i^B - C_j^A\} - \lambda_{C^A} C_j^A\end{gathered} \tag{7}$$

The dynamics for this mode B are given in the Fig. 3. We can observe the emergence of sharper complex structures than in mode A and the information transported by NO is less homogeneous than in that mode. The origin is the anisotropy of the NS.

Mode C.- The NS is considered as an anisotropic environment for NO diffusion in the same structural level than the mode B. This anisotropy will be in terms of a constant value ($F/3$). The only effect present in this mode is the produced by DHN. Here the NO diffusion is represented by the Eq. (8).



(a) (b)

Fig. 4. a) Comparative study of weight's dynamic. Classical Hebb's law (—), mode A (- - -), mode B (-.-), mode C (...). b) Comparative study of weight convergence in relation to the average value of the inputs. Average value of the inputs (*), classical Hebb (o), mode A (+), mode B (square), mode C (five-pointed star).

$$\begin{aligned} \dot{C}_i^B &= D_B\{C_{i+1}^B + C_{i-1}^B - 2C_i^B\} - \{(F_i/3) - C_j^A\} - \lambda_{C^B} C_i^B + F_i \\ \dot{C}_j^A &= \{(F_i/3) - C_j^A\} - \lambda_{C^A} C_j^A \end{aligned} \tag{8}$$

The dynamics for this mode C is more similar to the mode A dynamics and the weight dynamics is affected because the DNB effects are not considered. All comparative studies between classical Hebb's law and new Hebb's law supported by retrograde neuromessenger NO effect, can be observed in the Fig. 4. We can conclude that the diffusive Hebb's law presents a best learning capacity in whatever analised mode, Fig. 4b.

3 Conclusions

In this paper we have reached important conclusions on the new kind of neural communication, VT, by the NO and its effect on Hebbian learning, in both biological and artificial neural networks. We have confirmed our preliminary studies on the establishment of a new and more general framework of neural learning, which is based on synaptic and diffusive terms [6].

We have showed that the NO diffusion effect in a Hebbian neural system can be an underlying mechanisms for supporting the Hebb's learning law, showing the capacity of NO diffusion for the transmission of neural information. We have proposed a new mathematical expression for the Hebb's learning law, the diffusive Hebb's law, with points of stable convergence in the space of weights. This new expression embodiments the post-synaptic effects as a function of the gradient of NO concentrations between post- and pre-synaptic side. From the comparative study between the classical Hebb's law formulation and the diffusive Hebb's law we can deduce that our proposal is biologically more plausible presenting best learning capacity. We confirm the non-local and the non-synapse-specific character of the Hebbian learning. We have also study the effect of NO saturation in time in the learning processing and in the emergence of complex neural structures. We have confirmed the important effect of the DHN in the neural architecture and learning processing. We have demonstrated the goodness of our compartmental model of the NO diffusion and the theoretical abstraction of the compartment, for explaining the diffusion process of NO in relation to learning.

References

1. Huizhong W. Tao and Mu-ming Poo. Retrograde signaling at central synapses. (2001).PNAS. 98, N20, 11009-11015.
2. Brown, T.H., Kairiss, E.W., Keenan, C.L. Hebbian Synapses: Biophysical Mechanisms and Algorithms. (1990). Annu. Rev. Neurosci., 13, 475-511
3. Tanzi, E. I fatti e la induzioni nell 'odierna istologia del sistema nervioso. (1893). Riv. Sper. Freniatr. Med. Leg. Alienazioni Ment. Soc.Ital. Psichiatria 19:419-472.
4. Gluck, M.A. and Granger, R. Annu. Rev. Neurosci. (1993). 16, 667-706.
5. Hebb D.O. The organization of behaviour. (1949). New York, Wiley and Sons.
6. Suárez Araujo, C.P. Study and Reflections on the Functional and Organisational Role of Neuromessenger Nitric Oxide in Learning: An Artificial and Biological Approach. (2000). Computer Anticipatory Systems, AIP 517, 296-307.
7. Fernández López P., Suárez Araujo C.P., García Báez P., Sánchez Martín G. Diffusion Associative Network: Diffusive Hybrid Neuromodulation and Volume Learning. (2003). Lecture Notes in Computer Sciences, Vol. 2686, pp. 54-61.
8. Sergei A. Korneev, Volko Straub, Ildik Kemenes, Elena I. Korneeva, Swidbert R. Ott, Paul R. Benjamin, Michael OShea. Timed and Targeted Differential Regulation of Nitric Oxide Synthase (NOS) and Anti-NOS Genes by Reward Conditioning Leading to Long-Term Memory Formatio. (2005) J. Neurosci. 25(5):1188-92
9. Suárez Araujo C.P., Fernández López P., García Báez P. Towards a Model of Volume Transmission in Biological and Artificial Neural Networks: A CAST Approach. (2001). Lecture Notes in Computer Sciences, Vol. 2178, pp 328-342,.
10. Fernández López, P., Suárez Araujo, C.P., García Báez, P., Regidor García, J. A Model of Nitric Oxide Diffusion based in Compartmental Systems. In Press.

Deterministic Modelling of Randomness with Recurrent Artificial Neural Networks

Norman U. Baier and Oscar De Feo

Ecole Polytechnique Fédérale de Lausanne (EPFL),
IC ISC LANOS, Station 14, CH-1015 Lausanne, Switzerland
oscar.defeo@epfl.ch

Abstract. It is shown that deterministic (chaotic) systems can be used to implicitly model the randomness of stochastic data, a question arising when addressing information processing in the brain according to the paradigm proposed by the EC APEREST[1] project. More precisely, for a particular class of recurrent artificial neural networks, the identification procedure of stochastic signals leads to deterministic (chaotic) models which mimic the statistical/spectral properties of the original data.

1 Introduction

Recently a method for the *"chaos-based modelling of diversity and synchronisation-based categorisation of approximately periodic signals"* has been proposed [1]; this method combines theoretical results about generalised synchronisation of chaotic systems [2, 3] with nonlinear dynamical systems identification [4] and, by means of the implicit diversity and self-similarity of chaotic trajectories, exploits chaos to represent the uncertainty of signals and, afterwards, chaos synchronisation for categorising them into distinct classes. Clearly, key point of this method is the identification of a chaotic system given example data: the identified model must implicitly account for data diversity via the self-emerging chaos diversity [4].

Founding on these results, the European project APEREST[1], out of which we are presenting some results here, investigates whether or not neuronal erratic (chaotic) behaviour is similarly involved in representing diversity. Under such hypothesis, neuronal chaos is required to model knowledge and stimuli diversity also when this diversity is non deterministic.

Here, also in view of building new bio-inspired information systems, we address this point by resorting to artificial modelling; hence, the above key requirement translates to the need for chaos to emerge in artificial neural networks also when identifying random processes. In particular, we investigated this point considering a recently proposed recurrent artificial neural network (RANN) paradigm [5, 6], called either "Liquid State Machine"(LSM) or "Echo State Networks" (ESN). Since this paradigm has been argued to be biologically plausible, it has been a good candidate for artificially investigating the emergence of chaos, when randomness is to be modelled, keeping a foundation of biological plausibility.

[1] APEREST: IST-2001-34893 and OFES-01.0456. http://aperest.epfl.ch

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 255–260, 2005.

2 The "Echo State Networks" (ESN)

The general topology of ESN [7] is as shown in Fig. 1 and has state equations

$$\mathbf{x}_{k+1} = \alpha \mathbf{x}_k + \beta \tanh(W_{\text{int}} \mathbf{x}_k + W_{\text{back}} y_k + B) \tag{1}$$

$$y_k = \tanh(W_{\text{out}} \mathbf{x}_k) \ ; \tag{2}$$

where the value at the nodes at time k is given by the state vector

$$\mathbf{x}_k = \left[x_{1,k}, x_{2,k}, \ldots, x_{n,k}, \ldots, x_{N,k}\right]^T \ ; \tag{3}$$

the $\tanh(\cdot)$ is meant elementwise, *i.e.* the activation function is a sigmoid, scaled by β and its arguments are the contributions of the other nodes and a bias B; and, finally, the memory of the nodes is controlled by the parameter α.

ESN are recurrent artificial neural network (RANN) consisting of a layer of internal nodes and an output node, in turn fed back into the internal nodes. The internal nodes are interconnected with weights held in the matrix W_{int}, while the feedback from the output is weighted by W_{back}. In the ESN paradigm these weights are determined once in the beginning, independently of the application, and remain fixed; which is the novelty of ESN compared to traditional RANN [5, 7]. During the identification, only the output weights (W_{out}) are trained.

To identify the model of a given time series the output connections are broken and, instead of the signal at the output node, the training sequence y_{train} is fed into the network generating a state sequence $\mathbf{X}_{\text{train}}$. The identification is performed solving the output equation (2) for W_{out} [7]. If the activation function is invertible, this is performed by means of a least square regression, *i.e.*

$$W_{\text{out}} = \text{atanh}(y_{\text{train}}) \mathbf{X}_{\text{train}}^T \left(\mathbf{X}_{\text{train}} \mathbf{X}_{\text{train}}^T\right)^{-1} \ . \tag{4}$$

The connection weights, other than the output connections assigned during the identification, are usually assigned randomly out of few possible values [7]. Since the paradigm assume sparse connectivity (few parameters), we have performed an almost exhaustive search in the parameter space to determine optimal

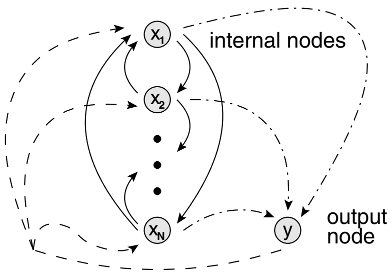


Fig. 1. General topology of Echo State Networks. Arrows indicate internal (solid, W_{int}), feedback (dashed, W_{back}), and output (dash-dotted, W_{out}) connections, respectively.

parameter settings [8]. As result, a network dimension of 60 was chosen; the bias vector B has been set with 25% of the values at 0.14, 25% at -0.14, and the remaining at 0; the final connection probability of the internal connections (W_{int}) was $1.33 \cdot 10^{-2}$ and the maximal eigenvalue of W_{int} did not exceed the value 1; finally, the scaling variables were $\alpha = 0.604$, $\beta = 0.44$, as proposed by Jaeger [7].

3 Deterministic and Stochastic Diversity

In the Introduction we have presented the aim of this work, *i.e.* implicitly modelling the stochastic diversity of approximately periodic time series with deterministic chaos. Here it will be explained what is meant by *diversity.* We start by considering the deterministic chaotic time series shown in Fig. 2(a,b): (a) shows few pseudo-periods of a time series generated by the Colpitts oscillator [9]; in (b) the same time series is represented in a so called stroboscopic plot, *i.e.* the pseudo-periods (grey lines) are plotted on top of each-other. From this latter the approximate periodicity of the signal can clearly be identified; however, it is also clear that the pseudo-periods are *diverse* around the black line, which represents the mean stereotype, *i.e.* the temporal mean value of the grey lines. Chaotic time series represent an extreme case of perfectly structured diversity, where the value of the time series at a given time is completely deterministic and uniquely determined by its previous values.

On the other extreme there are purely stochastic signals, *e.g.* a signal produced by a stochastic process followed by a filter, as that shown in Fig. 2(c,d). In this case the diversity is completely unstructured. Although there is a certain similarity between the structured and the unstructured cases, in the latter all correlation within the time series has been introduced by the filter and the time series is non-deterministic, while it is in the former.

Other signals, falling between these two extreme cases, can easily be conceived. Close to the completely structured case, time series surrogates of deterministic signals can be imagined [10]: constructed by taking a deterministic time series and adding phase noise to it. They destroy every determinism while preserving the spectral properties, *i.e.* the amplitude spectrum of the surrogate signal is the same as that of the original signal, but the diversity is now unstruc-

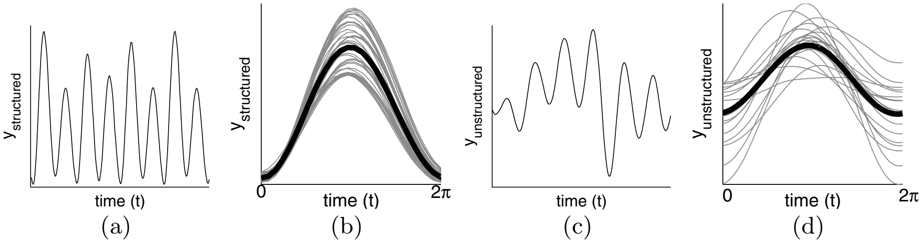


Fig. 2. Structured (a,b) and unstructured (c,d) diversity view in the time domain (a,c) and in the stroboscopic plots (b,d)

tured. Oppositely, close to the purely stochastic case, periodic signals corrupted by noise with assigned spectral properties can be conceived, *e.g.* corrupted by red-like noise, where the power scales in frequency as $\frac{1}{f^\gamma}$ and γ is referred as noise colour.

4 Self-emergence of Chaos in RANN Identification

We have investigated whether it is possible that deterministic systems (RANN) can identify stochastic data producing a deterministic (chaotic) model, whose structured diversity mimics the statistical (spectral) properties of the unstructured diversity of the original data. Here are reported our results on this question.

ESN have already been shown capable of identifying chaotic behaviour [7]. Hence, not surprisingly, they can also correctly identify time series from the Colpitts oscillator [8]. Though, it has never been shown whether the identification works because the time series is deterministic and the diversity structured, or whether the algorithm described in Section 2 could also identify time series with similar spectral properties but stochastic in nature. This point has been addressed identifying surrogates of chaotic signals, constructed from the Colpitts oscillator time series as described in the previous section. The results are summarised in Fig. 3. Figure 3(b) shows that the output of identified system has its own diversity; since the ESN used is completely deterministic, this diversity can only be generated by the nonlinear dynamical complexity of the system, i.e. complex tori or chaos. Unfortunately, given the high dimensionality of the system, a numerically computed Lyapunov spectra would not allow to distinguish the two cases. Though, the unimodal and fractal nature of the peak-to-peak plot and analysis (*cf.* Fig. 3(c)) corroborate the chaotic nature of the identified system [11]. Furthermore, visual inspection of Figs. 3(a), (b) and (c), and a deeper analysis [8], shows that the identified system mimics the spectral/stochastic properties of the original signal, whose diversity is, however, not structured, but stochastic.

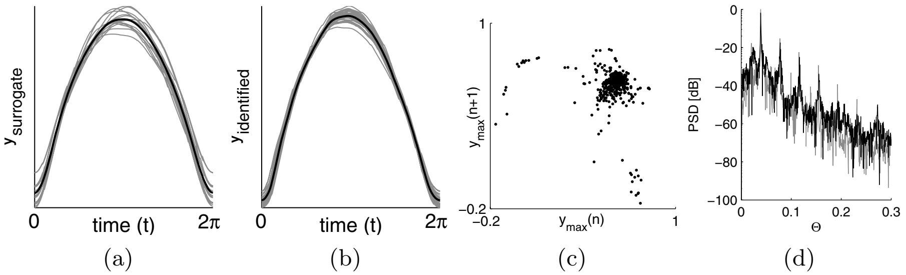


Fig. 3. Identification of surrogates of chaotic signals. Stroboscopic plot of: (a) – identifying surrogate signal; (b) – signal from the identified system; (c) – peak-to-peak map of the identified system; (d) – comparison of the spectral properties of identifying (grey) and identified (black) signals.

To qualify the extent to which a chaotic (deterministic) system can mimic a stochastic behaviour, periodic time series corrupted by noise with assigned power spectrum (red-like noise) have been considered. The spectra of the identifying and identified signals, with respect to the *colour* and intensity of the corrupting noise, are reported in Fig. 4. From a visual inspection emerges that for ambiguous noise (shaded) the identified system does not mimic the spectral properties of the identifying signals. However, for intense highly coloured noise (in white) the identification provided a deterministic system whose spectral properties mimic those of the original data, independently from their stochastic nature; not shown here for lack of space, a peak-to-peak analysis of the corresponding identified systems corroborate the chaotic origin of emerging diversity, whilst a stroboscopic plots analysis confirms the similarity between identifying and identified diversity. Finally, note that the spectra at low frequencies are not always well mimicked, this is due to the spectral properties of the emerging invertible chaos, which do not admit sudden jumps.

Concluding, chaotic deterministic RANN can mimic signals whose spectra are not necessarily physically plausible, as it is the case of surrogate chaotic time series, and indeed time series with some arbitrary spectra can be mimicked.

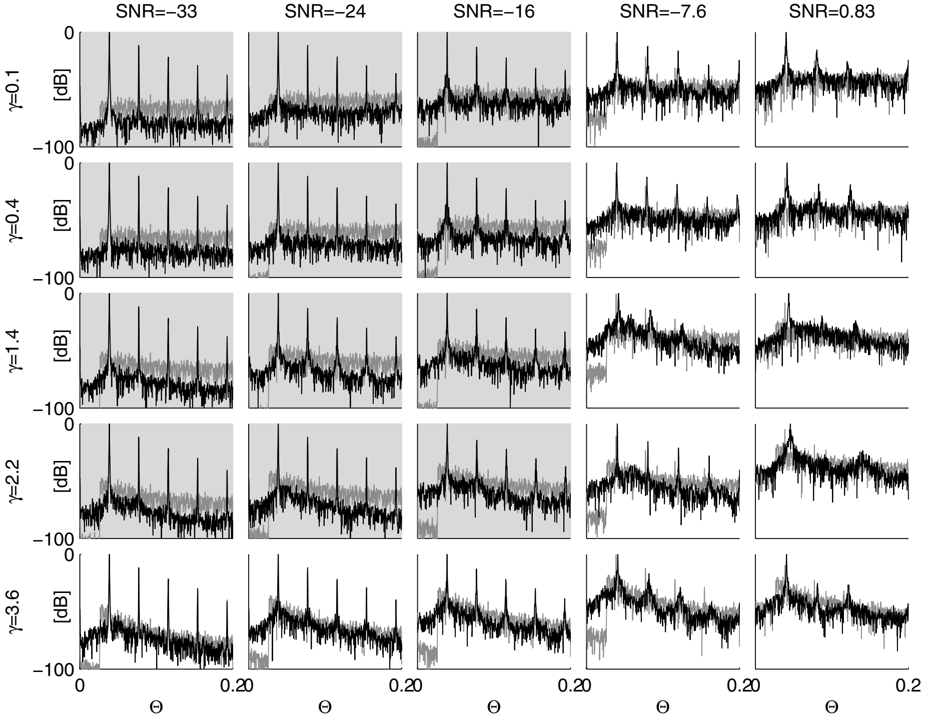


Fig. 4. Identification of stochastic signals. Spectra of the identifying (grey) and identified (black) signals with respect to *colour* and intensity of the corrupting noise.

Though, the diversity of the considered signals has to be *persuasive*; on the contrary, when the diversity is too noise-like, *i.e.* too weak or too white, the identified system is likely to not mimic it correctly.

5 Conclusions

It has been shown that chaos can indeed be used to model randomness, a question arising when addressing information processing in the brain according to the paradigm proposed by the APEREST project.

This result completes a recently introduced technique of classifying approximately periodic time series [4, 2, 3]. Indeed, in these previous works chaos has been observed emerging when identifying irregular behaviour; though, it has not been shown whether this happens also when identifying purely stochastic signals.

As mentioned in the Introduction, the identification technique discussed here can be complemented with chaos synchronisation [2, 3] to perform classification. This is out of the scope of this communication and will be presented elsewhere; however, it can be anticipated that the achieved results are indeed good also when compared to other techniques [8].

References

[1] De Feo, O., Hasler, M.: Modelling diversity by chaos and classification by synchronization. In Pikovsky, A., Maistrenko, Y., eds.: Synchronization: Theory and Application. Kluwer Academic Publishers, Dordrecht, Germany (2003) 25–40

[2] De Feo, O.: Qualitative resonance of Shil'nikov-like strange attractors, part I: Experimental evidence. Int. J. Bifurcat. & Chaos **14** (2004) 873–891

[3] De Feo, O.: Qualitative resonance of Shil'nikov-like strange attractors, part II: Mathematical analysis. Int. J. Bifurcat. & Chaos **14** (2004) 893–912

[4] De Feo, O.: Self-emergence of chaos in the identification of irregular periodic behaviour. Chaos **13** (2003) 1205 – 1215

[5] Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: A new framework for neural computation based on perturbations. Neural Comput. **14** (2002) 2351–2560

[6] Jaeger, H., Haas, H.: Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. Science **304** (2004) 78–80

[7] Jaeger, H.: The echo state approach to analysing and training recurrent neural networks. Technical Report 148, Fraunhofer Institute for Autonomous Intelligent Systems (2001)

[8] Baier, N.: Approximately Periodic Time Series and Nonlinear Structures. PhD thesis, Ecole Polytechnique Fédérale de Lausanne (2005)

[9] Maggio, G.M., De Feo, O., Kennedy, M.P.: Nonlinear analysis of the Colpitts oscillator and applications to design. IEEE T. Circuits-I **46** (1999) 1118 – 1130

[10] Schreiber, T., Schmitz, A.: Surrogate time series. Physica D **142** (2000) 197 – 385

[11] Candaten, M., Rinaldi, S.: Peak-to-Peak dynamics: A critical survey. International Journal of Bifurcation and Chaos **10** (2000) 1805–1819

Action Understanding and Imitation Learning in a Robot-Human Task*

Wolfram Erlhagen[1], Albert Mukovskiy[1], Estela Bicho[2], Giorgio Panin[3], Csaba Kiss[3], Alois Knoll[3], Hein van Schie[4], and Harold Bekkering[4]

[1] Dept. Mathematics for Science and Technology, University of Minho, 4800-058 Guimaraes, Portugal
{wolfram.erlhagen, am}@mct.uminho.pt

[2] Dept. Industrial Electronics, University of Minho, 4800-058 Guimaraes, Portugal
estela.bicho@dei.uminho.pt

[3] Informatics, Chair for Robotics and Embedded Systems,
Technical University Munich, 85748 Garching, Germany
{panin, knoll, kiss}@in.tum.de

[4] Nijmegen Institute for Cognition and Information, Radboud University Nijmegen, 6500 HE Nijmegen, The Netherlands
{H.vanschie, H.bekkering}@nici.ru.nl

Abstract. We report results of an interdisciplinary project which aims at endowing a real robot system with the capacity for learning by goal-directed imitation. The control architecture is biologically inspired as it reflects recent experimental findings in action observation/execution studies. We test its functionality in variations of an imitation paradigm in which the artefact has to reproduce the observed or inferred end state of a grasping-placing sequence displayed by a human model.

1 Introduction

In robotics research imitation has attracted a lot of attention in recent years since it is considered a promising learning mechanism to transfer knowledge from an experienced teacher (e.g. a human) to an artificial agent. Most work has been focused on motor learning paradigms in which the imitating robot has to match as close as possible the kinematics and dynamics of an observed movement (for review see [1]). However, a growing body of experimental findings in imitation studies with humans indicate that the imitator most likely does not encode the full detail of the observed motions but the interpretation of those motions in terms of the demonstrator's goal. Very often differences in embodiment (e.g., child-adult) and/or task constraints (e.g., obstacles) simply do not allow for a matching on the level of the movement trajectory. In the goal-directed theory of imitation proposed by Bekkering and colleagues [2,3],

* The work was supported by the European grant*ArteSImit* (IST-2000-29686). We would like to thank Drs Leonardo Fogassi and Giacomo Rizzolatti for numerous discussions about this work.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 261–268, 2005.

imitative behavior can be considered successful whenever the end state of the witnessed action is reproduced. The action means, on the other hand, may or not coincide with the observed ones.

In the work reported here: (a) we present a robot control architecture for goal-directed imitation which reflects processing principles discovered in recent experimental findings of action observation/execution studies, (b) we propose a biologically plausible learning scheme for establishing the links between means and goals during development and practice, (c) we test the complete control architecture in variations of an imitation paradigm in which a robot tries to reproduce the observed or inferred outcome of a grasping and placing sequence displayed by a human, (d) we show that knowledge about the meaning of an object may be transferred to the robot by imitation.

Goal-directed imitation requires of course that the imitator understands the action of the model. The neuro-cognitive mechanisms underlying action understanding are currently topic of an intense debate (e.g., [4,5]). A growing body of empirical evidence supports the notion that the production and perception as well as the interpretation of others' actions rely on a common distributed neural system. The 'direct matching hypothesis' proposed by Rizzolatti and colleagues [5] based on their discovery of the mirror system states that an action is understood when its observation activates the motor representations controlling the execution of the same action. However, the correspondence problem between dissimilar embodiments challenges an explanation purely based on a simple and direct resonance phenomenon of the motor system. Moreover, humans and monkeys are able to infer goals without a full vision of the action by integrating additional contextual cues.

The proposed model architecture for action understanding and goal-directed imitation in artefacts is based on the theoretical framework of dynamic fields [6,7]. It aims at implementing the idea that inferring motor intention is a continuous process which combines sensory evidence, prior task knowledge and a goal-directed matching of action observation and action execution.

2 Experimental Setup

For the robotics work we adopt a paradigm which has been developed to experimentally investigate in humans the idea that actions are organized in a goal-directed manner (van Schie and Bekkering, in preparation). The paradigm contains two objects of different color that must be grasped and then placed at one of two laterally presented targets that differ in height. The possible hand trajectories are constrained by an obstacle in form of a bridge (see Panel A in Fig. 1). Depending on the height of the bridge, the lower target may only be reached by grasping the object with a full grip and transporting it below the bridge. Placing the object at the higher target, on the other hand, may require combining a precision grip and a hand trajectory above the bridge.

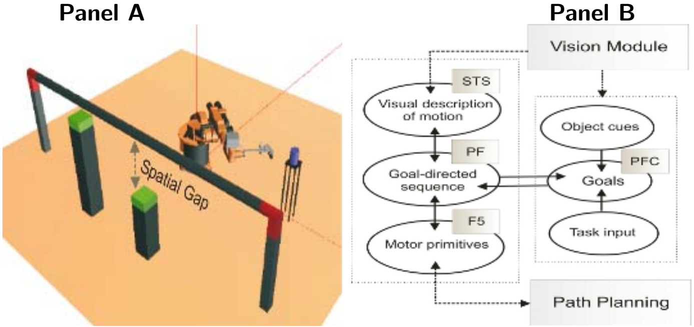


Fig. 1. **Panel A:** Bridge Paradigm. **Panel B:** Robot control architecture.

3 The Control Architecture

The artefact used in the imitation study consists of an industrial robot arm (Kuka with 6 DOFs) with a four-fingered anthropomorphic robot hand (Graal-Tech, University of Genova) and a real-time vision system. Three interconnected modules define the robot control architecture (Panel B in Fig. 1).

The *vision module* provides the environmental variables of the task setting (shape and position of bridge, position of object and targets etc.) by means of a semi-automatic calibrated stereo camera system. In addition, it tracks the detected object(s) and the hand of the instructor, classifies the demonstrated action in terms of grip and trajectory type, and identifies the placing target. All outputs are globally available for the other modules.

In the *cognitive module*, decisions about the action goal and the means to achieve that goal are made. Its layered architecture is biologically inspired, as it represents the basic functionality of neuronal populations in interconnected brain areas known to be involved in action observation/execution tasks (for details see [8]). The core part consists of three reciprocally connected layers, STS, PF and F5, representing the mirror circuit. The fundamental idea is that within this circuit the matching of action observation and action execution takes place on the level of motor primitives which abstract from the fine details of the movements [5]. Concretely for the bridge paradigm, we distinguish two types of grasping primitives (precision (PG) and full (FG) grip) and two types of transporting primitives for avoiding the obstacle (below (BT) or above (AT) the bridge). The visual description of the observed action is stored in STS. In the motor layer F5 the representations of the respective primitives become active both during action observation and action execution, that is, we assume that those primitives already exist in the motor repertoire of the robot. The representations in the intermediate layer PF reflect recent neurophysiological findings in brain areas PF/PFG that suggest a goal-directed organization of action means. Using a sequence task, Fogassi and colleagues [9] described a population of grasping neurons which showed a selective response in dependence of the final goal of the action (eating or placing) to which the grasping act belongs. For the bridge paradigm, we abstract this finding by assuming representations of specific combinations of primitives (e.g., PG-AT) which allow achieving a specific goal. Layer PF is reciprocally connected with a prefrontal area (PFC) in which the goals parameterized by their height relative to the bridge are encoded. A goal representation may be triggered or influenced by visual input (placed object), through the links to PF, and/or learned associations to representations of object cues (e.g., color) and memorized task information (e.g., number and probability of goals).

In the *path planning module*, the abstract primitives of layer F5 are translated into a movement plan generating the right kinematics. We assume that the path planning takes place in posture space. This requires that a model of the inverse kinematics for the arm-hand system is known. For the planning we employ the framework of wave expansion networks [10] with nodes representing stored arrays of joint angles. The sequence of postures defining a collision-free path for the robot arm-hand is found by propagating an activity wavefront between nodes encoding the initial and the desired goal postures. Posture nodes which are impossible due to the obstacle are inhibited. They are found by explicitly testing for spatial overlap in Cartesian space between the to-be-assumed posture and the bridge (forward maps). Moreover, the ensemble of nodes which can become part of the wavefront is further constrained by the motor primitives in F5. For instance, we use again forward maps to check whether a particular node represents ‘all links in a high position’ as required by an AT-trajectory. This integration of prior information together with the inherent parallelism of the wavefront operations makes a real-time path planning for artefacts with higher degrees of freedom feasible.

4 Hebbian Learning of the Synaptic Links

Each layer of the cognitive module is formalized by a Dynamic Field [6,7] in which self-sustained patterns of excitation encode task specific information. The layer dynamics is governed by the following equation:

$$\tau\frac{\delta}{\delta t}u(x,t) = -u(x,t) + g(u(x,t)) \left[\int w(x-x')f(u(x',t))dx' - \right.$$
$$\left. -w_{inhib}\int f(u(x',t)dx'\right] + h + \sum_i S_i(x,t) \qquad (1)$$

where $\tau > 0$, $h < 0$ and $w_{inhib} > 0$ are constants. The non-linear functions $f(u)$ and $g(u)$ are of sigmoid shape, the excitatory connections, $w(x,x')$, are modelled as a Gaussian profile. The excitation patterns evolve under the influence of multiple information sources, $\sum_i S_i(x,t)$, representing input from the visual module and from excitation in connected layers. Recurrent inhibition in each layer creates a competition between response alternatives (e.g., type of grasping) and guarantees the stability of the decision process represented by the excitation patterns.

To develop the goal-directed organization within the distributed network model, the synaptic links between the various layers have to be established during practice. We apply a correlation based learning rule for the synaptic connections, $a(x, y)$, between any two neurons x and y belonging to two different layers (for a discussion in the context of action understanding see [4]):

$$\tau_s \frac{\delta}{\delta t} a(x, y, t) = -a(x, y, t) + \eta\, f(\bar{u}_1(x)) f(\bar{u}_2(y)) \tag{2}$$

where $\tau_s \gg \tau$, $\eta > 0$ and $\bar{u}_1$, $\bar{u}_2$ denote the equilibrium solutions of the relaxation phase in layer 1 and layer 2, respectively. Importantly, we assume that an internally generated reinforcement signal representing a successful path planning toward the desired goal posture defines the time window for the learning. As a result, the matching of action observation and action execution becomes goal-directed, since the metric for the learning is not defined by the similarity in the kinematics but the similarity in the end state [11].

5 Experimental Results

A set of imitation experiments within the Bridge paradigm has been performed which differ mainly in the amount and nature of the visual information available to the robot. In the first experiment, a complete visual description of the teacher's action in terms of the grasping and transporting behavior exists and

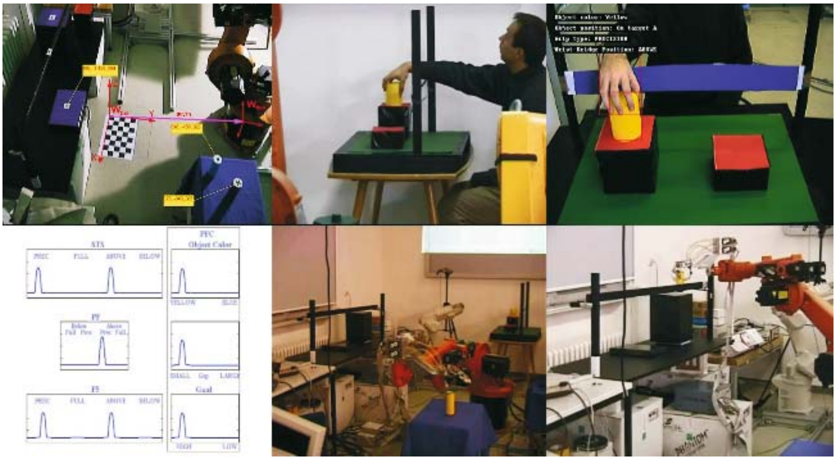


Fig. 2. Example of an imitation task in which the robot reproduces the placing on the higher goal using the means displayed by the human teacher (PG-grip, AT-trajectory). The upper row depicts the visual analysis of the world configuration (left) and the teacher's action (right). On bottom, decisions for goal and means represented in the dynamic field model (left) and two snapshots of the robot in action are shown.

the visual system identifies the goal. The visual description in STS resonates via the matching mechanism in the mirror circuit with the congruent motor primitives in PF and F5. If the covert path planning toward the desired goal-posture turns out to be successful, the observed response strategy can be associated with the goal for future use by the learning procedure described above. Figure 2 illustrates the result of this learning by imitation in an example in which the robot copies the demonstrated precision grip and the trajectory above the bridge to place the object at the higher goal. By using objects with different properties (e.g., color), the robot may acquire additional knowledge in repeated imitation trials by learning an association between object cues and the goal where to place a particular object ('object meaning'). For instance, yellow objects have to be placed at the higher and blue objects at the lower target.

The second experiment shows that the learned link from the mirror circuit to the goal representation is crucial. The bar of the bridge is removed for the human teacher but not for the robot (Panel A in Fig. 3). Because of this change in the environmental constraints, the teacher now uses a full grip for placing the yellow object at the higher target. For the robot, a direct matching on the level of motor primitives would result in a collision with the bridge. As shown in the snapshot of the stable state of the network dynamics in Panel A of Figure 3, the decisions in layer F5 represent the motor primitives PG (grip) and AT (trajectory) previously associated with the higher goal parameterized by the smaller

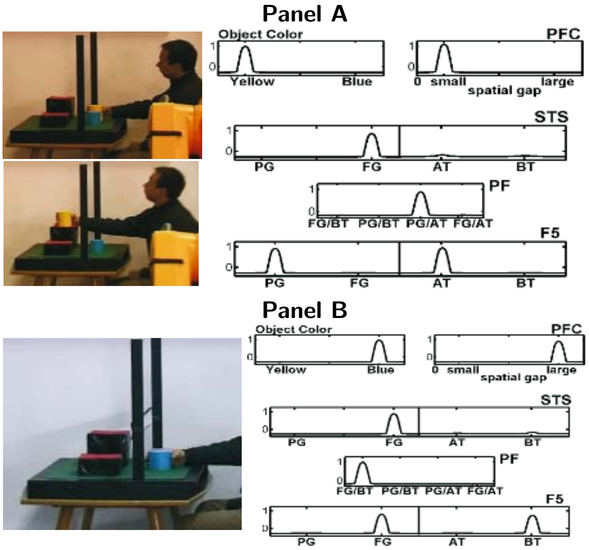


Fig. 3. **Panel A:** Conflict in the grasping behavior: the teacher uses a full grip for placing the object. As shown in layer F5 of the field model, the robot decides to use a precision grip to reproduce the end-state. **Panel B:** Inference task: only the grasping behavior is observable, the transpor ting and placing is hidden from view. The stable state in layer PFC of the field model represents the inferred goal.

spatial gap relative to the bridge. The observation of the placing triggers the respective goal representation which in turn biases the decisions in PF and F5 toward the PG-AT sequence. The input from PFC may override the direct mapping in the mirror circuit since in known task settings the goal representations in PFC evolve faster compared to the visual motion representations in STS (for details see [8]).

The third experiment reflects a situation which is very common for agents acting in cluttered and dynamic environments. Only partial visual information about the action displayed by another agent is available and the observer has to infer the action goal by integrating additional contextual cues and prior task information. Again we assume that the underlying mechanism for discerning motor intention is a goal-directed motor simulation. Consistently, it has been recently reported that grasping mirror neurons fire when the crucial final part of the demonstrated action is hidden but the monkey knew that there is a graspable object behind the occluding surface (for review [5]). In Panel B of Figure 3 we show a snapshot of the model dynamics in an inference task in which only the grasping with a full grip was observable and the rest of the action was hidden from view. Despite the missing visual information, the dynamics has relaxed in each model layer to a stable peak solution. The representation of the lower goal (parameterized by the larger spatial gap relative to the bridge) is triggered via the learned STS-PF-PFC pathway. Note that for the specific height of the bridge, the FG-BT sequence is the only sequence including the FG-grip which is associated with a particular goal. If the robot and the human model already share the knowledge about the meaning of the object, the color information serves as an additional, redundant input for the goal representation in PFC.

6 Discussion

The experiments with the real robot system illustrate some of the advantages of a goal-directed organization of imitative behavior compared to other current approaches which emphasize a matching on the trajectory or path level [1,11]. It allows coping with differences in embodiment and task constraints known as the correspondence problem [12] in robot imitation. Most importantly, it enables the robot to infer the purpose of the observed movement which is crucial for transferring specific knowledge from the model to the imitator. The idea that the movement production system is essentially involved in action understanding has been proposed in the context of robotics research several times in the past (for review [1]). For instance, Demiris and Hayes [13] used internal forward models known from motor control theory to predict the sensory consequences of observed actions in an imitation task. However, the use of forward models implicitly requires that imitator and demonstrator share similar bodies. Wolpert and colleagues [14] have recently proposed a computational scheme which also includes hierarchically higher levels of motor control in the simulation loop for action understanding. On this view, the goal-directed organization of action means in layer PF may be seen as part of an abstract forward model for interpreting an ongoing action. We have recently shown that the goal-directed control architecture may even allow acquiring the meaning of an observed motor act which is not strictly in the motor repertoire of the observer [8].

In our current work, we consider paradigms that involve a richer set of motor primitives and more complex goal-directed sequences.

References

1. Schaal, S., Ijspeert, A., and Billard, A.: Computational approaches to motor learning by imitation. Phil. Trans. R. Soc. Lond. B Vol. 358 (2003)537-547
2. Bekkering, H., Wohlschläger, A., and Gattis, M.: Imitation of gestures in children is goal-directed. The Quartely Journal of Experimental Psychology, Vol. 53A (2000) 153-164
3. Wohlschäger, A., Gattis, M., and Bekkering, H.: Action generation and action perception in imitation: An instantiation of the ideomotor principle. Phil. Trans. R. Soc. Lond. B. Vol. 358 (2003)501-515
4. Keysers, C., and Perrett, D. I.: Demystifying social cognition: a Hebbian perspective. Trends in Cognitive Science. Vol.8 (2004)501-507
5. Rizzolatti, G., Fogassi, L., and Gallese, V.: Neurophysiological mechanisms underlying the understanding and imitation of action. Nature Reviews. Vol.2 (2001)661-670
6. Amari, S.: Dynamics of pattern formation in lateral-inhibitory type neural fields. Biological Cybernetics. Vol.27 (1977)77-87
7. Erlhagen, W., and Schöner, G.: Dynamic field theory of movement preparation. Psychological Review. Vol.109 (2002)545-572
8. Erlhagen, W., Mukovskiy, A., and Bicho, E.: A dynamic model for action understanding and goal-directed imitation. Submitted
9. Fogassi, L., Ferrari, P. F., Gesierich, B., Rozzi, S., Chersi, F. and Rizzolatti, G.: Parietal lobe: from action organization to intention understanding. Science. Vol.308 (2005) 662-667
10. Kassim, A., and Kumar, B.V.K.V.:The wave expansion neural network. Neurocomputing. Vol.16 (1997)237-258
11. Billard, A., Epars, Y.,Calinon, S., Schaal, S. and Cheng, G.: Discovering optimal imitation strategies. Robotics and Autonomous Systems. Vol.47 (2004)69-77
12. Alissandrakis, A., Nehaniv, C.L., and Dautenhahn, K.: Imitation with ALICE: Learning to imitate corresponding actions across dissimiliar embodiments. IEEE Transactions on Systems, Man, and Cybernetics- Part A. Systems and Humans. Vol.32 (2002)482-496
13. Demiris, J., and Hayes, G.: Imitation as a dual-route process featuring predictive and learning components: A biologically plausible computational model. Imitation in Animals and Artifacts. Dautenhahn, K. and Nehaniv, C.L. MIT Press, Cambridge MA (2002)327-361
14. Wolpert, D.M., Doya, K. and Kawato, M.: A unifying computational framework for motor control and social interaction. Phil. Trans. R. Soc. Lond. B. Vol.358 (2003)593-602

Comparative Investigation into Classical and Spiking Neuron Implementations on FPGAs

Simon Johnston, Girijesh Prasad, Liam Maguire, and Martin McGinnity

Intelligent Systems Engineering Laboratory, Faculty of Engineering, Magee Campus,
University of Ulster, Northland Rd, Derry, N.Ireland, BT48 7JL
sjohnston@ulster.ac.uk

Abstract. The current growth of neuron technology is reflected by the increasing focus on this research area within the European research community. One topic is the implementation of neural networks (NNs) onto silicon. FPGAs provide an excellent platform for such implementations. The development of NNs has led to multiple abstractions for various generations. The different demands that each generation pose, present different design challenges. This has left ambiguous decisions for the neuroengineer into what model to implement. The authors have undertaken an investigation into four commonly selected neuron models, two classical models and two formal spike models. A software classification problem is combined with hardware resource requirements for FPGAs, implemented utilising a novel design flow. This provides an overall comparative analysis to be made and identification of the most suitable model to implement on an FPGA.

1 Introduction

Neural networks have evolved vastly from the first generation McCulloch and Pitts (1943) neuron model [1], to the more biologically realistic spike models [2]. A neuroengineer is now confronted with a much wider choice and ambiguity in his/her decision into which model to employ. Classical non-spiking neuron models still remain the most popular choice. Spiking neurons (SNs) are more biologically plausible neuron models that offer new information processing paradigms for neuroengineers. Voltage spikes are used to temporally and spatially encode information. In this paper the application of two classical models, the multilayer perceptron (MLP) and radial basis function (RBF) along with two formal spike models, namely the leaky integrate and fire (LIF) and spike response model (SRM) are analysed. Section 2 contains an overview of the chosen neuron models. Section 3 presents the results for the first part of the investigation, the models development in software to solve a non-linearly separable classification problem. Section 4 contains the second part of the investigation, the hardware implementations of the various models onto an FPGA. Individual models and their equivalent networks are examined. Section 5 concludes with an overall comparative analysis and selection of most proficient model.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 269 – 274, 2005.

2 Models

Spike models introduce a new level of biological sophistication absent from classical models. Different levels of abstractions for these models exist. The complex biophysical neuron abstractions, such as compartmental and the Hodgkin and Huxley models [2] render them impractical for large scale simulations and implementations, being computationally too demanding. This has promoted the development of simpler formal spike models. Two such models are the (SRM) and (LIF). These models consist of two important features: a) stimuli are integrated over time and b) if a threshold is surpassed, a voltage spike is generated. Although classical models lack this biological refinement they are still the most popular neuron choice and have been implemented successfully in a greater number of applications in comparison to spike models. For these reasons two of the most popular classical neuron models have been selected to be compared with two commonly selected spike models.

2.1 Classical Models

The first classical model chosen was the feedforward MLP trained using back propagation (BP), in a supervised manner. Mathematically it can be described by (1).

$$y_k = f(\sum_{j=1}^{m} w_{kj} x_j + b_k) \,. \tag{1}$$

where *x* represent the input features : *w* are the synaptic weights of the neuron *k* : b_k is the bias : *f(.)* is the activation function : y_k is the neuron output.

A sigmoid activation function was employed for the MLP networks, Equation (2).

$$f(x) = \frac{1}{1+e^{-ax}} \,. \tag{2}$$

where a is the slope parameter of the sigmoid function.

RBF neurons are also arranged as multilayer feedforward networks but use a different Gaussian type activation function, expressed by (3).

$$\Phi(r) = \exp(\frac{\| x-\mu \|^2}{\sigma^2}) \tag{3}$$

where σ is the spread : μ is the centre : ||.|| represents Euclidean distance.

2.2 Spike Models

For full mathematical notations and descriptions of the LIF and SRM model equations (c.f. [2]). The internal dynamics of the LIF neuron is modelled by the differential equation (4).

$$c \frac{du(t)}{dt} + \frac{u(t)}{R} = I(t) \,. \tag{4}$$

where C represents the capacitance : R the resistance : u the potential : I(t) the current.

The SRM is a generalisation of the LIF model, where the parameters are made time dependent instead of voltage dependent. SRM is not defined using differential equations instead the membrane potential is considered as the integration of input spikes. Mathematically it is defined as (5).

$$u_i(t) = \eta \ (t - t_i^{(f)}) + \sum_j w_{ij} \sum_f \varepsilon_{ij} (t - t_j^{(f)}) \ . \tag{5}$$

where μ_i(t) is the membrane potential of neuron *i* : *t,* time since last spike output *:* $t_j^{(f)}$, spike time of presynaptic neurons j : η *(.)* the kernel to model refractory period : ε_{ij} *(.)* kernel to model post synaptic potential of neuron *i* induced by a spike from neuron *.j*: w_{ij} the synaptic strength.

3 Comparative Evaluation of Models in Matlab

The first stage of this investigation focuses on determining the ability of the selected models to successfully classify the Iris dataset benchmark problem. The models were arranged into specific network topologies to suit the problem and a supervised learning algorithm was employed. All software models were developed and implemented in Matlab v7.0. For the MLP networks, the default training method basic gradient descent (BP) was used. For the RBF networks the default 'newrb' was applied which adds neurons to the hidden layer until the desired MSE is reached. With respect to the spike models, supervised training was achieved by employing an evolutionary strategy (ES) [3].

3.1 Iris Dataset

The Iris dataset contains 3 classes, which two are not linearly separable. The dataset contains 150 samples of 4 input variables. The 150 samples were divided into an equal training and test set of 75 samples each. A 1-D encoding scheme employed by Belatreche [3] was utilised, this pre processing scheme allowed 4 inputs to be temporally encoded using 4 receptive fields. This same pre processing was used for all the networks.

A network of 4x4x1 was sufficient for the MLP and the spike models to solve the Iris dataset. The RBF required a larger network size of 4x8x1. Both classical models used a MSE 0.01 as a stopping criterion. With concern to the spike models, target firing times of 6, 10 and 14 ms were chosen to classify the three respective species of flowers. A weight range of [-10, 10] and a delay range of [0, 9] was used for the spike models. Table 1 displays the Iris dataset results.

These results suggest that for such static a classification problem the choice of model is not well defined, as they all possess high classification accuracy. Therefore

Table 1. Iris classification accuracy

			Accuracy %	
Model	Algorithm	Hidden	Training set	Test set
LIF	ES-Approach	4	98.67	96
SRM	ES-Approach	4	98.67	96
MLP	Matlab BP	4	100	94.67
RBF	Matlab newrb	8	100	96

the choice in selection would stem greater on the models efficiency with respect to hardware implementation evaluations, for such problems.

4 Hardware Investigation

The second stage of the investigation is the hardware design, simulation and implementation of the selected models onto an FPGA. The approach uses Xilinx System Generator (XSG) for the simulation and implementation of the neuron models and their networks, as previously investigated in [4]. This flow was employed to facilitate the development of novel hardware equivalent circuits of the selected models. 16 bit representation was used for all models.

4.1 MLP Implementation

The MLP neuron employs a sigmoid activation function as described in 2.1. A second order non-linear function exists which can be used as a satisfactory approximation for the sigmoid function [5], detailed and implemented previously by Blake [6].This same expression was implemented in XSG, the sigmoid function circuit was then combined with the necessary block to develop the MLP neuron model shown below.

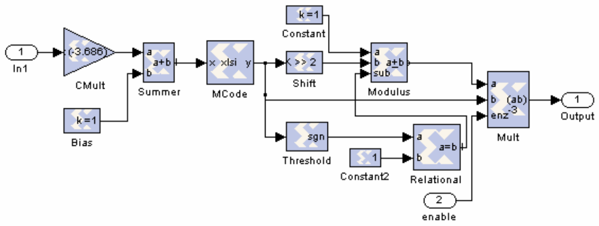


Fig. 1. MLP nex on circuit

Regarding RBF networks, it is acknowledged that these particular networks may require a greater number of neurons to accomplish the same classification as standard MLP networks using BP, although they train much faster. This fact was encountered in the software investigation. Therefore as time is not a criterion it would serve no purpose to develop their hardware equivalent circuits for this investigation i.e. due to the increased number of neurons (8 hidden neurons), as their hardware evaluation is measured only in area consumption.

4.2 SNN Implementations

Both spike models contain exponential components in their synapses. Figure 2 is the hardware equivalent synapse and soma circuit for the LIF neuron. It can be observed that the synapse is modelled as a weighted first order recursive filter. The output of the synapse is fed into the soma. This value is compared with a threshold value, which is user definable. If greater than the threshold a spike is produced and the circuit will enter a refractory period.

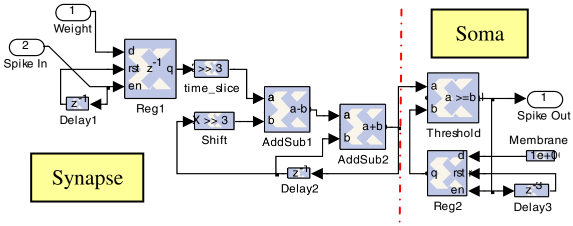


Fig. 2. LIF neuron circuit

Figure 3 below shows the SRM hardware equivalent synapse. In direct comparison to the LIF synapse, a multiplier, counter and other combinational logic are required as extra hardware for the circuit.

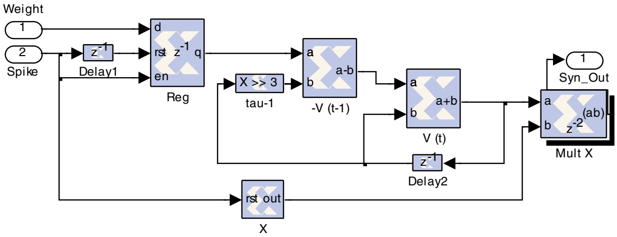


Fig. 3. SRM synapse circuit

Multipliers are expensive in hardware real estate terms therefore, the amount of circuitry required by the LIF synapse is smaller, see Tables 2 and 3. Real-time functional verification of the models in hardware was carried out by the Xilinx debug tool, ChipScope Pro 6.1i [7].

Table 2. Model hardware requirements on a Virtex II xc2v4000

Unit Synthesised	# CLB slices
LIF synapse	35
SRM synapse	195
2 input soma	20
MLP neuron	248

The network connectivity implemented in the designs is fully parallel. For the spike models, a synapse is needed for each physical connection and a multiplier for each MLP connection. Therefore, the number of combinational logic block (CLB) slices is given for each respective spike synapse, separate to the spike neuron (soma), as illustrated in Table 2.

The amount of resources consumed by the SRM synapse is clearly much higher than that of the LIF model. The MLP consumes the greatest amount of resources, but this contains the full neuron circuit. When the models are placed into their respective networks a more realistic conclusion can be drawn, as the network results of Table 3 illustrate.

Table 3. Network hardware requirements on a Virtex II xc2v4000

Network Synthesised	# CLB slices IRIS
LIF	1656
SRM	4448
MLP	2193

The SRM model significantly consumes the most hardware resources. This facilitates a clear comparison, highlighting the LIF model as the best choice in terms of resources consumed.

5 Conclusion

Four different models have been reviewed. All models were capable of solving the selected benchmark problem to a high accuracy. For the purpose of this paper these results highlight that the latest generation of SNs are just as capable, in comparison to the classical models, in solving a non-linearly separable problem. These factors coupled with the significant decrease in hardware real estate, suggests the LIF model as the most suitable model to implement on FPGAs.

Concerning spike models, the SRM is a function approximation of the more biologically realistic response of a neurons synapse. This investigation has demonstrated that this response does not provide any advantage over the less complex LIF response. This would therefore indicate that the emphasis lies in how the actual spike models communicate and not the complexity of post synaptic potential.

The presented work provides a valuable insight into NNs for neuroengineers and others involved in the selection of neuron models. A continuation of the research will involve the ability to solve more complex problems, including network robustness and a comparison to other implementations.

References

[1] W.S. McCulloch and W. Pitts "A logical calculus of the ideas immanent in nervous activity", BullMath. Biophy. 5:pp115-133, 1943.

[2] G. Wulfram and W. Werner "Spiking Neuron Models", Cambridge University Press, 2002.

[3] A. Belatreche, L. P. Maguire, T.M. McGinnity and Q, Wu, "A Method for the Supervised training of Spiking Neural Networks", IEEE Cybernetics Intelligence – Challenges and Advances CICA, Reading, UK, 2003.

[4] S. Johnston, G. Prasad, L.P. Maguire, T.M. McGinnity and Q. Wu, "A Design Flow for the Hardware Implementation of Spiking Neural Networks onto FPGAs", IEEE Cybernetics Intelligence – Challenges and Advances CICA, Reading, UK, 2003.

[5] H.K. Kwan, Simple sigmoid-Like activation function suitable for digital implementation, Electron. Lett 28(15)(1992)1379-1380.

[6] J.J. Blake, L.P. Maguire, T.M. McGinnity, B. Roche, L.J. McDaid The implementation of fuzzy systems, neural networks and fuzzy neural networks using FPGAs. Inform. Sci. 112(1-4)(1998)151-168

[7] http://www.xilinx.com/chipscope

HYDRA: From Cellular Biology to Shape-Changing Artefacts

Esben H. Østergaard[1], David J. Christensen[1], Peter Eggenberger[2], Tim Taylor[3], Peter Ottery[3], and Henrik H. Lund[1]

[1] AdapTronics group, Maersk Mc-Kinney Moller Institute for Production Technology, University of Southern Denmark
[2] Artificial Intelligence Laboratory, Department of Information Technology, University of Zürich
[3] Mobile Robotics Research group, Institute of Perception, Action and Behaviour, University of Edinburgh

Abstract. The HYDRA work provides insight into the exploitation of holistic behavioural and morphological adaptation in the design of new artefacts. The potential of the new design principle has been exemplified through the construction of robotic systems that can change morphology. Two prototype building block systems has been developed, HYDRON for a fluid scenario, and ATRON for a terrestrial scenario. In the HYDRON case, the individual module can perform 3D motion and is able to arrange in clusters of specific formation without the necessity of physical connections. In the ATRON case, the modules are individually simpler, attach through physical connections, and perform 3D motions by collective actions. Control mechanisms identified from cellular biology has been successfully transferred to the physical building blocks.

1 Introduction

The HYDRA project focuses on the design of building blocks for self-reconfigurable artefacts. The building blocks allow robust and efficient morphological development of artefacts, in order to allow end-users to design new artefacts in an easy manner. Inspired by biological principles, the HYDRA project realises engineering structures with the properties of differentiation and self-reconfiguration.

Investigations of biological principles reveal that the cell is an appropriate basis for this work, so we investigate building blocks modeled as cells. This leads to control mechanisms based on inspiration from cellular mechanisms such as cell division, cell motion, cell death, cell adhesion, change of cell shape, and cell differentiation and induction [6].

By exploring different possible building blocks in software and hardware development, the project defines physical building blocks that allow development of systems comprising hundreds of basic building blocks that exhibit self-assembly, self-repair, and shape-change. The potential of the new design standard has been exemplified through the construction of two robotic systems that can change morphology.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 275–281, 2005.

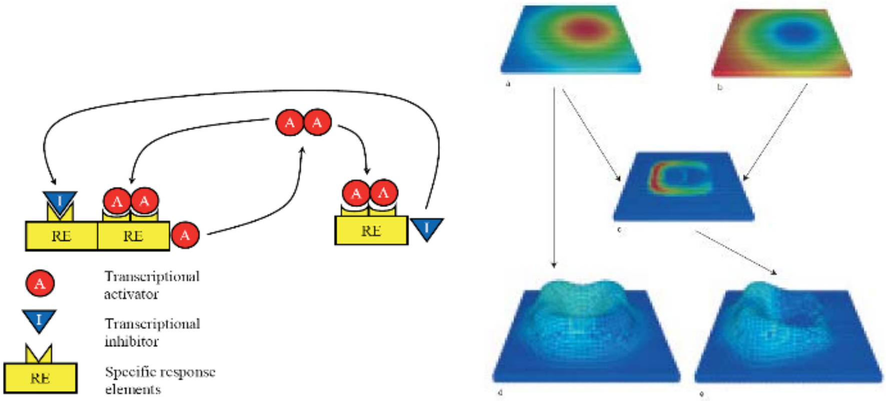

A
A
A
A
A
A
I
I
RE
RE
RE
A
Transcriptional activator
I
Transcriptional inhibitor
RE
Specific response elements
a
b
c
d
e

Fig. 1. Left: Example of gene regulation. Structural genes (A and I), are controlled by responsive elements (RE). A is here a transcription factor regulating its own synthesis and activating also inhibitor I. **Right:** Interactions of genes, morphogens and cellular physics. Physical strains of cells are controlled by the two chemical gradients, shown in a) and b). a) alone produces the shape in d), while their combination produces the shape shown in e). From [3].

The controlling mechanism found in simulation has been modified, implemented and tested as control for the physical building blocks. Especially, the abstract gradient-based control mechanisms from the simulations has proven useful to control real hardware systems. The HYDRA hardware includes 20 HYDRON modules for aquatic use and 100 ATRON modules for terrestrial use, both systems capable of self-reconfiguration in 3D.

2 Cellular Biology

The biological mechanisms of self-assembly and self-repair were investigated and modeled within an artificial evolutionary system in the context of cellular systems. Generic principles identified from these developmental mechanisms were used to implement control algorithms for the two HYDRA hardware platforms.

Having chosen the cell as our level of abstraction, our main task was to identify those developmental processes, which allowed an artificial evolutionary system to mimic growth and regeneration [4] in a cellular context. For this purpose, different developmental processes were simulated and explored by evolution. A set of basic cellular mechanisms was identified, which could be used to simulate a wide range of higher level mechanisms such as pattern generation or co-evolution of morphology and behaviour.

All the biological mechanisms that are essential for development, cell division, growth, differentiation, pattern formation and morphogenesis are mediated ultimately by proteins. They act either directly or as enzymes to produce other molecules. These proteins are encoded by genes, so development is controlled to a large degree by gene expression. The pattern of gene expression in the embryo determines where, when and in what quantity particular proteins are made and therefore governs the properties of each cell. Since proteins play such an important role for development, the mechanism ligand-receptor interactions was implemented, which can mimic many specific interactions among proteins abstractly. A receptor is usually a large protein, folded in a way to be able to recognize specifically a partner molecule. For a ligand (a signaling molecule) to be useful it must act selectively on particular targets such as gene regulators or receptors. This means that a receptor will only recognize ligands of a certain precise type and ignore closely related molecules. This principle of binding-site and ligand specificity can be found almost everywhere in multicellular organisms. This complementary specificity, which is based on the very exact molecular recognition properties of molecules, is central to explaining many of the phenomena of developmental biology.

The basic mechanisms identified in cells - cell division, cell death, cell adhesion, expression of receptors, and production of signaling molecules - were used to evolve and simulate higher-level processes such as cell differentiation, pattern generation, morphogenesis, growth of neural networks with inter-neuronal communication and co-evolution of morphology and behaviour. Figure 1 provides an overview of the simulated cellular mechanisms.

3 The HYDRON Module

A HYDRON unit is shown in figure 2. Each unit is roughly spherical, with a diameter of approximately 11 cm, suspended in water, and actuated in the horizontal plane. A HYDRON unit has four nozzles which expel water drawn through an impeller at the bottom of the unit when activated, and which are selected by a rotating collar. A syringe draws or expels water through the bottom of the unit to control unit buoyancy, and thereby actuate the unit along the vertical axis. Each units hull also supports a small set of switchable optical sensors and emitters capable of transmitting data over short ranges. Optical sensors and transmitters were chosen because they provide a simple and flexible underwater communication mechanism.

Simulation work on a Cellular Adhesion Molecule (CAM) based control approach [8] shows how this simple, biologically-inspired approach to decentralized multi-robot control can be used for forming a variety of spatial patterns, as shown in figure 2. This can be achieved in modules with limited capacity for communication and locomotion. The simulation results are also consistent with the predictions of Steinberg's Differential Adhesion Hypothesis for sorting of biological cellular aggregates [9]. The approach is flexible and robust, and the design of the controller permits easy transfer to the real HYDRON robots.

The combination of the CAM controller with the Genetic Regulatory Network (GRN) controller [10] shows that the GRN can be evolved to produce time-varying expression of CAMs on a robots (virtual) membrane in order to achieve specific behaviours. Especially for more complex tasks (such as react-

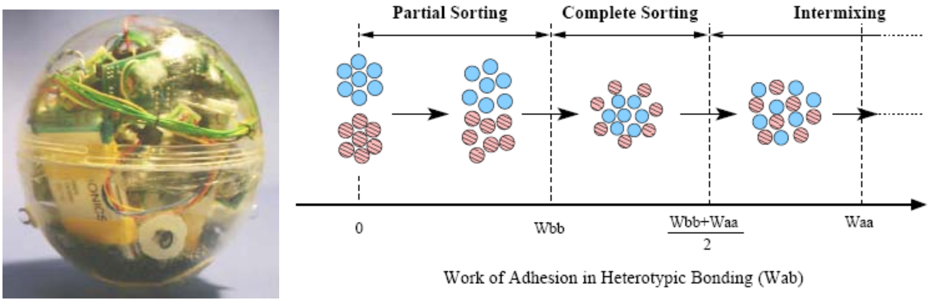


Fig. 2. Left: The HYDRON hardware. **Right:** Cellular Adhesion Control in 2D of a small group of simulated HYDRON modules. By changing the attractive and repulsive force between types of cells, different configurations can be achieved.

ing to an external signal, or producing differentiated behaviour from an initially homogeneous cluster), the evolutionary power of the combined GRN-CAM controller is able to produce better performance than could be achieved by either of the primitive controllers individually.

4 The ATRON Module

The ATRON modules, shown in figure 3 and further described in [5], are lattice based self-reconfigurable robot modules for 3D operation in land environments. Greatly simplified, an ATRON module is composed of two hemispheres joined together by a rotation mechanism

ATRON modules can connect using mechanical hooks which attach to an arrangement of bars on a neighbour, similarly adhesion proteins bind to ligands on the surface of an adjacent cell. On each half module, there are two female (bars) and two actuated male connectors (hooks). The novel mechanical connector design, ensures a strong and reliable connection. A module may communicate with neighbouring modules through IR communication.

When placed in the surface-centred cubic lattice structure, the modules can self-reconfigure to achieve different overall arrangements or movements. The shape allows one module to move to an adjacent hole in an otherwise fully packed structure (without colliding with other modules). Indeed, the design was guided by considerations on how to reduce control complexity of self-reconfiguration, while having a simple module design. However, compared to biological cells and the HYDRON module, the ATRON has very hard constraints on motion. Gravity related restrictions include static stability of the configuration, not exceeding motor torque limits, and obeying structural stiffness. Also, care should be taken during self-reconfiguration to avoid module collisions and to maintain structural connectivity. The system also has similarities with cellular systems, in that a cluster of ATRONs is composed of many identical semi-autonomous units. The question is how the principles from biology should be transferred to suit the hard constraints of the ATRON hardware.

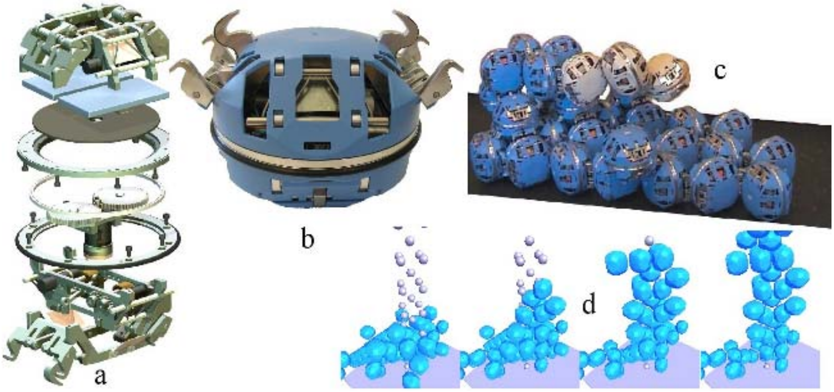


Fig. 3. a) Exploded view of the ATRON CAD drawing, and **b)** a photo of the final hardware. **c)** Experiment on the ATRON platform showing three white modules as a meta-module in the process of migrating on a substrate of modules. **d)** The simulated migration of ATRON meta-modules permits approximation of the target shape defined as attraction points.

In [1] we describe a GRN-style system, where each module is controlled by simple rules of the form $\{precondition, action\}$, based on the local neighbourhood and the modules actuators. The activation of rules is determined by the hormone gradients, such that the effort of each individual module is orchestrated from organism scale chemical gradients. Work along these lines have shown that such simple rules based on local morphology can generate scalable and robust behavior at organism level, such as cluster walk, obstacle avoidance and terrain following [7].

Chemical gradients are known to guide migrating cells [6]. In order to achieve a desired self-reconfiguration, this mechanism has been transferred to the ATRON platform [2]. ATRON modules can use their IR-based neighbour-to-neighbour communication to simulate a gradient that attracts other modules. Migration is achieved using a meta-module consisting of three modules, see 3 c). Such a meta-module has the ability to move relatively freely on the substrate of other modules. Meta-modules emerge from the structure of modules, migrate based on cues from its environment, and die when reaching their target, once again becoming part of the substrate. This approach is similar to the division and death of biological cells. The combination of gradients and migrating meta-modules enable the system to change its shape, see figure 3 d), and thereby adapt to the required functionality of the system.

5 Conclusion

The HYDRA work provides insight into the exploitation of holistic behavioural and morphological adaptation in the design of new artefacts.

Investigations of biological principles have revealed that the cell is an appropriate level of abstraction, so the building blocks are modeled as cells. This leads to control mechanisms based on inspiration from cellular mechanisms such as cell division, cell motion, cell death, cell adhesion, change of cell shape, and cell differentiation and induction. HYDRA simulation work shows how such mechanisms can be used to control the morphological creation of forms. The controlling mechanism found in simulation has been modified, implemented and tested as control for the physical building blocks. In addition, gradient-based control mechanisms have been abstracted from the simulations.

20 HYDRON modules and 100 ATRON modules have been produced, and experiments has been been performed on these modules. The robotic system's ability to reconfigure in 3D has been high on the agenda. The successful outcome of the project has been videotaped for presentation and the physical modules demonstrations have been showcased at various PR events and scientific events worldwide.

Acknowledgment

This work was funded by EU's Information Society Technologies Programme, IST-2001-33060, HYDRA, Future and Emerging Technologies.

References

1. D. Brandt and E. H. Østergaard. Behaviour subdivision and generalization of rules in rule based control of the atron self-reconfigurable robot. In *Proceedings of International Symposium on Robotics and Automation (ISRA)*, pages 67–74. Secretaria de Educación Pública, 2004.
2. D. J. Christensen, E. H. Østergaard, and H. H. Lund. Metamodule control for the ATRON self-reconfigurable robotic system. In *Proceedings of the The 8th Conference on Intelligent Autonomous Systems (IAS-8)*, pages 685–692, Amsterdam, 2004.
3. P. E. Hotz. Genome-physics interaction as a new concept to reduce the number of genetic parameters in artificial evolution. In *Proceedings of The IEEE 2003 Congress on Evolutionary Computation (CEC2003)*, 2003.
4. P. E. Hotz. Exploring regenerative mechanisms found in flatworms by artificial evolutionary techniques using genetic regulatory networks. In *Congress on Evolutionary Computation (CEC)*, Canberra, 2004.
5. M. W. Jørgensen, E. H. Østergaard, and H. H. Lund. Modular ATRON: Modules for a self-reconfigurable robot. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2068–2073, 2004.
6. H. Meinhardt. A model for pattern formation of hypostome, tentacles, and foot in hydra: how to form structures close to each other, how to form them at a distance. *Developmental Biology*, 157(2):321–333, June 1993.
7. E. H. Østergaard, K. Tomita, and H. Kurokawa. Distributed metamorphosis of regular M-TRAN structures. In *Proceedings of 6th International Symposium on Distributed Autonomous Robotic Systems (DARS), Toulouse, France, June 23-26*, pages 161–170, 2004.

8. P. Ottery and J. Hallam. Steps toward self-reconfigurable robot systems by modelling cellular adhesion mechanisms. In F. Groen, N. Amato, A. Bonarini, E. Yoshida, and B. Kröse, editors, *Proceedings of the The 8th Conference on Intelligent Autonomous Systems (IAS-8)*, pages 720–728, Amsterdam, Mar. 2004. IOS Press.
9. M. S. Steinberg. Reconstruction of tissues by dissociated cells. *Science*, 141:410–408, 1962.
10. T. Taylor. A genetic regulatory network-inspired real-time controller for a group of underwater robots. In F. Groen, N. Amato, A. Bonarini, E. Yoshida, and B. Kröse, editors, *Proceedings of the The 8th Conference on Intelligent Autonomous Systems (IAS-8)*, pages 403–412, Amsterdam, Mar. 2004. IOS Press.

The CIRCE Head: A Biomimetic Sonar System

Herbert Peremans and Jonas Reijniers*

Universiteit Antwerpen, Departement MTT,
Prinsstraat 13, Antwerpen 2000, Belgium
{herbert.peremans, jonas.reijniers}@ua.ac.be

Abstract. The investigation of biomimetic perception is part of the broader research field of biorobotics which aims to investigate biological sensorimotor control systems by building robot models of them. The intent of bio-inspired engineering is to distil from the principles found in successful nature-tested mechanisms specific "crucial functions" [1]. In addition to its interest for engineers, we believe that the CIRCE biomimetic sonar system, a robotic system which reproduces, at a functional level, the echolocation system of bats, is a unique experimental tool for biologists to systematically investigate how the world is not just perceived but actively explored by bats.

1 Introduction

Despite an ever increasing speed of technological achievements, biological systems still outperform technical systems in many ways. Part of the explanation why biological systems show superior performance in less structured environments lies in nature's ability to create well-integrated systems comprised of many components, each of which contains evolutionarily embedded knowledge about the particular tasks it performs. In particular, the bats sonar system outperforms man-made sonar technology in its ability to support autonomous navigation as well as a variety of other tasks in demanding natural environments. As such, bats are a clear example of how the use of intertwined acoustic and neural signal processing with various feedback control loops spanning these stages can lead to unparalleled performance.

In order to obtain a better understanding of the role of the different sonar system components and their interplay in a highly coupled system, an attempt is made to reproduce this system -at a functional level- with a robotic model: the CIRCE head. The primary objective for this biomimetic bat head is to provide hypotheses for biological function. The applicability of these hypotheses can then be tested in experimental work performed with real animals. Besides this objective in basic science, a biomimetic sonar system that would exhibit a bat′s navigation and prey-capture skills would lead to orders of magnitude improvements in all areas where technical sonar systems are used nowadays.

* This work is supported by the European Union, IST-LPS Initiative (IST-2001-35144). The CIRCE work reviewed here is the collective effort of the consortium.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 283–288, 2005.

2 The CIRCE Biomimetic Bathead

The starting point of the CIRCE project is that the processing required to turn acoustic field information into useful information about the environment is done both in the acoustic and in the neural domain. Hence, it is important to accurately model the acoustic field around the bats head. Consequently, the ratio of the size of the most significant head structures and the sound wavelength has to be kept constant. Hence, the CIRCE head has to combine in a small space ($\oslash$=4-8 cm) actuated antennae for emission and reception, transducers (one transmitter, two receivers) and signal conditioning electronics in a way that meets a demanding set of functional specifications. This results in various technological challenges, the remainder of this section describes our current status in confronting those.

2.1 Biomimetic Antennae Shapes

The approximately 1000 species of bats have evolved a highly diverse set of baffle shapes which surround the sites of sound emission (mouth or nose) and reception (ears). These shapes act as beamforming antennas, optimized to suit the different sonar tasks these animals have to solve. As detailed shape information on these structures was not available prior to the CIRCE project, we have developed a micro-tomography based toolset that can be used to build 3D-CAD model descriptions of the morphology of those shapes (s. Fig. 1(a)) for different species of bats.

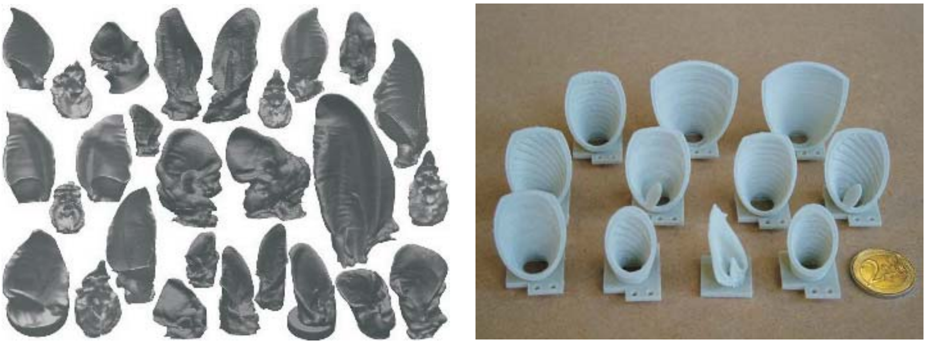


Fig. 1. Pinna shapes: (a) Scanned bat pinnae (micro-CT), (b) Nylon pinnae (selective laser sintering)

Next, a functional analysis of these shapes was performed making use of a, time-domain, finite element numerical simulation of the diffraction effects around them. The results, after transformation into the frequency domain, are used to estimate the ear's directivities. These simulations have shown that the tragus, a frontal flap which can be very prominent in certain bat species, has a systematic effect on the shape of the directivity pattern. Furthermore, in experimenting with simpler shape models it has become apparent, that there is an interaction between the effect of the tragus and the presence of a surface ripple on the inner surface of the pinna.

To validate these results a family of artificial ear shapes was designed and fabricated (s. Fig. 1(b)) which use the basic shape of an obliquely truncated horn proposed as an idealized pinna model [2] and augments it with a flap modelling the tragus and a ripple pattern. The fabricated ear shapes are made out of Nylon 7000 using a rapid prototyping tool (selective laser sintering) and are mounted on the transducer using a snap-fit allowing for easy switching between them.

2.2 Transducer Technology

Bats are able to produce high-energy sonar pulses in a very efficient manner. In addition, bats can rely on the superb sensitivity of the mammalian hearing system on the receiver side. Transducer specifications based on reasonable approximations of the proven capabilities of biological sonar systems: flat response over 20-200 kHz, transducer diameter < 1-1.5 cm, generated sound pressure level of 80-100 dB at 1m, equivalent acoustic noise level < 45 dB can not be achieved by commercially available in-air ultrasonic transducers. Hence, new transducer technology is required.

EMFi is a polypropylene film, thickness 30-70 μm, that has a cellular structure which results from its manufacturing process [3]. During manufacturing voids created in the non-polar material are internally charged by inducing microplasma discharges. The resulting build up of internal charge at the surfaces of the voids turns the latter into macroscopic dipoles. EMFi foils are light, thin and flexible making them easy to process. In addition, the good impedance match between EMFi and air guarantees a much more efficient transduction than is possible with standard piezo materials.

A prototype comprising a patch of EMFi (15x15mm) mounted on a copper plate as backing (s. Fig. 2(a)) was used for investigating the properties of the EMFi material. Actuated with an a.c. voltage of up to 600Vpp, the films deflection (thickness mode oscillation) was measured as a function of frequency (s. Fig. 2(b)) clearly showing the broadband capabilities of this technology.

The reciprocal nature of the transduction mechanism allows the use of the same material for designing broadband receivers (Fig. 2(c)). In order to minimize the size of the receiver, a low noise preamplifier circuit is implemented on the bottom PCB of the sandwich structure shown in Fig. 2(d). One of the pinna shapes described in the previous section is mounted on top of this broadband receiver (s. Fig. 2(e)).

2.3 Actuation

Many bat species posses the capability to orient/deform their outer ears at will, enabling them to adapt their perception apparatus to the task at hand.

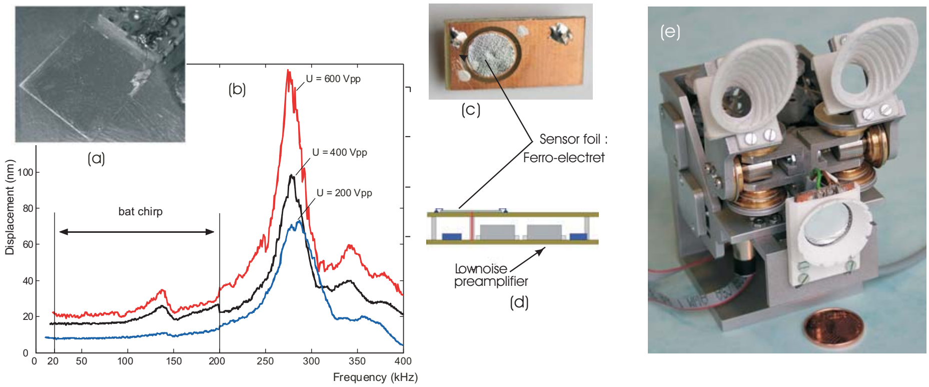


Fig. 2. (a) EMFi transmitter (15x15 mm); (b) frequency response of transducer surface displacement. (c) The sandwich structure of the ultrasonic receiver. (d) The EMFi-transducer foil acting as a receiver is glued to the top layer of the sandwich. (e) The biomimetic head prototype.

The actuation subsystem of the biomimetic head (s. Fig. 2(e)) needs to provide the means for duplicating the essential features of such real bat head actuation capabilities in a volume comparable to that of a real bat head (sphere ⊘=40-80 mm). In particular, we have chosen independent panning and tilting of each of the two outer ears (specifications: range=60°, bandwidth=1-10Hz, accuracy $< 0.1°$) for reproduction in the biomimetic bat head.

The prototype design which meets all specifications makes use of a differential architecture combining the movements of two motors to move each ear with the required two rotational degrees of freedom. A very flexible 0.6 mm stainless steel cable is woven through the differential pulleys in such a way that the satellite pulley performs a pan movement when both motor pulleys are driven in the same sense and that it performs a tilt movement when the motor pulleys are driven in opposite senses. The pre-tensioned cable system exhibits zero-backlash behaviour. Combined with especially designed zero-backlash connections from motor shaft to motor pulley and from motor pulley to cable, this allows a determination of the exact position (pan and tilt) of the mounted pinna at the motorside.

2.4 Neuromimetic Signal Processing

The bat's auditory system is structured in the same way as that of other mammals [4]. In the cochlea, transduction from sound stimuli into neural activity is performed by the inner hair cells (IHC). Each of the 700-2200 [5] non-spiking inner hair cells synapses with up to 20 spiral ganglion cells of which there are 13000-55000. The latter are spiking neurons and their axons form part of the auditory nerve that transfers their spiking responses into the bat's brain.

We have preferred a simple, efficient cochlear model, which selectively reproduces functionally significant features of the neural code that emphasizes a quantitatively correct representation of the code in the auditory nerve as this allows for the study of code properties on a neural population level. The CIRCE neuromimetic cochlea as implemented on Xilinx Virtex II FPGA hardware [6] consists of a pipelined parallel architecture for a filterbank containing 750 bandpass filters with a frequency span from 20-200 kHz per ear. Demodulation is performed in each frequency channel by a combination of half-wave rectification and low-pass filtering. After automatic gain control these analog signals are converted into a spike code representation by applying multiple thresholds to each frequency channel output.

3 System Evaluation in Biosonar Tasks: 2D Target Localization

Observing biosonar systems shows that a large amount of information can be extracted from a single measurement [7]. In addition, bat sonar is an active system, i.e. the bat can control its perception apparatus in such a way as to simplify subsequent feature extraction.

We reconstruct the environment, i.e. localize the reflecting targets, by comparing the returned echoes with predefined spectral templates corresponding with echoes from different angles (s. Fig. 3).

Fig. 3(a)-(d) shows how echoes arising from spatially separated point reflectors in a 2D environment do indeed reveal the position of the reflectors through

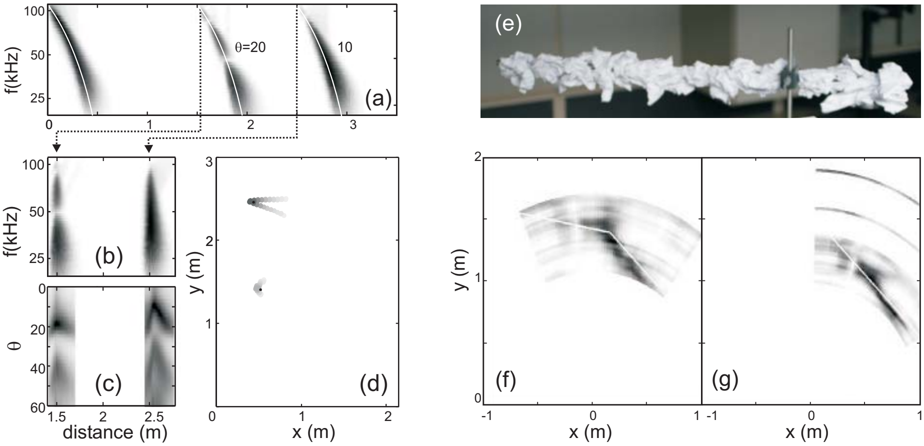


Fig. 3. (a) Cochlear time-frequency representation of picked up call + echoes from two spatially well separated reflectors. (b) Cochlear output after call induced delay variation has been compensated for. (c) The correlation calculated for the cochlear output shown in (a). (d) The estimated positions (larger gray dots) and the true positions (black dots) of the reflectors. (e) A 2D extended reflector (multiple reflecting facets). Reconstructed environment (f) rotation sonarhead = −5°, (g) rotation sonarhead = −25°.

correlation with the angular templates, i.e. the normalised cochlear output for an isolated reflector at a particular angle θ. Various extensions of this powerful approach can be considered. In Fig. 3(e)-(g) we show how the same approach can be applied to more complex distributions of reflectors, as models of natural vegetation. In [8] we show how active control of the ear configuration allows binaural sonar systems to eliminate most of the interpretation errors that occur in such more realistic environments with a monaural system. In [9] it is shown how the same mechanism can be extended to 3D localization of isolated targets for a binaural sonar system.

4 Conclusion

The biorobotics approach, in particular the biomimetic CIRCE head described in this paper, provides the opportunity to tap into the large pool of biosonar experience built by millions of years of natural selection. Hence, we propose it presents a valuable extra source of information for robotic sensor designers attempting to build the next generation of more advanced robotic echolocation systems.

In addition to the technological innovations resulting from the building/using of the biomimetic bat head, as described here, the head makes possible the systematic study of active sensing strategies evolved by bats for coping with the complex echoes arising while flying/hunting in the foliage.

References

1. NASA: http://www.nasatech.com/Briefs/May03/NPO21142.html (2004)
2. Fletcher N.H. and Thwaites S.: Obliquely truncated simple horns: idealized models for vertebrate pinnae. Acustica, (1988) 194-204.
3. Bauer S., Gerhard-Multhaupt R. and Sessler G.M.: Ferroelectrets: Soft Electroactive Foams for Transducers. Physics Today, February, (2004) 37-43.
4. Pickles J.O.: An Introduction to the physiology of hearing. Academic Press, London, (1982).
5. Vater M.: Cochlear physiology and anatomy in bats. In: Animal Sonar Processes and Performance, P.E. Nachtigall and P.W.B. Moore (Eds.), (1988) 225-240.
6. Clarke C., Qiang L., Peremans H. and Hernández Á.: FPGA implementation of a neuromimetic cochlea for a bionic bat head. LNCS 3203, J. Becker, M. Platzner and S. Vernalde (Eds.), (2004) 1073-1075.
7. Suga N.: Cortical computational maps for auditory imaging. Neural networks, 3, (1990) 3-21.
8. Reijniers J. and Peremans H.: Towards a theory of how bats navigate through foliage. In: Proc. of the 8th Int. Conf. on the Simulation of Adaptive Behaviour, (2004) 77-86.
9. Peremans H., Walker A. and Hallam J.: 3d object localisation with a binaural sonar-head, inspirations from biology. Proc. of the IEEE Int. Conf. on Robotics and Automation, (1998) 2795-2800.

Tools for Address-Event-Representation Communication Systems and Debugging

M. Rivas, F. Gomez-Rodriguez, R. Paz, A. Linares-Barranco,
S. Vicente, and D. Cascado

Departamento de Arquitectura y Tecnología de Computadores, Universidad de Sevilla,
Av. Reina Mercedes s/n, 41012-Sevilla, Spain
{mrivas, gomezroz, rpaz, alinares, satur, danic}@atc.us.es
http://www.atc.us.es

Abstract. Address-Event-Representation (AER) is a communications protocol for transferring spikes between bio-inspired chips. Such systems may consist of a hierarchical structure with several chips that transmit spikes among them in real time, while performing some processing. To develop and test AER based systems it is convenient to have a set of instruments that would allow to: generate AER streams, monitor the output produced by neural chips and modify the spike stream produced by an emitting chip to adapt it to the requirements of the receiving elements. In this paper we present a set of tools that implement these functions developed in the CAVIAR EU project.

1 Introduction

Address-Event-Representation (AER) was proposed in 1991 by Sivilotti [1] for transferring the state of an array of neurons from one chip to another. It uses mixed analog and digital principles and exploits pulse density modulation for coding information. The state of the neurons is a continuous time varying analog signal.

Fig. 1 explains the principle behind the AER. The emitter chip contains an array of cells (like, e.g., an imager or artificial retina chip) where each pixel shows a state that changes with a slow time constant (in the order of milliseconds). Each pixel includes an oscillator that generates pulses of minimum width (a few nanoseconds). Each time a pixel generates a pulse (called "event"), it communicates with the periphery and its address is placed on the external digital bus (the AER bus). Handshaking lines (Acknowledge and Request) are used for completing the communication.

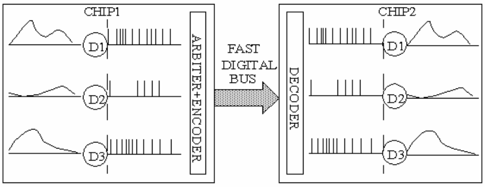


Fig. 1. AER inter-chip communication scheme

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 289 – 296, 2005.

In the receiver chip the pulses are directed to the pixels or cells whose address was on the bus. This way, pixels with the same address in the emitter and receiver chips will "see" the same pulse stream. The receiver cell integrates the pulses and reconstructs the original low frequency continuous-time waveform.

Transmitting the pixel addresses allows performing extra operations on the images while they travel from one chip to another. For example, inserting memories (e.g. EEPROM) allows transformations of images.

There is a growing community of AER protocol users for bio-inspired applications in vision and audition systems, as demonstrated by the success in the last years of the AER group at the Neuromorphic Engineering Workshop series [2]. The goal of this community is to build large multi-chip hierarchically structured systems capable of performing complicated array data processing in real time. The CAVIAR EU project has the objective to demonstrate this technology by targeting and following a moving ball. The planned AER system under CAVIAR uses the following AER chips: one Retina, four Convolutions, four Winner-Take-All (Object) and one Learning chip. To make possible the right communication of these chips and for debugging purposes it is essential to have a set of instruments that would allow to:

- Sequence: Produce synthetic AER event streams that can be used as controlled inputs while testing and adjusting a chip or set of chips.
- Monitor: Observe the output of any element in the system.
- Map: Alter the stream produced by an emitter and send the modified stream to a receiver

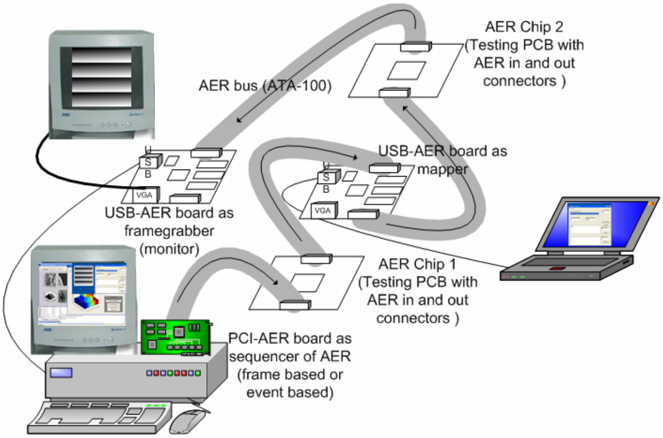


Fig. 2. AER tools usage scenario

For these purposes we have designed and implemented two different instruments: a PCI board capable of sequencing and monitoring events at a rate of over 15Mevents/s and a versatile board that can be used for sequencing, monitoring and mapping. This last board can be used either in a stand alone mode or connected to an external computer through a USB bus. A possible scenario for these tools is shown in Fig. 2 where a computer with a PCI-AER board produces output for AER chip1. The output from this chip is remapped by a USB-AER board and fetched to AER chip 2. The stream produced by chip 2 is monitored by another USB-AER board which can send its output directly to a VGA monitor or to a computer through USB bus.

To be useful for debugging an AER tool should be able to receive and also send a long sequence of events interfering as little as possible with the system under test.

As neurons have the information coded in the frequency (or timing) of their spikes, the pixels that transmit their address through an AER bus also have their information coded in the frequency of appearance of those addresses in the bus. Therefore, inter-spike-intervals (ISIs) are critical for this communication mechanism. Thus, a well designed tool shouldn't modify the ISIs of the AER.

Sections 2 and 3 present the PCI and the USB solutions and their applications in AER testing. Section 4 presents a Switch-AER. Section 5 presents a small version of a USB board with lower capabilities and performance, but very simple to use. And finally in section 6 we conclude with two examples of connectivity.

2 PCI-AER Interface

Before the development of our tools the only available PCI-AER interface board was developed by Dante at ISS-Rome (See [3]). This board is very interesting as it embeds all the requirements mentioned above: AER generation, remapping and monitoring. Anyhow its performance is limited to 1Mevent/s approximately. In realistic experiments software overheads reduce this value even further. In many cases these values are acceptable but, currently many address event chips can produce (or accept) much higher spike rates.

As the Computer interfacing elements are mainly a monitoring and testing feature in many address event systems, the instruments used for these purposes should not delay the neuromorphic chips in the system. Thus, speed requirements are at least 10 times higher than those of the original PCI-AER board. Several alternatives are possible to meet these goals: extended PCI buses, bus mastering or hardware based Frame to AER and AER to Frame conversion.

The previously available PCI-AER board uses polled I/O to transfer data to and from the board. This is possibly the main limiting factor on its performance. To increase PCI bus mastering is the only alternative. The hardware and driver architecture of a bus mastering capable board is significantly different, and more complex, than a polling or interrupt based implementation.

The theoretical maximum PCI32/33 bandwidth is around 133Mbytes/s. This would allow for approximately 44Mevent/s considering 2 bytes per address and two bytes for timing information. Realistic figures in practice are closer to 20Mbyte/s. Thus, in those cases where the required throughput is higher a possible solution is to transmit the received information by hardware based conversion to/from a frame based representation. Although this solution is adequate in many cases, there are circumstances where the developers want to know precisely the timing of each event, thus both alternatives should be preserved.

The physical implementation of all the steps is equal. They differ in the VHDL FPGA code and in the operating system dependent driver. The first design was a VIRTEX based board which was completely redesigned after the first tests. It was established that most of the functionality, demanded by the users, could be supported by the smaller devices in the less expensive SPARTAN-II family. The Spartan Version of the board is shown in Fig. 3.

Currently a Windows driver that implements bus mastering is being tested. The Linux version with bus mastering is still under development. An API that is compatible, as much as permitted by the different functionality, with that used in the current PCI-AER board has been implemented. MEX files to control the board from MATLAB have also been developed.

Current performance of PCI-AER board is around 15 Mevents/second using PCI mastering capabilities.

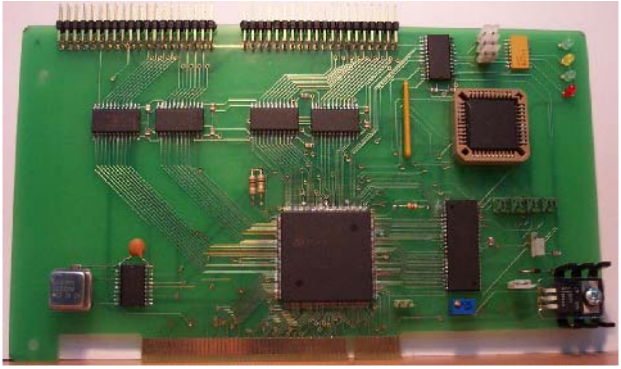


Fig. 3. CAVIAR PCI-AER board

3 USB-AER

The CAVIAR PCI-AER board can perform Address Event sequencing and monitoring functions but has no hardware mapping capabilities. Although software based mapping is feasible a specific device for this purpose is needed if we want to build AER systems that can operate without requiring any standard computer. This standalone operating mode requires to be able to load the FPGA and the mapping RAM from some type of non volatile storage that can be easily modified by the users. MMC/SD cards used in digital cameras are a very attractive possibility. However in the development stage the users prefer to load the board directly from a computer and, for this purpose USB seems the most suitable solution.

Many AER researchers would like to demonstrate their systems using instruments that could be easily interfaced to a laptop computer. This requirement can also be supported with the USB-AER board as it includes a relatively large FPGA that can be loaded from MMC/SD or USB, a large SRAM bank and two AER ports. Thus the board can be used also as a sequencer or a monitor. Due to the bandwidth limitations of full speed USB (12Mbit/s) hardware based event to frame conversion is essential in this board for high, or even moderate, event rates.

The USB-AER board is based around a Spartan-II 200 Xilinx FPGA, with a 512K*32 12ns SRAM memory bank. The board uses a Silicon Laboratories

C8051F320 microcontroller to implement the USB and the MMC/SD interface. A simple VGA monitor interface is also provided to allow the board to act as a monitor (frame grabber).

The board will act as a different device according to the module that is loaded in the FPGA either through a MMC/SD card or from the USB bus. Currently the following Modes are implemented:

- Mapper: 1 event to 1 event and 1 event to several events.
- Monitor (frame-grabber): using either USB or VGA as output. For the VGA output there are two possibilities: B/W VGA, using the VGA connector of the board. And Gray VGA, using a VGA-DAC board connected to the out-AER connector of the board.
- Sequencer: based on hardware frame to AER conversion using the Random or Exhaustive methods [4][5][6]. Can produce up to 25 Mevents/second. (40 ns per event).
- Datalogger: allows to capture sequences of up to 512K events with timestamps and send them to the PC offline through USB bus.
- Player (under development): to play up to 512Kevents with their timestamps.

These two modules are very interesting when a researcher wants to use the output stream produced by a chip from another researcher (probably in other country) as input to his or her chip.

This new board was interfaced in Telluride 04 [7] to the current version of the CAVIAR retina and to an imager developed at JHU. Later in the CAVIAR meeting in September 04 it was interfaced to the remaining project chips. The USB-AER board is shown in Fig. 4.

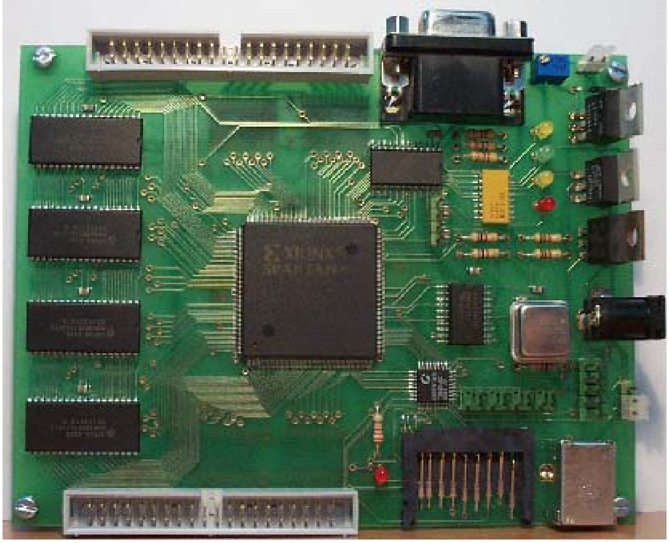


Fig. 4. USB-AER Board

A simple interface to control this board is available under windows. It allows loading modules into the FPGA, uploading or downloading data to the FPGA, and showing the received images when the board acts as a monitor. There is also available a MATLAB interface that support the same functionality.

A Linux driver for the USB-AER is currently under test. With this driver the USB-AER board can be easily integrated with several MATLAB applications developed at INI [8].

4 AER-Switch Board

A 4 to 1 and 1 to 4 AER-switch is presented in this paper. This board allows:

- The connection of more complex AER systems.
- An easier debugging by inserting PCI-AER or USB-AER board without modifying the structure of the global system to be tested.

This board has a CPLD as a communication centre, that manages the different modes and controls asynchronously the protocol lines. It can work in 2 different modes: 4 input, 1 output mode and 1 input, 4 output mode, both in unicast mode (selecting one output) or broadcast mode. This functionality should be configured by jumpers. There are 5 different AER ports, where one of them works always as an output, and another as an input. The others three are bidirectional. Fig. 5 shows the current version of this board.

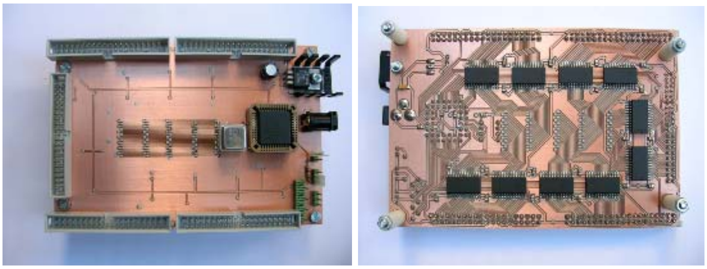


Fig. 5. AER-Switch Board

5 Mini-USB Board

For those tests or applications where it is not needed high speed performance, a small version of the USB board is available. This one doesn't have FPGA, nor MMC/SD card. This board can be connected to the PC through the USB bus, and all the functionality (Monitor or Sequencer) has to be programmed into the microcontroller under C code. Fig. 6 shows the current version of this board. The board has been developed also under CAVIAR project by INI partner and authors.

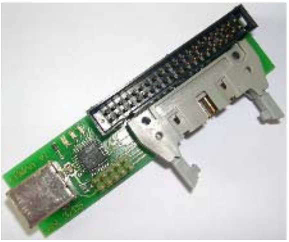


Fig. 6. AER-Switch Board

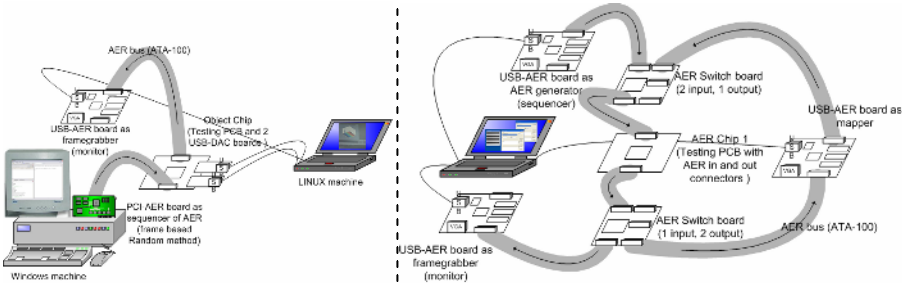

AER bus (ATA-100)
USB-AER board as framegrabber (monitor)
Object Chip (Testing PCB and 2 USB-DAC boards)
LINUX machine
PCI AER board as sequencer of AER (frame based Random method)
Windows machine
USB-AER board as AER generator (sequencer)
AER Switch board (2 input, 1 output)
USB-AER board as mapper
AER Chip 1 (Testing PCB with AER in and out connectors)
AER Switch board (1 input, 2 output)
AER bus (ATA-100)
USB-AER board as framegrabber (monitor)

Fig. 7. Two demonstration Scenarios

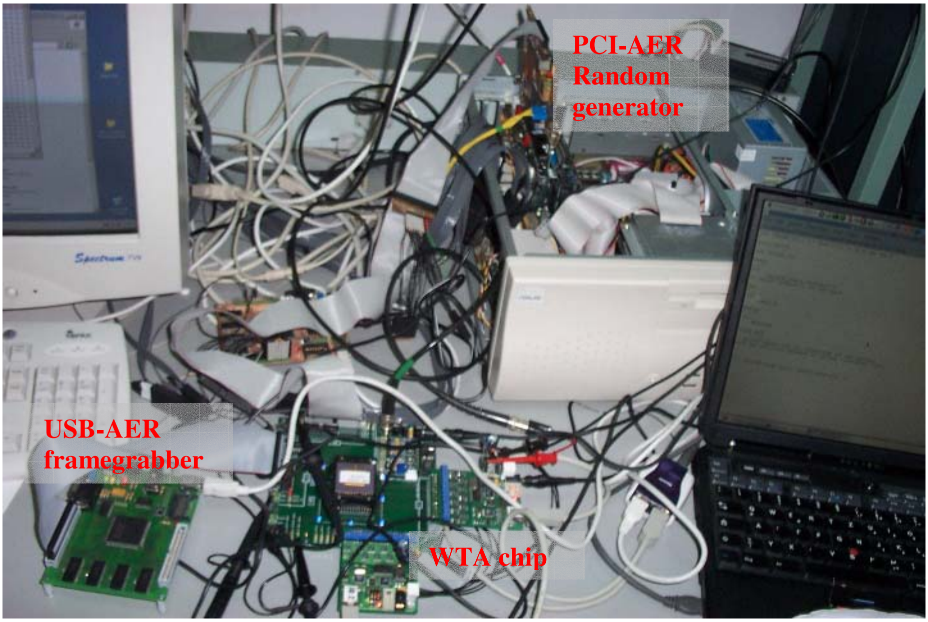


Fig. 8. Scenario Photograph

6 Conclusions

A set of tools has been developed that allow efficient testing and demonstration of address event based systems. Two demonstration scenarios are shown in Fig. 7. In the left case a PCI-AER board is generating a stream of events from a digital frame, using a hardware synthetic method. This sequence is used to feed a WTA filter chip, developed at INI. The output of the WTA chip is captured using a USB-AER board configured as a frame-grabber. A photograph of this scenario is shown in fig. 8.

On the right a USB-AER is working as a frame to AER sequencer to feed a AER chip. This chip receives also the transformed output of another AER chip using the AER-Switch. The output of the second chip can be viewed in the laptop screen, using another USB-AER as a monitor.

In this scenario only the presented tools are shown. In real world cases the tools are used to evaluate or tune neural chips. In the CAVIAR project the chips have been interfaced to two different retinas, a convolution chip, a winner take-all (object) chip and a learning chip.

Acknowledgements

This work was in part supported by EU project IST-2001-34124 (CAVIAR), and Spanish project TIC-2003-08164-C03-02 (SAMANTA).

References

1. M. Sivilotti, Wiring Considerations in analog VLSI Systems with Application to Field-Programmable Networks, Ph.D. Thesis, California Institute of Technology, Pasadena CA, 1991.
2. A. Cohen, R. Douglas, C. Koch, T. Sejnowski, S. Shamma, T. Horiuchi, and G. Indiveri, Report to the National Science Foundation: Workshop on Neuromorphic Engineering, Telluride, Colorado, USA, June-July 2001. [www.ini.unizh.ch/telluride]
3. V. Dante. “PCI AER Adapter board”, http://neural.iss.infn.it/Board/draft.html.
4. A. Linares-Barranco. Estudio y evaluación de interfaces para la conexión de sistemas neuromórficos mediante Address- Event-Representation. Ph.D. Thesis, University of Seville, Spain, 2003
5. A. Linares-Barranco, R. Senhadji-Navarro, I. García-Vargas, F. Gómez-Rodríguez, G. Jimenez and A. Civit. Synthetic Generation of Address-Event for Real-Time Image Processing. ETFA 2003, Lisbon, September. Proceedings, Vol. 2, pp. 462-467.
6. Linares-Barranco, A.; Jimenez-Moreno, G.; Civit-Ballcels, A.; Linares-Barranco, B.; On synthetic AER generation. ISCAS '04. Proceedings of the IEEE 2004 May 2004 Pages:V-784 - V-787 Vol.5
7. Avis Cohen, et.al., Report on the 2004 Workshop On Neuromorphic Engineering, Telluride, CO. June - July , 2004 [www.ini.unizh.ch/telluride/previous/report04.pdf]
8. M. Oster, Serverbased Software Architecture for AER systems [http://www.ini.unizh.ch/~mao/AerSoftware/SoftwareOverview.pdf]

New Ears for a Robot Cricket

Ben Torben-Nielsen[1], Barbara Webb[2], and Richard Reeve[2]

[1] Institute for Knowledge and Agent Technology,
Universiteit Maastricht, The Netherlands
B.Torben-Nielsen@cs.unimaas.nl

[2] Institute of Perception, Action and Behaviour, School of Informatics,
University of Edinburgh, UK

Abstract. Cricket females perform phonotaxis towards the specific sounds produced by male crickets. By means of a well-tuned peripheral auditory system the cricket is able to extract directional information about the sound, and the neural system can recognize the species specific characteristics of the song. Crickets use sounds coming from at least four body openings to derive the directional signal. A new artificial four-input apparatus which implements a detailed model of the cricket's peripheral auditory system, together with our most recent model of the neural system are used for this study. A series of experiments is conducted to validate the new auditory input device and to benchmark the neural model. This study shows that (i) the new auditory input device provides the robot with realistic inputs, and, (ii) most behavioral features ascribed to previous neural models still hold for the new neural model. The relevant differences are discussed.

1 Introduction

Female crickets are known to track a calling song made by male crickets as part of the mating process. The cricket's auditory system is evolutionarily tuned for this purpose and enables the cricket to efficiently recognize and localize the calling song [1]. The peripheral auditory system consists of two ear drums (tympanal membranes) located on the forelegs and connected through a system of tracheal tubes. One extra sound opening on each side of the cricket body (the spiracle) is also connected to the tracheal tubes. The trachea allow sound coming from one side to travel to the other side (Figure 1, left). This way, both ear drums receive "sound on the external surface and sound on the internal surface coming from at least three different pathways" [2]. The amplitude of vibrations of the tympanal membrane depends on the phase of the sounds converging on either side of the ear drum. The relative phases depend on the frequency and direction of the sound [1].

Our previous cricket robots were equipped with an auditory input apparatus that *functionally mimicked* the peripheral auditory system, i.e. acting as a pressure difference receiver [3] so that, for sounds of a specific carrier frequency, the gradient to the sound source can be determined. However, it used only two

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 297–304, 2005.

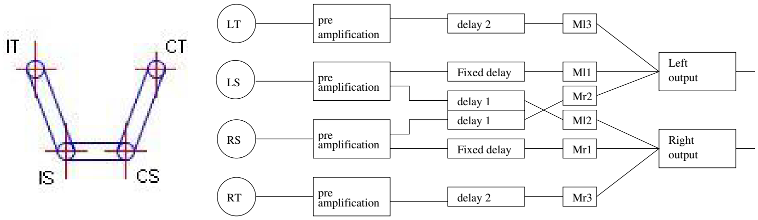


Fig. 1. Left diagram: placement of the tympana and spiracles. The circles represent the sound inputs and the tubes represent connecting tracheal tubes. The tracheal tube between the ipsilateral spiracle (IS) and the contralateral spiracle (CS) connects both sides; the tympana (IT and CT) are located on the forelegs and in connection with the other sound openings by the tracheal tubes. Right diagram: block diagram of the electrical circuit used for amplifying and phase delaying the sound inputs.

sound inputs. This study enhances the previous implementation by using a new auditory mechanism with four sound inputs, closely based on cricket morphology.

An essential aspect of the cricket's behavior that must be reproduced by the robot is the preference for a specific calling song. Preference is mainly determined by recognition of songs with correct temporal features. The carrier frequency of the song is also important since frequencies of around 5kHz are required to obtain a directional signal. Carrier frequency 'preference' may be implicit in the frequency dependence of directionality in the peripheral auditory system [3]. But the preference for temporal features is a property of the subsequent neural processing. Using the new auditory device, this paper tests the behavior of a robot controlled by a neural model first presented in [4]. The aims are: to validate the new four-input implementation; to compare the behavior of the two-input and four-input systems; and to determine whether the new neural model accounts for the same range of cricket behavior as a previous model [5].

2 Methods

Robot Model. The robot model consists of (i) a standard Khepera robot with mounted auditory input device and (ii) the neural networks running on an attached workstation. Two auditory input devices are used in this report: a two-input device (described and tested in [3]) and a new enhanced four-input device. The latter is a based more closely on the cricket morphology, with four microphones mimicking the four sound entrances which contribute to the directional hearing of the cricket. These are positioned to preserve the spatial relationships between the real sound entrances (Figure 1, left). The analog input from the microphones is amplified and phase delayed in order to get a directional signal, as illustrated in Figure 1 (right). Settings for the internal gains and delays are taken from laser vibrometry measurements of these properties in real crickets [2]. The exact settings for the placement of the microphones, the amplifications and delays can be found in [6]. The analog signal produced by the electrical circuit is then converted to Poisson spike trains which are fed into the neural network.

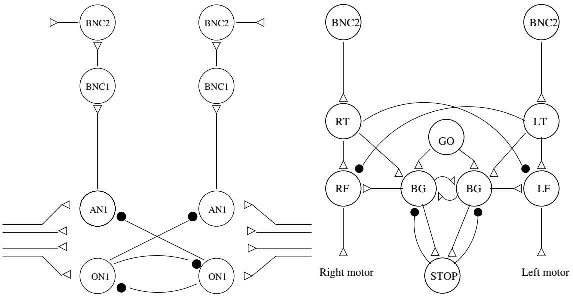


Fig. 2. Neural controller. Signals come from the tympana into the auditory network (left) and go on to the motor network (right) which passes signals out to the motors.

The neural networks used in this study are shown in Figure 2; more details of the model can be found in [4]. The difference with the previous model [5] lies in the enhancement of the auditory model and the extension of motor network.

Experimental Setup. The experiments are conducted according to the Kramer treadmill paradigm [7] with the subject at a fixed distance from the sound source; sensory feedback is obtained by rotating in the sound field. The stimuli consist of a computer generated calling song with configurable temporal features and carrier frequency of 4.7kHz. Analysis of the robot behavior is mostly done by examination of the neuronal activity recorded at every time step. A quantitative measurement is needed to compare our results with previous experiments and biological evidence. The length vector and its direction are adopted from [5] and defined as $L = \sqrt{(\sum \cos\theta)^2 + (\sum \sin\theta)^2}/N$, and, $D = \arctan(\sum \sin\theta / \sum \cos\theta)$. θ is defined as the cumulative sum of the instantaneous angles of the robot which are directly derived from the wheel encoders on the robot: $\theta = \frac{(v_l - v_r)\Delta t}{r}$.

3 Results

Taxis to Cricket Song. A basic test of the complete phonotactic behavior is whether the robot reliably tracks toward a cricket song. The tracking experiment is set up to match the criteria used for cricket in [7]: the robot should turn toward the sound and oscillate around the direction of the sound, and whenever the sound changes direction the robot should follow. Two speakers are placed on the azimuth with a separation of 90° at a distance of 20cm from the robot. The song is first played from one side for 10s, switched to the other side for 10s and finally switched back for 5s. Figure 3 (left) illustrates the results. The top diagram illustrates a run with the two-input implementation, the bottom diagram illustrates a run obtained by the new four-input implementation. Both robot models meet this criteria for phonotaxis.

Selectivity for Different Syllable Rates. Cricket phonotaxis is selective toward certain calling song patterns, and the syllable repetition interval (SRI) is regarded as the most important temporal feature, e.g. *Gryllus bimaculatus* show reduced phonotaxis as calling songs differ from their optimal SRI of 42ms. Following the biological tests of [7], the robot was tested with SRI 10-98 in steps of 8ms. The *phonotactic response* [5] measure defined for crickets in is used for the analysis of the results: $PR = L{\cdot}turn\%{\cdot}cos\,(\alpha - \eta)$. L is the length vector (see methods), $turn\%$ is the percent of time the robot spends turning, and $(\alpha - \eta)$ is the difference between the heading of the robot and the sound direction.

Figure 3 (right) shows results obtained by both the two-input and four-input implementation. Reduced phonotaxis is observed for SRI smaller than $26ms$, and the response tends to decrease as the SRI increases beyond $74ms$. The selectivity is slightly more pronounced in the four-input implementation, but is still rather less selective than the cricket. It is also less selective than was found for the previous neural model [5], where the robot only responded for SRI 26-66.

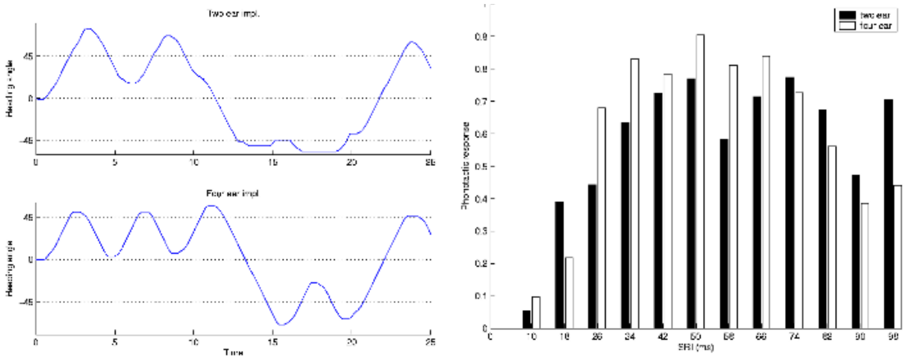


Fig. 3. Left: Heading angle of a robot implementation performing phonotaxis; top left diagram is a reference run taken obtained by the two-input implementation; the bottom left diagram is obtained by the four-input implementation. Right: Phonotactic response obtained by both robot models.

As well as comparing robot and cricket behavior, we can compare neural activity, for example the response of the simulated brain neurons to data recorded in the cricket by [8]. Figure 4 illustrates the neural activity recorded in the BNC2 neuron in the two-input and four-input implementation as the SRI is varied from 10-98ms. Both models have a firing-rate pattern that resembles the behavioral results obtained by [7], and the shape and magnitude of the firing-rate in the four-input model is directly comparable to the firing-rate found in the BNC2a neuron of the cricket [8]. However it appears this selectivity at the neural level has, in the robot, been lost in the translation into a motor response, due to the tuning of the RT and LT neurons in the motor network. As illustrated in Figure 5, these neurons will fire once per chirp in response to *any* activity in the BNC2 neurons, and this results in a turn lasting approximately the length of a chirp. For longer SRIs, the BNC2 response is reduced but not absent (see Figure 4) so the RT or LT neuron still fires, and the motor response still occurs.

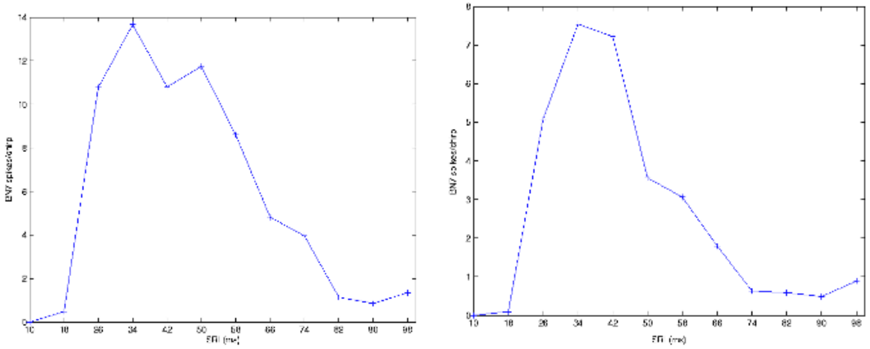


Fig. 4. The firing-rate measured in the BNC2 neuron when calling songs with different SRI are presented. The left diagram illustrates the results obtained by the two-input implementation, the right diagram illustrates the results from the four-input implementation.

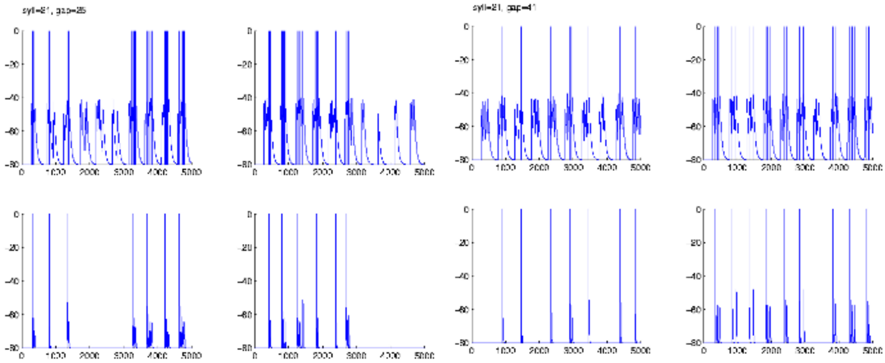


Fig. 5. Source of discrepancy between neural and behavioral selectivity. The top diagrams show the response of the BNC2, left and right respectively. The lower diagrams show the response of the LT and RT, neurons. For the four right diagrams the calling song was composed of 21ms syllables alternating with a gap length of 25ms; for the left side diagrams the calling songs was composed with 21ms syllables and a gap length of 41ms. The BNC2 neuron fires more to faster songs but the LT + RT response remains the same.

Previous experiments with this neural circuit [4] did not test the selectivity of the robot behavior so this limitation of the model was not apparent.

Songs from Above. This experiment uses an omnidirectional stimulus to replicate cricket experiments that are intended to show whether the localization and recognition of sound are independent processes. If a song is played from directly above the cricket, no consistent directional information is available. The fact

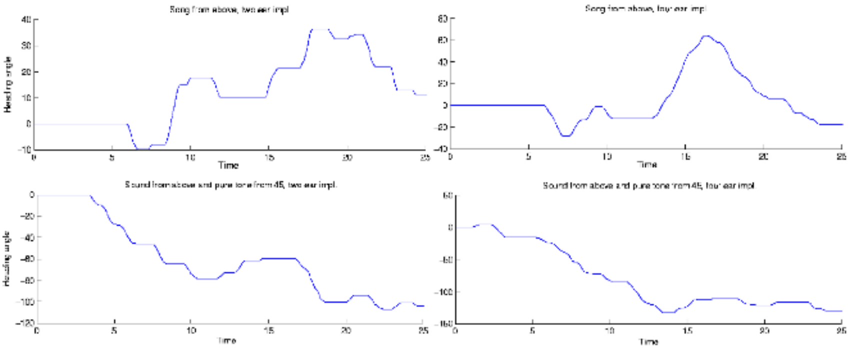


Fig. 6. Robotic behavior in the overhead sound experiment.Top: The sound is started after five seconds and both robots detect the undirectional signal. Bottom: both robots turn away from the added pure tone after a short delay. Note that the y-axis scale differs in the diagrams.

that crickets respond as though attempting to track the song has been presented as evidence that an independent recognition mechanism has 'switched on' the phonotaxis response, although the sound cannot be localized [9]. On the other hand, the addition of an unpatterned sound from one side in this paradigm causes the cricket to turn *away* from the side where the combined sound is loudest, and toward the side where the song pattern is clearer. This suggests a serial process in which recognition of the signal on each side feeds into a comparator to localize the sound. However, our previous robot model could replicate the cricket behavior described above [5], despite the absence of any independent recognition mechanism or explicit comparison. Figure 6 illustrates the results obtained for the new neural model, using both the two-input and four-input systems. In the top graphs, when sound is presented from above, the robot starts turning as though performing an undirected phonotaxis. In the bottom graphs, when a continuous tone is presented at 45° in addition to the overhead sound, both the implementations turn away from the direction of the pure tone to start tracking the opposite direction. This replication of the cricket behavior suggests that the overhead sound paradigm cannot be considered a definitive test for the independence or otherwise of recognition and localization processes in the cricket.

Split-Song. In [5] it was found that the previous neural model failed to replicate cricket behavior when presented with the split-song stimuli used in [10] to investigate the interactions between the localization and recognition tasks. In this experiment two speakers are placed on the azimuth with a separation of 135°. An unattractive song with a SRI 84 (composed with a syllable length of 21ms and a gap length of 63ms) is played simultaneously from both speakers, with one delayed by 42ms. In a region around the bisector of the speakers, the two songs are superimposed and form one attractive song with SRI 42. Crickets are reported to track the direction of the bisector [10]. Figure 7 illustrates the results obtained by the two-input and four-input implementation. The four-input

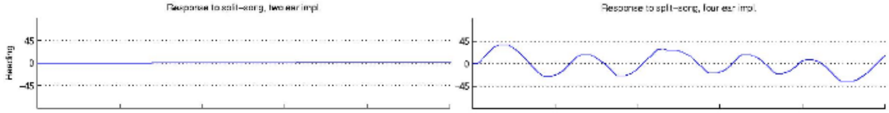


Fig. 7. Robot response to a split-song with a correct composite song. The left and right diagram illustrate the result obtained by the two-input and four-input implementation, respectively. The horizontal axis denotes the time.

implementation is clearly tracking the direction of the bisector while the two-input implementation is not responding at all to the split-song stimulus. The main reason for this difference, which can be confirmed by examining the model neuron responses, is that the two input implementation of the auditory system is more strongly directional than the four input implementation. The latter results in some ‘cross-talk’ between the two sides, sufficient for the combined SRI 42ms pattern of syllables to be intermittently represented in the auditory interneurons on one side or the other, and hence recognized and responded to by the robot. The two input implementation only encodes the respective SRI 84ms signal on each side, producing a weak recognition response on both sides, which cancels out resulting in no motor response.

4 Discussion

This study is one of a series of implementations of robot models of cricket behavior. In each iteration, we have attempted to improve the biological accuracy of the model. The robot described here combines a new version of the peripheral auditory processing (closely based on cricket morphology) with our most recent model of the neural processing (incorporating more of the known neurophysiological data) and is tested on a range of tasks derived from cricket experiments. A particular aim was to determine whether the full range of behavioral capabilities demonstrated with an earlier, and simpler, neural model [5] could also be demonstrated with the new, biologically more plausible, neural model. The results of all the tests performed in this validation process and a detailed analysis can be found in [6]. The selection of tests reported here illustrate a broad similarity of capability, with a few interesting differences. The robot could track cricket song, could show a selective response to different song patterns, and responded in the same way as the cricket to complex stimulus paradigms such as song from above and split song.

For most of the tests there was not a significant difference between the implementations with two inputs and four inputs. The exceptions to this are revealing: the two sound input produced stronger lateralization of the sound than a cricket but as a consequence failed to respond in the same way as a cricket to a split song stimulus. This suggests that the ‘tuning’ of the peripheral auditory system in the cricket may involve a tradeoff between strong lateralization (to better locate the sound) and weaker lateralization (to improve the chances of sound detection and recognition).

The robot behavior did not match the cricket as well as expected for its selectivity to different SRIs. This failure could be attributed to the way in which the motor circuit responded to the output of the auditory circuit. This motor circuit design was highly speculative, as there is almost no neurophysiological data on the output side for cricket phonotaxis. It was designed to account for the descriptions of cricket motor responses given in [11,12]. However, very recent experiments on the cricket (Hedwig and Poulet, personal communication), analyzing phonotaxis behavior at a high time resolution, suggest a very different mechanism for motor control. It now appears likely that there is a direct connection from the AN1 or ON1 interneurons to the motor interneurons responsible for producing turns, such that every syllable encoded in the auditory interneuron response causes a small deviation in the path. This 'fast' pathway is then modulated on a much slower time scale (i.e. over several chirps) by the output of the 'recognition' circuit, represented in our model by BNC2. We are currently testing a new neural model based on these results.

Acknowledgments. The authors thank the Dutch Ministry of Economic Affairs, grant nr: BSIK03024 (ICIS project) and the UK Biotechnology and Biological Sciences Research Council for funding the research in this paper.

References

1. A. Michelsen. The tuned cricket. *News Physiol. Sci.*, 13, 1998.
2. A. Michelsen, A.V. Popov, and B. Lewis. Physics of directional hearing in the cricket *Gryllus bimaculatus. J. Comp. Physiol. A.*, 175:153–164, 1994.
3. H.H. Lund, B. Webb, and J. Hallam. A robot attracted to the cricket species *Gryllus bimaculatus.* In *Fourth European Conference on Artificial Life*, 1997.
4. R. Reeve and B. Webb. New neural circuits for robot phonotaxis. *Phil. Trans. R. Soc.*, 361:2245–2266, 2003.
5. B. Webb and T. Scutt. A simple latency-dependent spiking neuron model of cricket phonotaxis. *Biol. Cybern.*, 82:247–269, 2000.
6. B. Torben-Nielsen. New ears for a robot cricket. Master's thesis, University Maastricht, 2004.
7. J. Thorson, T. Weber, and F Huber. Auditory behaviour of the cricket ii. simplicity of calling-song recognition in gryllus and anomalous phonotaxis at abnormal carrier frequencies. *J. Comp. Physiol. A*, 146:361–378, 1982.
8. K. Schildberger. Temporal selectivity of identified interneurons in the cricket brain. *J. Comp. Physiol. A*, 155:171–185, 1984.
9. J. Stabel, G. Wendler, and Scharstein H. Cricket phonotaxis: localization depends on recognition of the calling pattern. *J. Comp. Physiol. A*, 165:165–177, 1989.
10. T. Weber, J. Thorson, and F. Huber. Cricket phonotaxis: Females with forelegs fixed off the ground can track male songs. *Naturwissenschaften*, 75:317–319, 1988.
11. J. Rheinlaender and G. Blatgen. The precision of auditory lateralization in the cricket, *Gryllus bimaculatus. Physiological Entomology*, 7:209–218, 1982.
12. B.P. Oldfield. Accuracy of orientation in female crickets *Teleogryllus oceanicus* (gryllidae): dependence on song spectrum. *J. Comp. Physiol.*, 41:93–99, 1980.

Reinforcement Learning in MirrorBot

Cornelius Weber, David Muse, Mark Elshaw, and Stefan Wermter

Hybrid Intelligent Systems, SCAT, University of Sunderland, UK
{cornelius.weber, david.muse, mark.elshaw,
stefan.wermter}@sunderland.ac.uk
http://www.his.sunderland.ac.uk

Abstract. For this special session of EU projects in the area of NeuroIT, we will review the progress of the MirrorBot project with special emphasis on its relation to reinforcement learning and future perspectives. Models inspired by mirror neurons in the cortex, while enabling a system to understand its actions, also help in the solving of the curse of dimensionality problem of reinforcement learning. Reinforcement learning, which is primarily linked to the basal ganglia, is a powerful method to teach an agent such as a robot a goal-directed action strategy. Its limitation is mainly that the perceived situation has to be mapped to a state space, which grows exponentially with input dimensionality. Cortex-inspired computation can alleviate this problem by pre-processing sensory information and supplying motor primitives that can act as modules for a superordinate reinforcement learning scheme.

1 Introduction

Brain-inspired computation has the prospect of unprecedented control of artificial agents in addition to giving insights into neural processing. In the MirrorBot project, cortical mirror neurons which link perception and action have been chosen as a topic of study and a source of inspiration for building artificial systems. Mirror neurons which have been found in the motor cortex of the monkey are not only active when a monkey performs an action, but also when it observes the corresponding action being performed by somebody else (e.g. [1]). Thus, they have sensory neuron properties. This justifies that we generalise models of the sensory cortex, in particular from vision, to the motor cortex. Such models can learn, instead of a representation of a visual image, a representation of a sensory-motor mapping that has been given as input during learning and that may originate from a reinforcement learner. In reinforcement learning, the input state space grows exponentially if actions are extended. Here a motor cortex module can become a replacement. The hierarchical structure of the cortex furthermore suggests a capability of action organisation, and if given motivational input such as the reward (or Q-value) used for reinforcement learning [2] then the cortex might represent and act in response to such values. A view emerges that a reward-driven reinforcement module is surrounded by a cortex that not only pre-processes its input but also learns to understand, duplicate and anticipate it. We use a simple, but expandable robot docking manoeuvre as an example of a real world demonstration.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 305–310, 2005.

2 A Visually Guided Robotic Docking Task

Grasping of an object is a fundamental task for monkeys and humans. The robot equivalent is the docking, where it has to approach a table in a fashion that it can grasp an object lying on it. Figure 1 shows the geometry. A video can be seen at: `http://www.his.sunderland.ac.uk/robotimages/Cap0001.mpg`.

We have managed to control the robot performing this task based almost entirely on neural networks. Figure 2 shows the model. This consists of several modules which have partially been trained independent of each other.

First, the lower visual system consists of the mapping from the raw pixel image $\boldsymbol{I}$ to the hidden feature representation $\boldsymbol{u}$. This is trained unsupervised according to a generative model. Accordingly, the image has to be generated from the hidden code via feedback weights which are used only during training.

Second, the location associator weights from $\boldsymbol{u}$ to the area representing the perceived location $\boldsymbol{p}$ are trained supervised with the target object position given during training. After training the location can be filled in if it is missing, thus the attractor network does pattern-completion to localise the object.

Third, an action strategy is learnt by reinforcement using an actor-critic paradigm. Its input is the state $\boldsymbol{f}$ which is constructed as the outer product of $\boldsymbol{p}$ and a vector representing the robot angle $\boldsymbol{\varphi}$ w.r.t. the table. The critic value c represents the goodness of the current robotic state which is rewarded if the target object is at a graspable position, i.e. perceived near and in middle of view, and while $\varphi = 0$. During learning, the motor actions acquire a strategy to reach this goal [3]. In the following we will show that an alternative module can copy this action strategy to replace the reinforcement learner after task acquisition.

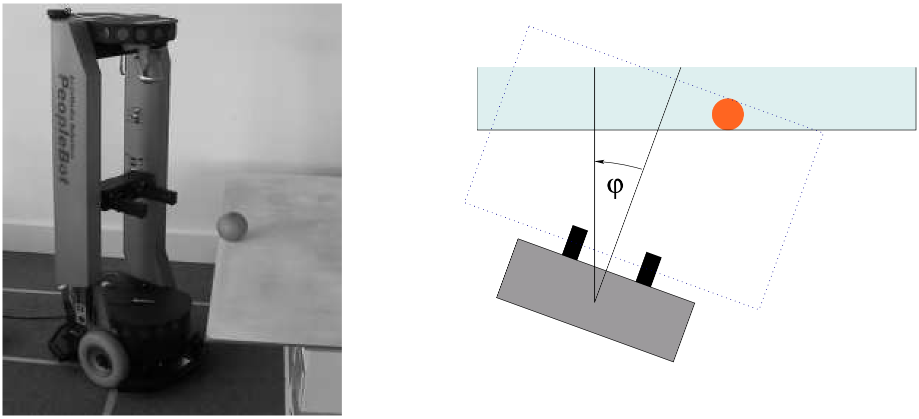


Fig. 1. Left, the PeopleBot robot aiming to grasp an orange fruit from a narrow table. Its camera is below the top-plate and is assumed fixed throughout learning and performance. Right, the geometry of the scenario with the table and target, above, and the robot with its grippers, below. The robot's input is the visual field (outlined by a dotted rectangle) and its angle φ to the table, obtained from internal odometry.

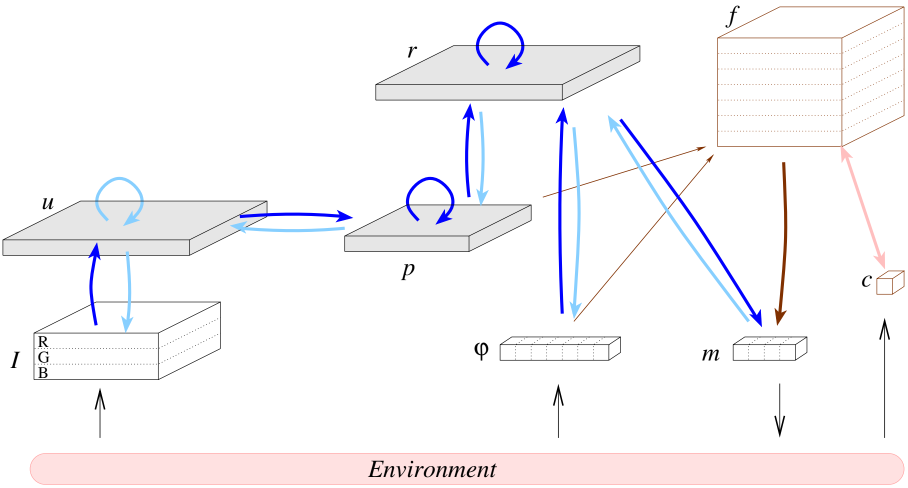


Fig. 2. The neural network which performs visually guided docking. The information flow is as follows: An RGB colour image $\boldsymbol{I}$ from the robot camera is given as input. A representation $\boldsymbol{u}$ is obtained on what we would identify as a V1 visual area, and from this we obtain the perceived location $\boldsymbol{p}$ of the target object within the image. This location and the rotation angle $\boldsymbol{\varphi}$ of the robot, together contain the information which is collated as state space vector $\boldsymbol{f}$. This is evaluated during learning by the critic c, and the motor actions $\boldsymbol{m}$ drive the robot's wheels. After task acquisition, the motor cortex representation $\boldsymbol{r}$ binds sensory-motor associations, and can produce actions $\boldsymbol{m}$ based on inputs $\boldsymbol{p}$ and $\boldsymbol{\varphi}$ by itself. This makes the state space available for learning other tasks. Thick arrows represent trained weights, the lighter of which are used only during learning. Shaded areas are supposed to belong to the cortex.

3 Motor Cortical Neurons Performing Docking

A further fourth module is a cortical representation $\boldsymbol{r}$ which associates this area's inputs $\boldsymbol{p}$, $\boldsymbol{\varphi}$ and $\boldsymbol{m}$. The combined input allows it to perform the stimulus-response mapping already performed via the state space. The intra-area attractor network connections are trained for prediction, allowing the network in addition to perform mental simulation. The idea is that automatic performance of the motor primitive by the motor cortex module makes the state space redundant and thereby makes it available to learn other tasks. A video showing the robot controlled by the simulated motor cortex can be seen at: `http://www.his.sunderland.ac.uk/supplements/NN04/MOV01065.MPG`

We propose to identify the model's state space with the basal ganglia, as these have been linked with reinforcement learning. There is biological evidence that the basal ganglia are active only during early phases of task acquisition [4].

4 Mirror Neurons for Multiple Actions

One advantage of the cortical modular motor action over reinforcement-trained agents is that cortical representations can easily be structured hierarchically, allowing multiple and composed actions to be represented. We produced three simulated robotic actions, "pick" which corresponds to the docking, "lift" during which (after picking an object) the robot would move back and turn and "go" during which the robot avoids objects and wanders around. We designed the environment and the robotic perception so that all behaviours act based on the same sensory input. The robot rotation angle $\boldsymbol{\varphi}$ and the distance to the wall and object $\boldsymbol{p}$ are contained in a "high-level vision" sensory input array.

Figure 3 shows the network architecture which performs these tasks which we have implemented on a simulator [5]. A top level area is added containing a vector $\boldsymbol{s}$, implemented as a SOM [6], which binds language input $\boldsymbol{l}$ together with a representation $\boldsymbol{r}$ that performs previously acquired sensory-motor bindings. Given language as input and thereby influencing the winner among $\boldsymbol{s}$ will influence the sensory-motor mapping. Vice versa, if complete sensory-motor stimuli are given, then the winner among $\boldsymbol{s}$ identifies the action which is being performed. Production and recognition share the same neural substrate as is the case for mirror neurons [1]. Also, action words are topographically arranged [7].

5 Extension of Docking via a Long-Range Strategy

The visually guided docking described in Section 2 requires that the robot is very close to the table and that the target object is visible. We are currently implementing a long-range table approach by reinforcement learning. While the robot's camera cannot find and identify the target object from a large distance, the table can be identified. This is particularly easy by an additional omni-directional camera fitted on top of the PeopleBot robot. Several design implementations are considered to integrate long-range and short-range docking.

(i) A straightforward approach is a monolithic state space spanning long- and short-range sensory input. In this case, however, the state space would become too large. In order to overcome these problems, it has been proposed to use adaptive state recruitment schemes [8][9].

(ii) A second approach is to train long-range and short-range behaviours separately using separate modules and to combine them sequentially. While this specific partitioning scheme might sound arbitrary, the switch between the use of the omni-directional camera for the long-range and the standard robot camera for the short-range clearly marks a boundary for the behaviour change. In humans, the switch might not be determined by the use of different sensors, but of actuators instead, such as the use of legs for long-range approach and of arms for the short range. Having defined behaviour modules, or *partial policies* which accomplish subtasks, it has been proposed to hierarchically implement a superordinate reinforcement scheme that acquires a policy for optimally switching between the subtasks as if they were primitive actions [10][11], see also [12]. This scheme, however, seems too powerful if there are just two modules.

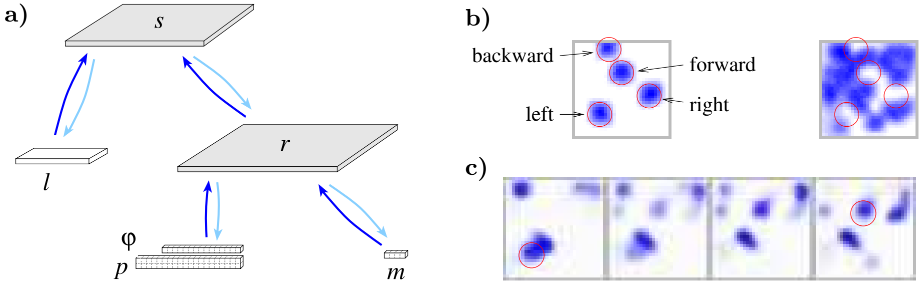


Fig. 3. a) The model architecture for multiple actions. The sensory-motor area representing $\boldsymbol{r}$ and binding sensory inputs $(\boldsymbol{p}, \boldsymbol{\varphi})$ with motor actions $\boldsymbol{m}$ stems from the model in Figure 2. New is the top-level SOM area where $\boldsymbol{s}$ associates language input $\boldsymbol{l}$ with a certain motor program $\boldsymbol{r}$. **b)** Left, the sensory-motor area and its innervation from the four motor area units. Each motor unit projects only onto a small (circled) region on this area. Right, the sensory-motor area and its combined innervation from the sensory inputs. One can see that the four areas which receive motor input (circled) are avoided by sensory input. **c)** Receptive fields of four SOM area units in the sensory-motor area. It can be seen that each SOM unit receives innervation from patchy regions in the sensory-motor area. The leftmost unit contains a sub-region (circled) that also receives input from the "left" motor unit, while the rightmost unit has a coinciding region (circled) with the "forward" motor unit. Thus the SOM area units perform dynamical feature binding, associating slightly different sensory input with different motor actions. The four units shown are all active during the "go" action; SOM units corresponding to different actions perform different bindings.

(iii) Contributing to such a hierarchical implementation, a behaviour module, or *action sequence* may be represented on the motor cortex, as we have proposed in Section 3. In this case the motor cortical units coding for that action sequence would be addressable by the reinforcement module just as single motor units are in the canonical implementation. The state space then would not need to consider any input that is accounted for by the cortical units. Both, long-range and short-range docking could be implemented by such motor units.

6 Discussion

Mirror neurons may play a major role in a distributed language representation of actions [7] by their multimodality [1]. In the MirrorBot project, also a modular neural architecture was devised to parse and understand a sentence [13] which we will integrate with the model described here. Further improvements have been done on attractor network models for visually focussing objects [14].

We have seen in Section 2 that a visual cortex-inspired module can deliver an object representation as required as input to a state space. This requires a fixed camera so that a visually perceived position can readily be used in the robot's motor coordinates. We are currently developing a coordinate transformation attractor network which will allow the camera to be rotated while the robot is docking. It associates *(i)* the visually perceived object location and *(ii)* the camera pan-tilt angle with *(iii)* the body-centred position of the target object. This is another strategy to extend the action range and limiting the state space.

In Section 3 we have seen that a motor-cortex inspired module can obviate the reinforcement module, and Section 4 demonstrated the potential of cortical action organisation. Our cortical models are inspired by the theory of generative models which reconstruct training data. Therefore, a "teacher" module, such as the reinforcement trained module, is required for any new action that the cortex then performs habitually. If the cortex receives the motivational, reward value used for reinforcement learning as additional input, then it is able to specifically perform such state-action associations which lead to a high reward [2]. With its associative and predictive capabilities, the cortex might directly use incoming stimuli to predict motivations of the agent, and enable a teleological behaviour.

Acknowledgements. This work is part of the MirrorBot project supported by the EU in the FET-IST programme under grant IST-2001-35282.

References

1. Rizzolatti, G., Fogassi, L., Gallese, V.: Motor and cognitive functions of the ventral premotor cortex. Current Opinion in Neurobiology **12** (2002) 149–154
2. Touzet, C.: Neural reinforcement learning for behaviour synthesis. Robotics and Autonomous Systems **22** (1997) 251–81
3. Weber, C., Wermter, S., Zochios, A.: Robot docking with neural vision and reinforcement. Knowledge-Based Systems **17** (2004) 165–72
4. Jog, M., Kubota, Y., Connolly, C., Hillegaart, V., Graybiel, A.: Building neural representations of habits. Science **286** (1999) 1745–9
5. Elshaw, M., Weber, C., Zochios, A., Wermter, S.: A mirror neuron inspired hierarchical network for action selection. In: Proc. NeuroBotics. (2004) 89–97
6. Kohonen, T.: Self-Organizing Maps. Springer (2001)
7. Pulvermüller, F.: The Neuroscience of Language. On Brain Circuits of Words and Serial Order. Cambridge University Press (2003)
8. Kondo, T., Ito, K.: A reinforcement learning with evolutionary state recruitment strategy for autonomous mobile robots control. Robotics and Autonomous Systems **46** (2004) 111–24
9. Lee, I., Lau, Y.: Adaptive state space partitioning for reinforcement learning. Engineering Applications of Artificial Intelligence **17** (2004) 577–88
10. Barto, A., Mahadevan, S.: Recent advances in hierarchical reinforcement learning. Discrete Event Dynamic Systems **13** (2003) 41–77
11. Kalmár, Z., Szepesvári, C., Lörincz, A.: Module-based reinforcement learning: Experiments with a real robot. Machine Learning **31** (1998) 55–85
12. Humphrys, M.: W-learning: A simple RL-based society of mind. In: 3rd European Conference on Artificial Life. (1995) p.30
13. Knoblauch, A., Markert, H., Palm, G.: An associative model of cortical language and action processing. In: Proc. 9th Neural Comp. and Psych. Workshop. (2004)
14. Vitay, J., Rougier, N., Alexandre, F.: A distributed model of visual spatial attention. In: Biomimetic Neural Learning for Intelligent Robotics. Springer (2005)

Varying the Population Size of Artificial Foraging Swarms on Time Varying Landscapes

Carlos Fernandes[1,3], Vitorino Ramos[2], and Agostinho C. Rosa[1]

[1] LaSEEB-ISR-IST, Technical Univ. of Lisbon (IST),
Av. Rovisco Pais, 1, TN 6.21, 1049-001, Lisbon, Portugal
{cfernandes,acrosa}@laseeb.org
[2] CVRM-IST, Technical Univ. of Lisbon (IST),
Av. Rovisco Pais, 1, 1049-001, Lisbon, Portugal
vitorino.ramos@alfa.ist.utl.pt
[3] EST-IPS, Setúbal Polytechnic Institute (IPS),
R. Vale de Chaves - Estefanilha, 2810, Setúbal, Portugal

Abstract. In this paper we present a Swarm Search Algorithm with varying population of agents based on a previous model with fixed population which proved its effectiveness on several computation problems [6,7,8][1]. We will show that the variation of the population size provides the swarm with mechanisms that improves its self-adaptability and causes the emergence of a more robust self-organized behavior, resulting in a higher efficiency on searching peaks and valleys over dynamic search landscapes represented here by several three-dimensional mathematical functions that suddenly change over time.

1 Introduction

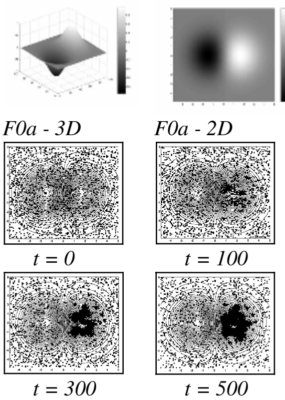
Swarm Intelligence (SI) is the property of a system whereby the collective behaviors of unsophisticated entities interacting locally with their environment cause coherent functional global patterns to emerge. SI provides a basis with which it is possible to explore collective (or distributed) problem solving without centralized control or the provision of a global model. The well-known bio-inspired computational paradigms ACO (*Ant Colony Optimization* [1]) and PSO (*Particle Swarm Optimization* [4]) are just two among many successful examples. To tackle the formation of a coherent social collective intelligence from individual behaviors we will address the adaptation of a social community to a cultural or informational dynamical landscape, represented here by several three-dimensional mathematical functions that change over time. Also, unlike past works [6,7,8], the size of our swarm population varies over time. We believe that *Swarms with Varying Population Size* (SVPS) provide a better model to mimic some natural features, improving not only the population ability to evolve self-organized foraging behavior as obtained in the past [6][1], while maintaining a self-regulated population adapted in real-time to different constraints in different search landscapes.

[1] Reference [6] is available at *http://alfa.ist.utl.pt/~cvrm/staff/vramos/Vramos-BMM.pdf*, as well as [7,8] and some related animated .AVI movies (check figure 6) at the main site.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 311 – 316, 2005.

2 The Swarm Landscape Foraging Model

The *Swarm with Fixed Population Size* (SFPS) model is described in [6] and it is based on the algorithm presented in [2,7,8]. The model simulates a population of ants evolving on 3D landscapes - (*x, y, z*) -, where *z* is the value of a mathematical function, quantized to fit on an [x,y] toroidal grid (discrete search space). The swarm, by means of pheromone deposition and evaporation (each ant tends to follow pheromone trails), evolves to higher/lower (maximization/minimization) regions of the landscape. A set of parameters ($\beta, \sigma, \eta\ k, p$) controls the swarm behavior – see [6] for details and [2] for a complete parameter analysis. Figure 1 shows the description of the model and the result of a run over function *F0a* [6]. First, the ants are randomly placed on the cells of the habitat. On each iteration, and for each ant, the probability P_{ik} (Eq. 1) to move to one of the surrounding cells is computed, according to the result of $W(\sigma)$ (Eq. 2). Once the ant is put on the new cell, the pheromone of that site is increased according to its height *z*. In the P(c) function, $\Delta_{max} = |\ z_{max} - z_{min}\ |$, being z_{max} the maximum altitude found by the colony so far on the function *habitat*, and z_{min} the lowest altitude. The other term $\Delta[c]$ is equivalent to (if our aim is to minimize any given landscape): $\Delta[c] = |\ z_c - z_{max}\ |$, being z_c the current altitude of one ant at cell *c*. If on the contrary, our aim is to maximize, then we should instead use $\Delta[c] = |\ z_c - z_{min}\ |$.



```
High-level description of the original Swarm Model
For all agents do place agent at randomly selected cell
End For
For $t$ = 1 to $t_{max}$ do  /* Main loop */
  For all agents do
    Compute $W(\sigma)$ and $P_{ik}$ /* According to Eq. 1 and 2 */
    Move to a selected neighboring cell not occupied by other agent
    Increase pheromone at cell $c$
        $P(c)= P(c)+[\eta+p(\Delta(c)/\Delta max)])$
  End For
  Evaporate pheromone by $K$, at all cells
End For

/* $\beta = 7; \sigma = 0.2; \eta = 0.07; k = 1.0; p = 1.93;$
 space [100, 100]; ants = 2000) */
```

Fig. 1. Model high-level description and maximization of *F0a* during 500 iterations [6]

$$P_{ik} = \frac{W(\sigma_i)w(\Delta_i)}{\sum_{j/k} W(\sigma_j)w(\Delta_j)} \qquad (1) \qquad\qquad W(\sigma) = \left(1 + \frac{\sigma}{1+\delta\sigma}\right)^{\beta} \qquad (2)$$

3 The Swarm Model with Varying Population Size

In this paper we propose and aim to analyze the behavior of a *Swarm with Varying Population Size* (SVPS). This characteristic is achieved by allowing ants to reproduce and die through their evolution in the landscapes. To be effective, the process of

```
For all ants do
    If ant meets ant do
        Compute $n$   /* number of occupied surrounding cells */
        Determine $P^*(n)$
        Compute reproduction probability  $Pr = P^*(n)\ [\Delta(c)/\Delta max]$ /*c is the cell of the main parent*/
        If random [0, 1] < $Pr$
            Create an ant and place it randomly on one of the free cells surrounding the main parent
    End if
End For
/* $P^*(0) = P^*(8) =0; P^*(4) = 1; P^*(5) = P^*(3) =0.75;  P^*(6) = P^*(2) =0.5; P^*(7) = P^*(1) = 0.25$ */
```

Fig. 2. High-level description of the reproduction procedure and *P*(n)* function

variation must incorporate some kind of pressure towards successful behavior, that is, ants that reach peaks/valleys must have some kind of reward, by staying alive for more generations – generating more offspring - or by simply having a higher probability of generating offspring in each time step. In addition, the population density in the area surrounding the parents must be taken into account during a reproduction event. When one ant is created (during initialization or by the reproduction process) a fixed energy value is assigned to it ($e[ant] = 1$). Every time step, the ant's energy is decreased by a constant amount of 0.1. The ant's probability of survival after a time step is proportional to its energy during that same iteration, which means that after ten generations, this and other ants will inevitably die ($e[ant] = 0$). Within these settings one ant that is, for instance, 7 iterations old, has a probability of 0.3 to survive through the current time step. Meanwhile, for the reproduction process, we assume the following heuristic: an ant (main parent) triggers a reproduction procedure if it finds at least another ant occupying one of its 8 surrounding cells (*Moore* neighborhood is adopted). The probability P_r of generating offspring – one child for each reproduction event – is determined as in fig. 2. Notice that when $n = 0$ or $n = 8$ no reproduction takes place and that higher/lower (if maximization/minimization) ants have more chance to reproduce ($\Delta(c)$ and Δmax as in section 2).

4 Results

From the set of functions used in [6] to test the fixed population size swarm, we chose *F0a* (fig. 1), and *Passino F1* (fig. 2) to compare the performance of the two models. The functions were adapted to a [100,100] toroidal grid. Parameters σ, η, k, p were set to 0.2, 0.1, 1.0 and 1.9 respectively, following previous tests conducted in [2] and [6]. Maintaining these values constant during all runs, we tested several configurations of SFPS and SVPS with β between 1 and 15 and initial population size (*IPS*) between 5 and 50 percent of the habitat size (10000 cells). By observing the results we concluded that SVPS clearly outperforms the fixed sized swarm when searching for peaks/valleys of the test functions, as well as the *Bacterial Foraging Optimization Algorithm*, BFOA, presented earlier by Passino [5], also compared in [6]. Comparing figure 1 and figure 2, we see that SVPS converges much faster to the desired regions of the habitat. We also notice that the variation of population mechanism' eliminates the wandering ants and drives the entire swarm towards peaks or valleys, simultaneously self-regulating the population when need it (e.g., peaks with a small area).

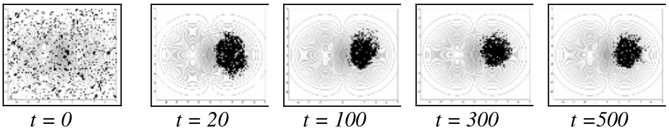


Fig. 3. Evolution of the SVPS over *F0a* during 500 iterations (maximization)

Such a clear difference between the performances of the two models may lead to the wrong idea that the population dynamics (*Reproduction*) is the sole responsible for the good performance of the SVPS and that pheromone fields (*Self-Organization*) are playing a back role. To investigate this possibility we tested SVPS with different values of *β*, from 1 (corresponding to a very low or absent tendency to follow pheromone) to 15. By analyzing the evolution of the median height of the swarm on the landscape we concluded that pheromone following is essential to a fast and self-organized convergence to peaks/valleys. Figure 4 shows the median height of our swarm over several time steps, for the maximization of *F0* and the minimization of *Passino F1*. In the first case the difference is not so obvious although a closer inspection reveals the poorer performance of *β=1* configuration. Meanwhile the *Passino F1*, with its multiple peaks and valleys, creates more problems to the colony and truly reveals the utility of the positive (*pheromone reinforcement*) and negative feedbacks (*evaporation*) introduced by the pheromone fields [6]. These results show that although pheromone following and varying population mechanisms can lead the swarms to the desired regions, the cooperation between the two processes result in a much powerful system.

By plotting the evolution of the population size of SVPS, we concluded that for each function and task (minimization/maximization) the population tends to evolve until it stabilizes around a specific value. In figure 5a) we can see this behavior when minimizing *Passino F1*, as *β* remains fixed and the initial population size varies from 10% to 30% of the habitat size. Also, the final values of the population size differ between functions and tasks: within *Passino F1* the population becomes stable around300 agents while the swarms minimizing *F0a* evolved populations with near 600 individuals; if we try to maximize *Passino F1* or evolve the swarm into other functions we see that the populations become stable around different values. That is, the mechanism here introduced is able to self-regulate the population according to the actual foraging environment (swarms adapt to it).

This pattern, however, is broken when the value of *β* decreases (representing a less self-organized behavior). As we can see in figure 5b), *β* equal to 1 leads the swarm to an unpredictable behavior. The limits to the population growth are imposed by the search space itself because only one ant may occupy each cell. Under these conditions, it is impossible to observe exponential growing of population size. On the other hand, mass extinction is possible. Almost all the extinctions observed in the exhaustive SVPS testing were related to models with *β*=1. These results emphasize the role of Self-Organization while evolution occur trough any requested aim (our task), enhancing the increasing recognition that *Natural Selection* (here coded via the reproduction process) and *Self-Organization* (here coded via the pheromone laying process) work hand in hand to form *Evolution*, as defended by Kauffmann [3].

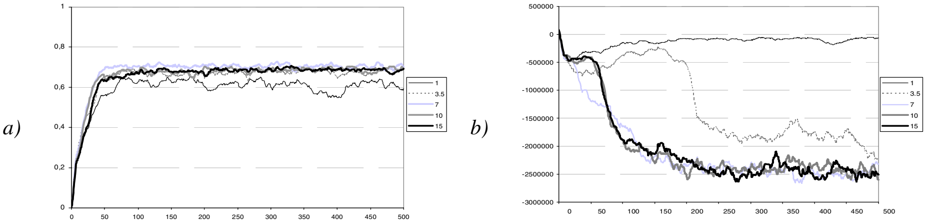


Fig. 4. Median height of the ants when maximizing *F0a* (a) and minimizing *Passino F1* (b). (500 iterations; β equal 1, 3.5, 7, 15; *IPS* = 20% ants for all configurations)

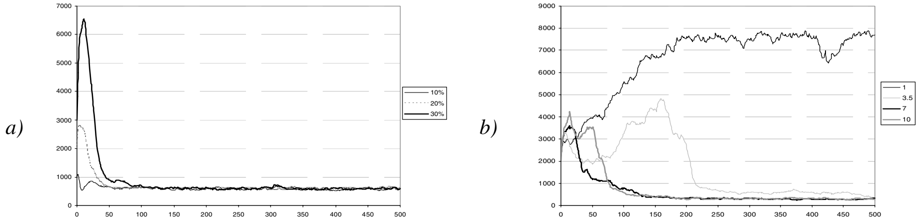
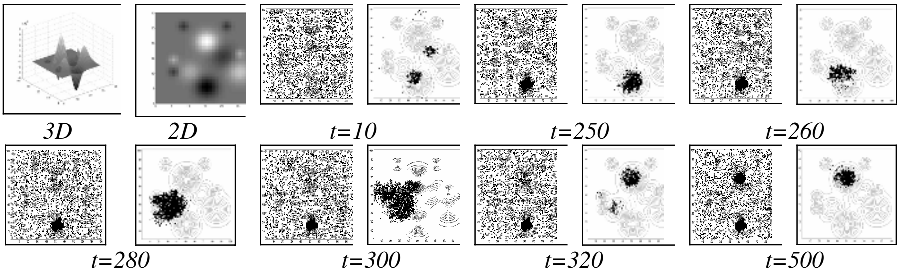


Fig. 5. Population growth (500 iterations) when minimizing *Passino F1*: a) *IPS* = (10%, 20%, 30%) and β = *7*; b) *IPS* = 20% and β = (1, 3.5, 7, 10)

One of the features of SFPS discussed in [6] was the ability to adapt to sudden changes in the roughness of the landscape. These changes were simulated by abruptly replacing one test function by another after the swarm reaches the desired regions of the landscape. Another way of simulating changes in the environment consists on changing the task from minimization to maximization (or vice-versa). The swarm performance was convincing and reinforced the idea that the system is highly adaptable and flexible. In here, we followed and conducted similar tests using SVPS and concluded that varying population size increases the capability of the swarm to react to changing landscapes. Figure 6 shows SFPS (6a) and SVPS (6b) trying to find the lower values of *Passino F1* until *t*=250, and then searching for the higher values.


3D *2D* *t=10* *t=250* *t=260*

t=280 *t=300* *t=320* *t=500*

Fig. 6. SFPS (left) and SVPS(right) evolving on *Passino F1*. From *t=0* to *t=250* the swarm is induced to search the valleys of the landscape. After *t=250* the task changes and the swarm must find the higher values of the function.

Notice, by observing the 2D and 3D representation of the function and the maps in figure 6, how SVPS uses the landscape specific characteristics to emerge a distinctive migration process (as a *flocking behavior*) after t=250: the swarm climbs the neighboring peak (local optima) on the left and from that point finally reaches the higher peak of the landscape, where it remains while self-regulates the population; by some complex mechanism the swarm does not climb the peak on the right side of the valley (which is higher) avoiding the valley that stands between that local optima and the desired region. We can see that SVPS not only evolves faster to valleys as it re-adapts better to the changing goal. In fact, at t=500, SFPS is still migrating to the higher peak in the landscape while SVPS has already performed the task.

5 Conclusions and Future Work

We showed that adding a mechanism of varying population size to a self-organized swarm search algorithm increases not only its capability to search/find peaks and valleys of fitness landscapes, but it also provides the system with some unexpected self-regulated behavior – for instance, population growth to predictable values even with rather different β and initial population size values. SVPS also proved to be more effective when evolving over changing dynamic landscapes. The connection between SVPS and co-evolutionary artificial life models is worth inspecting since species have to adapt to constant changes in the fitness landscape caused by mutations in other species. The ability revealed by SVPS to quickly readapt to changing environments suggests its utility in the study of co-evolutionary systems dynamics.

References

1. Bonabeau, E., Dorigo, M., Theraulaz, G., *Swarm Intelligence: From Natural to Artificial Systems*, Santa Fe Institute, Oxford Univ. Press, New York, Oxford, 1999.
2. Chialvo, D.R., Millonas, M.M., *How Swarms build Cognitive Maps*, In Steels, L. (Ed.): The Biology and Technology of Intelligent Autonomous Agents, 144, NATO ASI Series, 439-450, 1995.
3. Kauffmann, S.A., *The Origins of Order: Self-Organization and Selection in Evolution*, New York: Oxford University Press, 1993.
4. Kennedy, J. Eberhart, Russel C. and Shi, Y., *Swarm Intelligence*, Academic Press, Morgan Kaufmann Publ., San Diego, London, 2001.
5. Passino, K.M., *Biomimicry of Bacterial Foraging for Distributed Optimization and Control*, IEEE Control Systems Magazine, pp. 52-67, June 2002.
6. Ramos, V., Fernandes, C., Rosa, A., *Social Cognitive Maps, Swarm Collective Perception and Distributed Search on Dynamic Landscapes*, submitted to Brains, Minds & Media – Journal of New Media in Neural an Cognitive Science, NRW, Germany, 2005.
7. Ramos, V., Pina, P., Muge, F., *Self-Organized Data and Image Retrieval as a Consequence of Inter-Dynamic Synergistic Relationships in Artificial Ant Colonies*, Soft-Computing Systems – Design, Management and Applications, IOS Press, vol. 87, pp. 500-509, Netherlands 2002.
8. Ramos, V., Almeida F., *Artificial Ant Colonies in Digital Image Habitats – A Mass Behaviour Effect Study on Pattern Recognition*, , in Marco Dorigo, Marco Dorigo, Martin Middendorf & Thomas Stüzle (Eds.), ANTS'00, 2nd Int. Workshop on Ant Algorithms, pp. 113-116, Brussels, Belgium, 7-9 Sep. 2000.

Lamarckian Clonal Selection Algorithm with Application

Wuhong He, Haifeng Du, Licheng Jiao, and Jing Li

Institute of Intelligent Information Processing and
National Key Lab of Radar Signal Processing, Xidian University,
710071 Xi'an, China
hewuhong@163.com

Abstract. In this paper, Lamarckism and Immune Clonal Selection Theory are integrated to form a new algorithm, Lamarckian Clonal Selection Algorithm (LCSA). In the novel algorithm, the idea that Lamarckian evolution described how organism can evolve through learning, namely the point of “Gain and Convey” is applied, then this kind of learning mechanism is introduced into Standard Clonal Selection Algorithm (SCSA). In the experiments, ten benchmark functions are used to test the performance of LCSA, and the impact of parameters for LCSA is studied with great care. Compared with SCSA and the relevant evolutionary algorithm, LCSA is more robust and has better convergence.

1 Introduction

Antibody Clonal Selection Theory is a very important concept in the immunology. Clone means reproduction or asexual propagation. A group of genetically identical cells descended from a single common ancestor, such as a bacterial colony whose members arose from a single original cell as a result of binary fission.

Prior to Charles Darwin’s evolution theory of natural selection, Jean Baptiste Lamarck proposed that characteristics could be acquired and passed on to their offspring during an organism’s life [1], which means the experiences adapting to the environment can be inherited, and it has been known as Lamarckian evolutionary theory, namely Lamarckism. Although discredited in biology [2], Lamarckian evolution has been proven to be a effective concept for improving artificial intelligence algorithms [3]. In fact, Lamarckism has shown powerful performance in computing [4].

Learning mechanism is introduced into Standard Clonal Selection Algorithm (SCSA) [5], and Lamarckian Clonal Selection Algorithm (LCSA) is presented in this paper. Based on the idea “gain and convey”, LCSA makes full use of the experiences gained during the learning process to enhance the information communication among individuals, and improve the performance of algorithm. Relative function optimization experimental results shows that compared with SCSA and relative evolutionary algorithm, such as IGA [6], LCSA have better performance in global search and computing efficiency.

The rest of this paper is organized as follow. Section 2 describes the basic skeleton of SCSA as well as some problems in it. Section 3 illustrates LCSA and its

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 317 – 322, 2005.

implementation steps. Section 4 discusses the experimental results of LCSA, SCSA and relative evolutionary algorithm on 10 2-dimensional multimodal functions, and the impact of parameters on the performance of LCSA is analyzed. Finally, Section 5 concludes the paper with a short summary and gives a discussion for our future work.

2 Standard Clonal Selection Algorithm (SCSA)

Based on the antibody clonal selection in immunology, we have proposed the SCSA [5] in Fig.1.

Simple Clonal Selection Algorithm (SCSA)

Step 1 $k = 0$; Initialize antibody population $\boldsymbol{A}(0)$ at random and set the parameters of algorithm, then calculate the affinity of initialized population.

Step 2 According to the antibody affinity and the clone size, get a new antibody population $\boldsymbol{A}(k)$ by Clonal Selection Operator.

Step 3 $k = k + 1$; if the halted condition is satisfied, the iteration is terminated, otherwise, return to Step2.

Fig. 1. The skeleton of SCSA

By introducing learning mechanism into SCSA, evolution and learning are combined well and supplement each other, with the unification between the time and space and between the genotype and phenotype, thus evolving the population under the guide of learning and making algorithm find the global optima fast and efficiently.

3 The Skeleton of LCSA

According to the fitness distribution in population, several subpopulations are formed. Then the main steps of LCSA in this paper is shown in Fig. 2.

Lamarckian Clonal Selection Algorithm (LCSA)

Step 1 k=0; Initialize the population randomly, set the algorithm parameters;

Step 2 According to the fitness distribution in the population, divide the population into several parts;

Step 3 Through clonal selection operating, a new population is formed;

Step 4 Get Heroes' Experiences (HE) and Successful Civilians Experiences SCE, then Convey HE and SCE, the next generation is generation completely;

Step 5 $k = k + 1$; Check the halted condition of algorithm if the halted condition is satisfied, the iteration is terminated, otherwise, return to Step2.

Fig. 2. The main steps of LCSA

In general, the whole candidate population is divided into two parts: high-fitness subpopulation Heroes and low-fitness subpopulation Non-Heroes. Furthermore, Non-

Heroes can be divided into Employees evolving by conveying the Heroes' Experiences (HE) and Civilians evolving by themselves. After iterations, some individuals in Employees can go into Heroes, which are called Successful Employees (SE), while the others are named Unsuccessful Employees (USE). In the same naming way, Civilians can be divided into Successful Civilians (SC) and Unsuccessful Civilians (USC), and USC can get Successful Civilians Experiences (SCE) from SC.

Heroes with high fitness dominate the whole population and guide the evolution direction of the whole candidate population. During the process of learning, HE is the most important factor; on the other hand, SCE from a small population is a good helper of HE, which also should be noticed.

4 Validation of LCSA

4.1 Experimental Results

In order to prove the effect and efficiency of proposed algorithm, the similar type of evolutionary algorithm Immune Genetic Algorithm (IGA)[6] with prior knowledge is introduced. We regard the global hits(the percentage of the global maximum found) as the primary objective achieved by the algorithm and function evaluations as the second objective[7]. For 10 functions[8], the parameters of SCSA, LCSA are set as: population size n=50, the clone size n_c=5, coding length l=40, the maximum number of iteration 200, and the required precision 0.01. For comparisons, the parameters of IGA are fixed as: population size n=250, the others are set the same as two algorithms above. The statistical results of three algorithms out of 20 runs respectively are summarized in Table1. For f_3, f_4, f_5, f_6, the evolutionary process of three algorithms are shown in Fig.3. Where f_3 is Needle-in-a-haystack Function, it has uncountable local optima and only one global optimum.

Table 1. The results of IGA, SCSA and LCSA on f_1- f_{10},. "G.hit" is the global hits. "Eval" is the number of average evaluations out of 20 runs.

Prob	IGA		SCSA		LCSA	
	G. hit	Eval	G. hit	Eval	G. hit	Eval
f_1	16/20	12504	20/20	2315	**20/20**	**1605.8**
f_2	20/20	**1539.2**	20/20	1805	**20/20**	1700
f_3	20/20	11479	20/20	13385	**20/20**	**10915**
f_4	14/20	18756	19/20	12035	**20/20**	**8720.9**
f_5	18/20	7044.1	20/20	3335	**20/20**	**2797.8**
f_6	20/20	**15794**	19/20	24050	**20/20**	19927
f_7	18/20	11180	20/20	**8015**	**20/20**	10340
f_8	20/20	**2372.9**	20/20	6155	**20/20**	2822.9
f_9	14/20	18022	20/20	4955	**20/20**	**3952.3**
f_{10}	13/20	24624	20/20	10970	**20/20**	**5659.1**

And it is indicated from Table 1 that for all test functions, LCSA can find the global optima out 20 runs, namely the on the primary principle "G.hit" is 20/20, which shows its good robustness. For f_1, f_4, f_5, f_7, f_9 and f_{10}, IGA can't find the global solution. For f_4, f_6, SCSA can't achieve the global hits 20/20. These indicate LCSA outperforms SCSA and IGA with respect of global search. In terms of the second objective “Eval”, for functions f_2, f_6,f_7 and f_8 LCSA need a little more computer cost than SCSA and IGA, but for the others, the computer cost of LCSA outperforms is much less that two of them. Those are caused by the loss of diversity in the later evolutionary stage in LCSA. Finding better techniques to improve this weakness will be our future research.

4.2 The Analysis of Parameters

In LCSA, the main parameters are the population size n, the clone size n_c, the coding length l , the Civilians size c_n, Employees size e_n and mutation probability p_m. To test the impact of each parameter on the global convergence (Global hits) and the number of function evaluations (Eval) of our algorithm, 7 parameters are generally fixed as n=50, n_c=5, l=40, h_n=5, p_m=0.025, c_n and e_n vary randomly in range [2, n-h_n-2] and [2, n-h_n-2] respectively, and the sum of c_n and e_n is 45. Since c_n and e_n are random numbers, actually there are only 5 parameters here. In experiment, we let one of them changeable, and set 4 of them as above fixed values. Taking f_3 for example, the number of function evaluations (“Eval”) are acquired out of 20 runs for each value of parameter.

A) Influence of population size *n*

Sample n by the interval of 10 from 10 to 200. It is shown in Fig.3(a) that in case that n=10, “Eval” is the least, and with the increase of n, it increases and the converges speed slower, but relevant number of iteration becomes smaller. Those suggest that some candidate solutions that contribute less to generating the optimal solutions due to the increase of n.

B) Influence of clone size n_c

Sample n_c by the interval of 1 from 2 to 21 and the result shown in Fig.3(b) indicates ”Eval” is the least when n_c=4. While with n_c’s increase, the convergence speed is slower, and the relative number of iteration becomes smaller. It indicates that after n_c increases to some level, some candidate solutions contributing less to generating the optimal solutions, which is similar to the influence of n in term of this point.

C) Influence of coding length *l*

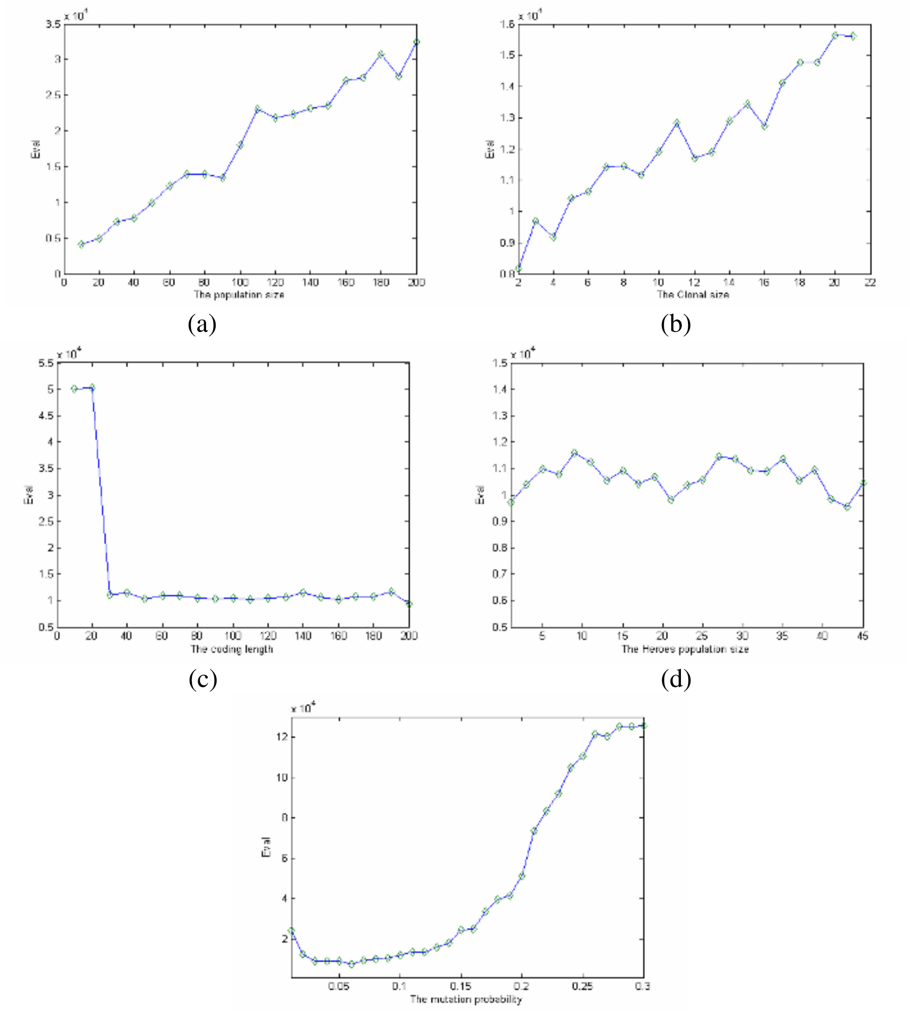
Sample l by interval of 10 from 10 to 100, from the result shown in Fig.3(c), l lying between 30 and 200 has a little influence on “Eval”. In addition, it should be note that with bigger coding length, the computer cost also gets more during decoding process.

D) Influence of Heroes size h_n

Sample h_n by interval of 2 from 1 to 45. It is implies from Fig.3(d) that h_n between 1 and 45 influence on “Eval” is not very obvious, and the values are about 10000, which is mainly caused by the random value of c_n and e_n, thus making the learning active and showing the advantage of learning to avoid the similarity between gaining and conveying of experiences.

E) Influence of mutation probability p_m

For the analysis, sample p_m by interval 0.01 from 0.01 and 0.3, but not set it as the reciprocal of coding length l in algorithm. It is suggested from Fig.3(e) that “Eval” doesn’t change a lot in the range of 0.01 and 0.1 which is a reasonable interval. But



(a) (b)

(c) (d)

(e)

Fig. 3. The main parameters population size n, clone size n_c, coding length l, Heroes size h_n and mutation probability p_m's impacts on the global search and numbers of function evaluations of LCSA, which are shown respectively in (a), (b), (c), (d) and (e). The horizontal axis is the tested parameter values, and the vertical axis is the average numbers of evaluation function out of 20 runs.

beyond this interval, bigger p_m destroys the current population solution, thus making the evolution stagnate and "Eval" bigger.

On the analysis of parameters, the parameters in LCSA only can influence the convergence speed, but a little the global convergence, which shows good robustness of algorithm. It should be noted that those parameters sets are not the best options, but we still get good global performances.

5 Conclusions

Based on immune clonal selection theory and Lamarckism, Lamarckian Clonal Selection Algorithm (LCSA) is presented in this paper,. The LCSA improves some of the drawbacks of the SCSA. The comparisons of the algorithms' performances are made considering the global hits and the function evaluations. The LCSA can provide more efficient performance with better reliability on most 2-D multimodal functions in test. Hence, using new techniques to make LCSA better will be included in our future work.

Acknowledgment

The work was sponsored by the National Natural Science Foundation of China (No: 60133010) grants and the Natural Science Foundation of Shaanxi (No: 2004F29).

References

1. Dawkins, R.: The Blind Watchmaker. Norton, 1996
2. Ross, B.J.: A Lamarckian Evolution Strategy for Genetic Algorithms. In: L. Chambers (ed.) Practical Handbook of Genetic Algorithm, Vol. 3, chapter 1, CRC Press (1999) 1-16
3. Mitchell, T.M.: Machine Learning, McGraw Hill, 1997
4. Hart, W.E., Belew, R.K.: Optimization with genetic algorithms hybrids that use local search. In: Belew, R.K., Mitchell, M. (eds.): Adaptive Individuals in Evolving Populations: Models and Algorithms, Vol. 26, chapter 27, SFI Studies in the Sciences of Complexity. (1996) 483--496
5. Du, H.F., Jiao, L.C., Wang, S.A.: Clonal Operator and Antibody Clone Algorithms. In: Shichao, Z., Qiang, Y., Chengqi, Z. (eds.): Proceedings of the First International Conference on Machine Learning and Cybernetics. IEEE, Beijing (2002) 506–510
6. Jiao, L.C., Wang L.: A Novel Genetic Algorithm based on Immunity. IEEE Trans. Systems, Man and Cybernetics, Part A. Vol.30, No.5 (2000) 552–561
7. Liang, K.H., Yao, X., Newton, C.: Evolutionary Search of Approximated N-Dimensional Landscapes. International Journal of Knowledge-Based Intelligent Engineering Systems. Vol.4, No.5 (2000) 172-183
8. Syrjakow, M., Szczerbicka, H.: Efficient parameter optimization based on combination of direct global and local search methods. In Davis, L.D., Jong, K.D., Vose, M.D., Whitley, D. (eds.): Evolutionary Algorithms, IMA program on mathematics in high-performance computing. Springer-Verlag, New York (1999) 227-249

Analysis for Characteristics of GA-Based Learning Method of Binary Neural Networks

Tatsuya Hirane, Tetsuya Toryu, Hidehiro Nakano, and Arata Miyauchi

Department of Computer Science and Media Engineering,
Musashi Institute of Technology, Tokyo, Japan

Abstract. In this paper, we analyze characteristics of GA-based learning method of Binary Neural Networks (BNN). First, we consider coding methods to a chromosome in a GA and discuss the necessary chromosome length for a learning of BNN. Then, we compare some selection methods in a GA. We show that the learning results can be obtained in the less number of generations by properly setting selection methods and parameters in a GA. We also show that the quality of the learning results can be almost the same as that of the conventional method. These results can be verified by numerical experiments.

1 Introduction

The binary neural network (ab. BNN) is a feedforward network in which input and output signals are binary. The BNN can realize a desired boolean function if a sufficient number of hidden neurons exist [1][2]. The BNN can be implemented easily by logic VLSI, and such an approach has been presented [3]. Application examples of BNN include pattern recognition [4], and error correcting codes [5]. As fundamental of these applications, many studies on the learning methods of BNN have been actively carried out. Ref.[2] presents an expand and truncate learning (ab. ETL) which is a simple and fast geometrical learning method. Ref.[6] presents the GAETL which is based on the ETL and uses a genetic algorithm (ab. GA) [7] to determine network parameters of BNN. The GAETL can reduce the number of neurons in the hidden layer and can suppress dispersion of parameters. However, detail analysis for the characteristics of GAETL has not been sufficient so far. Therefore, there have been some problems as follows.

1. Investigation of searching ability of a GA for BNN
2. Consideration of coding methods to a chromosome in a GA, selection methods and parameters in a GA
3. Calculation time of the GAETL

In this paper, we analyze learning characteristics of the GAETL for BNN. First, we consider coding methods for BNN, and discuss the necessary length of a chromosome in a GA for learning. We show that the chromosome length is enough if sufficient combination numbers for network parameters are expressible. We also show that a gray coding is more suitable than a binary coding for coding methods to a chromosome. Then, we compare some selection methods in a GA.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 323–329, 2005.

We show that for a complex function, the method including elitism and inverse-elitism [8] as selecting strategy can reduce the number of hidden neurons better than other selection methods. Finally, we show that the learning results can be obtained in the less number of generations by properly setting selection methods and parameters in a GA. We also show that the quality of the learning results can be almost the same as that of the conventional method. These results can be verified by numerical experiments.

2 Binary Neural Networks

In this paper, we deal with a three-layer BNN. Then the BNN is described by Eq.(1).

$$y = S\left(\sum_{i=1}^{M} w_i^o z_i - T^o\right), z_i = S\left(\sum_{j=1}^{N} w_{ij}^h x_j - T_i^h\right), S(X) = \begin{cases} 1 & \text{if } X \geq 0 \\ 0 & \text{if } X < 0 \end{cases} \quad (1)$$

where $x \equiv (x_1, \cdots, x_N)$ is an input, $z \equiv (z_1, \cdots, z_M)$ is a hidden layer output, y is an output and $x_j, z_i, y \in \{0, 1\}$. w_{ij}^h is a connection parameter between j-th input and i-th hidden neurons and w_i^o is a connection parameter between i-th hidden and output neurons. T_i^h and T^o are threshold parameters of i-th hidden and output neurons, respectively. All the parameters can be integer values by using the ETL or the GAETL introduced in the next section.

3 GA-Based Learning Method of Binary Neural Networks

In this section, we introduce basic concepts of the ETL [2] and the GAETL [6]. In the ETL, N-bit teacher signals are represented by the coordinates of an N-dimensional space. Each input vector corresponds to a vertex in the space. Let a true vertex (respectively, a false vertex) be the vertex whose desired output is one (respectively, zero). The ETL determines a set of the hyperplanes which linearly separate true and false vertices. Then, the number of the separating hyperplanes (ab. SHPs) corresponds to the number of hidden layer neurons.

The GAETL uses a GA in determining the SHPs in the ETL. In the GAETL, a connection parameter set $w_i^h \equiv \left(w_{i1}^h, \cdots, w_{iN}^h\right)$ is encoded to a binary string with length ηN, where η denotes the number of genes in the genotype of w_{ij}^h. This binary string is a chromosome in a GA. Fig.1 shows the flowchart of the GAETL in determining the SHPs. The overall algorithm of the GAETL is as follows. Step 1: Generate the initial population consisting of population size M_g with random initial values. Step 2: Calculate threshold parameters $T_i^h(p)$ for every chromosome, where p is the suffix of chromosome in the population. The threshold parameter $T_i^h(p)$ is determined such that each hyperplane corresponding to each chromosome can separate true vertices as much as possible. Step 3: Let $L_i(p)$ be the number of separated true vertices for the p-th chromosome, and

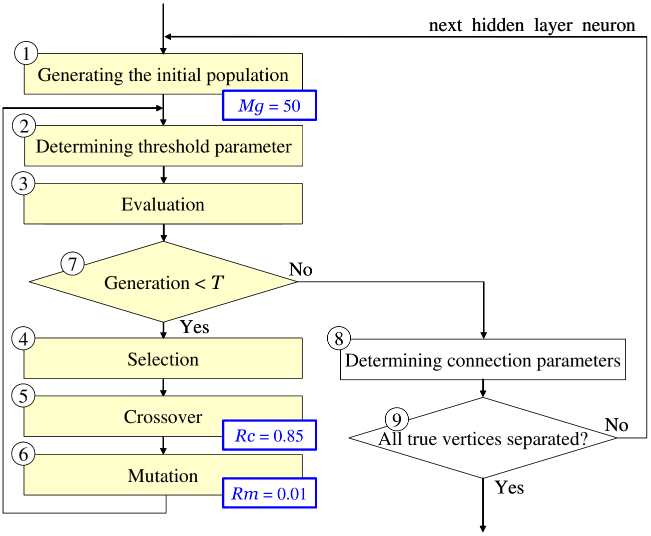
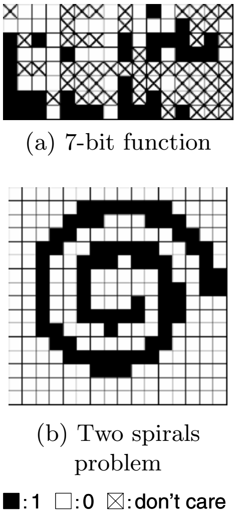

next hidden layer neuron
1
Generating the initial population
Mg = 50
2
Determining threshold parameter
3
Evaluation
7
Generation < T
No
Yes
4
Selection
5
Crossover
Rc = 0.85
6
Mutation
Rm = 0.01
8
Determining connection parameters
9
All true vertices separated?
No
Yes

Fig. 1. The flowchart of the GAETL in determining the SHPs


(a) 7-bit function

(b) Two spirals problem

Fig. 2. Examples of teacher signals

$L_i(p)$ is used as the fitness. Steps 4 to 6: Apply selection, crossover and mutation to the chromosome in the population as operations of a GA. Step 7: Repeat Steps 2 to 6 for constant generations T. Step 8: Determine the chromosome having the best fitness as network parameters of a single neuron in the hidden layer. Then, the separated true vertices are declared as "don't care". Step 9: If all true vertices can not be separated in the hidden layer neuron, then increment a neuron in the hidden layer and go to Step 1. The learning of the GAETL will continue until all true vertices are separated.

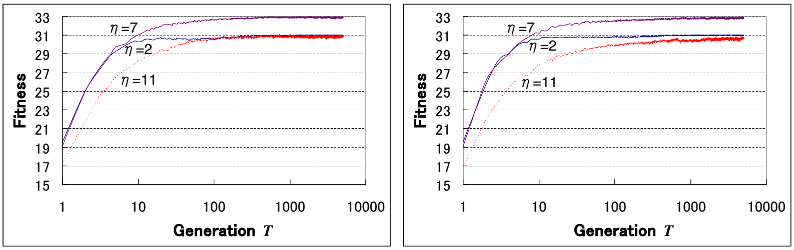
4 Numerical Experiments

In order to investigate searching ability of a GA for BNN, we perform numerical experiments. At first, we focus on generation-fitness characteristics in the case where the first neuron in the hidden layer is determined. Moreover, we compare the number of hidden layer neurons in learning results. We use "7-bit function [6]" in which $N = 7$ and "Two spirals problem [6]" in which $N = 8$ as examples of teacher signals. Fig.2 illustrates these functions.

We select the same parameters in a GA as the conventional method [6]; population size with $M_g = 50$, uniform crossover with rate $R_c = 0.85$ and normal mutation with rate $R_m = 0.01$.

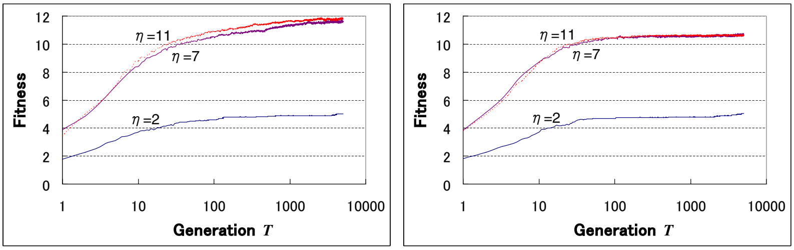
4.1 Investigation of the Necessary Length of Chromosome for the Learning

First, we investigate fitness characteristics for the parameter η. We fix generations $T = 5000$ and the roulette selection method in the experiments. Figs.3



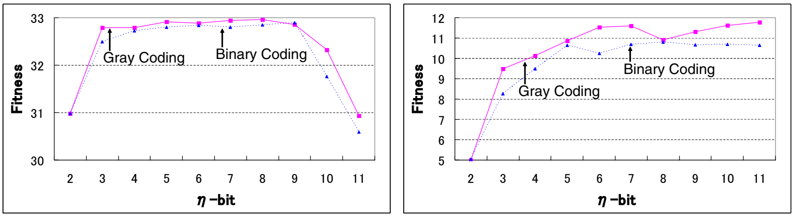
(a)Gray Coding (b)Binary Coding

Fig. 3. Fitness characteristics for the parameter η (7-bit function)



(a)Gray Coding (b)Binary Coding

Fig. 4. Fitness characteristics for the parameter η (two spirals problem)



(a)7-bit function (b)Two spirals problem

Fig. 5. Fitness at 5000 generation for the parameter η

and 4 show simulation results for 7-bit function and two spirals problem, respectively. Fig.5 shows the fitness values at 5000 generations in changing the parameter η from 2 to 11. These values are the average calculated by 100 trials. Figs.3 and 4 show that the convergent values of the fitness are low as the parameter η is small. As η increases, the combination numbers for network parameters increase. Then, the convergent values of the fitness increases. As η increases further, the combination numbers increase intensively. Then, calculation time in the GAETL increases intensively. In addition, convergence of the fitness can be slow as shown in Fig.3. These experimental results described above indicate that the chromosome length is enough if η can express sufficient combination numbers for network parameters. Also, Figs.3 to 5 show that the gray coding has better performance than the binary coding in the meaning of searching ability for the better network parameters.

As η and T increase, calculation time in the GAETL increases intensively. Hereafter, we fix $\eta = 7$ and $T = 100$ in which we can obtain reasonable fitness values as shown in the above results.

4.2 Comparison of Selection Methods in a GA

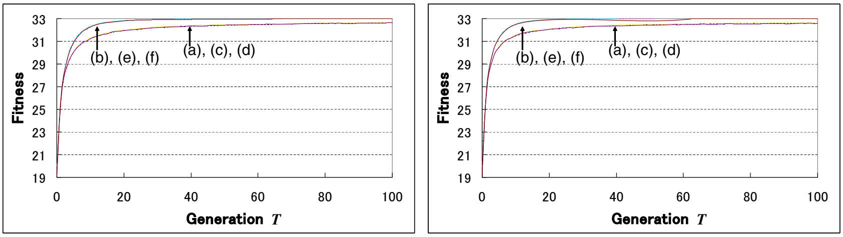
Next, we compare some selection methods in a GA. Adding to the roulette selection method, we consider "elitism" and "inverse-elitism" as the strategy. The inverse-elitism is a technique that prevents trapping local minimum [8]. A inverse elite individual is generated by reversing all bits of the genotype of an elite individual having the best fitness. Here, we also consider an inverse-elitism represented by reversing sign of the phenotype of an elite individual. We select one elite individual and one inverse elite individual in each generation. These individuals are certainly selected in the next generation. Based on the above condition, we compare the following six selection methods:

(a) roulette selection method (conventional method [6])
(b) (a) + 'elitism'
(c) (a) + 'inverse-elitism (all bits reversing)'
(d) (a) + 'inverse-elitism (sign reversing)'
(e) (a) + 'elitism' + 'inverse-elitism (all bits reversing)'
(f) (a) + 'elitism' + 'inverse-elitism (sign reversing)'

Figs.6 and 7 show generation-fitness characteristics for 7-bit function and two spirals problem, respectively. These fitness values are the average calculated by 2000 trials. Both Figs.6 and 7 show that the methods (b), (e) and (f) including elitism have better performances than the methods (a), (c) and (d). The effect of the inverse-elitism is not confirmed in case of determining only the first neuron in the hidden layer. Comparing coding methods, for 7-bit function, performances of the gray coding and the binary coding are the almost same as shown in Fig.6. However, for two spirals problem, the gray coding has better performance than the binary coding as shown in Fig.7.

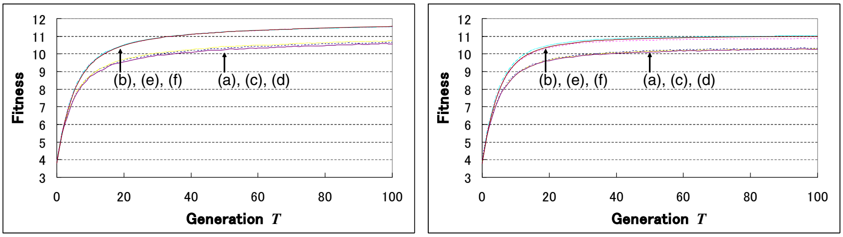
4.3 Comparison of the Number of Neurons in the Hidden Layer

Next, we investigate the statistics of M which is the number of neurons in the hidden layer in the end of learning for each method described above. The results are shown in Table 1. These values are the average calculated by 1000 trials. In the table, Min, Ave, and Dev denote minimum, average, and standard deviation, respectively. Table 1 shows that the method (e) or (f) has the best performances for each problem. Comparing coding methods, for 7-bit function, performances of the gray coding and the binary coding are almost the same. For two spirals problem, the gray coding has better performances than the binary coding. Two spirals problem is complex function as compared with 7-bit function. Therefore, the performances for each method might differ for more complex functions.



(a)Gray Coding (b)Binary Coding

Fig. 6. Fitness characteristics for some selection methods (7-bit function)



(a)Gray Coding (b)Binary Coding

Fig. 7. Fitness characteristics for some selection methods (Two spirals problem)

Table 1. Statistics of the number of hidden layer neurons in learning results

Problem	7-bit Function						Two spirals problem					
Coding	Gray Coding			Binary Coding			Gray Coding			Binary Coding		
Statistics	Min	Ave	Dev	Min	Ave	Dev	Min	Ave	Dev	Min	Ave	Dev
(a) T=100	2	2.238	0.4420	2	2.201	0.4343	19	25.03	3.105	19	26.08	3.253
(b) T=100	2	2.073	0.2787	2	2.050	0.2313	19	24.02	2.905	19	25.22	3.067
(c) T=100	2	2.220	0.4354	2	2.159	0.3738	19	24.86	3.084	20	26.00	3.165
(d) T=100	2	2.274	0.4764	2	2.166	0.4030	19	24.52	2.863	20	25.92	2.971
(e) T=100	2	**2.065**	0.2584	2	**2.036**	0.2018	19	23.96	3.050	19	**24.86**	2.825
(f) T=100	2	2.078	0.2755	2	2.058	0.2421	19	**23.75**	2.878	19	24.88	2.836

Finally, we discuss the number of generations in the GA. Ref.[6] has chosen the number of generations as $T = 5000$, and has gotten Min= 2 in 7-bit function and Min= 19 in two spirals problem. However we fix $T = 100$, almost the same performances can be obtained by properly setting selection methods and parameters in a GA. These results indicate that learning results for higher dimensional functions can be obtained in reasonable calculation time.

5 Conclusions

We have analyzed learning characteristics of GAETL for the three-layer BNN. First, we have shown that the chromosome length is enough if sufficient combination numbers for network parameters are expressible. We have also shown that the gray coding is more suitable than the binary coding as coding methods for BNN. Moreover, we have shown that for a complex function, the method including elitism and inverse-elitism can reduce the number of hidden layer neurons better than other selection methods. Finally, we have shown that the learning results can be obtained in the less number of generations by properly setting selection methods and parameters. Also, we show that the quality of the learning results can be almost the same as that of the conventional method.

Future problems include (1) analysis of learning characteristics of the GAETL to the BNN for more complex problems, (2) extension of learning algorithm to multi-bit output functions, and (3) applications to practical problems.

References

1. D.L.Gray and A.N.Michel, "A training algorithm for binary feed forward neural networks", IEEE Trans. Neural Networks, 3, 2, pp.176-194, 1992.
2. J.H.Kim and S.K.Park, "The geometrical learning of binary neural networks", IEEE Trans. Neural Networks, 6, 1, pp.237-247, 1995.
3. T.Windeatt and R.Tebbs, "Spectral technique for hidden layer neural network training", Pattern Recognition Letters, 18, 8, pp.723-731, 1997.
4. S.Gazula and M.R.Kabuka, "Design of supervised classifiers using Boolean neural networks", IEEE Trans. Pattern Anal. & Mach. Intell., 17, 2, pp.1239-1246, 1995.
5. X.-A.Wang and S.B.Wicker, "An artificial neural net Viterbi decoder", IEEE Trans. Commun, 44, 2, pp.165-171, 1996.
6. M.Shimada and T.Saito, "A GA-Based Learning Algorithm for Binary Neural Networks", IEICE Trans. Funds., E85-A, 11 pp.2544-2546, 2002.
7. L.Davis, Handbook of genetic algorithms, Van Nostrand Reinhold, 1991.
8. H.Kawanishi and M.Hagiwara, "Improved Genetic Algorithms using Inverse-Elitism", T.IEE Japan, 118-C, 5 pp.707-713, 1998.

Immune Clonal Selection Wavelet Network Based Intrusion Detection

Fang Liu and Lan Luo

School of Computer Science and Engineering, Xidian University,
Xi'an 710071, China
f63liu@163.com

Abstract. The ICSA (Immune Clonal Selection Algorithm) based structure and parameter learning of wavelet network for intrusion detection is proposed. The hierarchical structure is used in the coding scheme, thus we can realize evolution of topologic structure and the parameter learning of the wavelet network meanwhile. The experimental results show that the method based on ICSA can get higher true rate of IDS (Intrusion Detection System) than advanced wavelet network and Immune wavelet network. At the same time, the method proposed can reduce the false rate of IDS and has faster convergence speed in experiment.

1 Introduction

The problem of network intrusion detection is a division between the normal behaviors and the abnormal ones by network data stream ultimately. Therefore, it can be considered as a problem of pattern recognition. In most Neural Network (NN) applications, we will inevitably face the following problems: the network structure and other parameters lack the available theory guarantee. In this paper, we use the wavelet network .The wavelet space is regarded as the feature space for signal classification, the feature rules of signal are extracted by the NN classifier.

Learning process of an input-output mapping in the wavelet network can be regarded as the process of searching optimization. Then the optimum algorithm is the key for getting better performance of the wavelet network. Recently more and more scholars have proposed many new methods, such as Schaffer, J.D. [1], Mak, M.W. [2], Yao X. [3]. In these papers, genetic algorithm or evolutionary computation is applied to the learning of NN. A related research area, concerned with learning of network model, has been developing rapidly [4] [5] [6] [7].This paper describes a wavelet network model that uses ICSA to learn the parameters and structures.

2 Model of Wavelet Network (WN Model)

Zhang, Q.H [8] firstly presents a new type of network called wavelet network which is inspired by both the feed-forward neural networks and wavelet decomposition.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 331 – 336, 2005.

An input signal x(t) can be approximated by daughters of a mother wavelet $h_{a,b}(t)$:

$$\hat{x}(t) = \sum_{k=1}^{K} w_k h(\frac{t - b_k}{a_k}) \tag{1}$$

where $\hat{x}(t)$ is the simulated signal of $x(t)$, $h(t)$ is the mother wavelet and w_k, b_k, a_k are the weight coefficients, shifts, and dilations for each daughter wavelet. According to this, we presume the output of the n th unit of the p th sample y_n^p as following:

$$y_n^p = f\left[\sum_{k=1}^{K} w_{nk} \sum_{m=1}^{M} x^p(t_m)h\left[\frac{t_m - b_k}{a_k}\right]\right], \tag{2}$$

in which $f(z) = \frac{1}{1 + \exp(-z)}$, where w_{nk} is weight value between the k th unit in hidden layer and the nth unit in output layer.

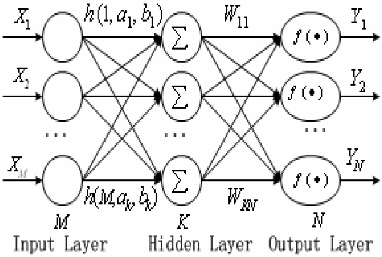


Fig. 1. The structure of wavelet networks

This approximation can be illustrated as Fig.1 which contains wavelet nonlinearities in the artificial neurons rather than the standard sigmoid nonlinearities. The hidden layers and the connection manner are alterable and the wavelet function selects Morlet wavelets.

The aim of learning algorithm in this WN model for intrusion detection is to adjust the energy function to the minimum. We employ the Least Mean Squares (LMS) energy for signal representation Min $E(w_k, b_k, a_k)$:

$$E = \frac{1}{2}\sum_{p=1}^{Q}\sum_{n=1}^{N}(y_n^p - \hat{y}_n^p)^2 \ . \tag{3}$$

3 The Theory of Immune Clonal Selection Algorithm (ICSA)

Clonal selection is an important part of the artificial immune system and ICSA has been presented by Jiao, L.C. [9], Du, H.F. [10], etc. Using the mechanism of clonal selection in the artificial immune system, they construct the clonal operator suitable to the artificial intelligent, such as immune operator, remembered operator, forgetting operator, etc. The main steps of ICSA are as following:

Step 1: *k*=0, initialize the antibody population $\vec{A}(0)$, set the algorithm's parameters and calculate the affinities of the initializing population;

Step 2: According to the affinities and the given integer relating to the scale of the clone, carry out the clonal operation, clonal mutation operation and clonal selection operation, obtain the new antibody population $\vec{A}(k)$;

Step 3: Calculate the affinities of the current population;

Step 4: *k*=*k*+1 if a stop condition is satisfied, stop the algorithm, else go to step2.

4 Immune Clonal Selection Wavelet Network for IDS

4.1 The Hierarchical Structure of the Chromosome Coding Scheme [11]

The structure of chromosome is arrayed by a series of genes hierarchically. Some genes control the status of others, called hierarchical structure. Shown in Fig 2, L is the maximum number of units in the hidden-layers. In this paper, every unit in the hidden-layers of wavelet network is coded as control genes of chromosome, and the weights and the wavelet parameters are coded as parameter genes accordingly.

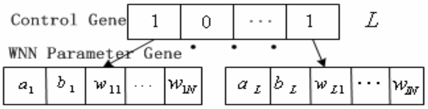


Fig. 2. Control genes and parameter genes in the hierarchical structure

4.2 Affinity Function and Related Operations

We need calculate the value of affinity function on each generation and each individual in the current population. So we set the affinity function as: $f = 1/E$.

- Conal Operation

$$T_c^C(\vec{A}(k)) = [T_c^C(A_1(k)) \quad T_c^C(A_2(k)) \quad \cdots \quad T_c^C(A_n(k))]^T \text{ where } q_i = Int\left(N_c \times \frac{f(A_i(k)}{\sum_{j=1}^{n} f(A_j(k))} \right),$$

N$_c$>*n, Int*(*x*) represents the minimum integer bigger than *x*. The clonal scale is adaptively adjusted according to the affinity between antibody and antigen. Then the antibody population changes into $\vec{A}'(k) = \{\vec{A}(k), \vec{A}_1'(k), \vec{A}_2'(k), \cdots, \vec{A}_n'(k)\}$, where

$\vec{A}_i'(k) = \{A_{i1}(k), A_{i2}(k), \cdots, A_{iq_i-1}(k)\},\ A_{ij}(k) = A_i(k),\ j = 1, 2, \cdots, q_i - 1$

- Clonal Mutation Operation

Unlike the general mutation operator in GA, in order to save the information of the original population, the clonal mutation is not performed on $A \in A'$. We carry out mutation operation on the current population which we get from the clonal operation according to the given probability of p_m^i , $\vec{A}''(k) = T_g^C(\vec{A}'(k))$.

- Clonal Selection Operation

For i=1,2,…,n, if existing the antibody after mutation: B={ $A_{ij}^{'}(k)$|max f($A_{ij}^{'}$), j=1,2,…, q_i-1}, then the probability in which B replaces the A_i(k)∈ $\overline{A}$ (k) is as follows.

$$p_s=\begin{cases}1 & f(A_i(k))<f(B)\\ \exp(-\frac{f(A_i(k))-f(B)}{\alpha}) & f(A_i(k))\geq f(B) \text{and} A_i(k) \textit{is not the best} \text{ an}\textit{tibody of population}\\ 0 & f(A_i(k))\geq f(B)\text{and}A_i(k)\textit{is the best} \text{ an}\textit{tibod of population}\end{cases} \tag{4}$$

where α>0, α is a value relating to the diversity of the population. The greater the diversity is, the larger the value of the α is.

5 Experimental Results

The MIT lpr system calls [12] are used to test the method. Firstly, we preprocess the data to get the short sequences of the traces (i.e. the k-length window is used to slip the system calls. Here k=6 according to Lee's research [13]). Secondly, the data is classified into two sets: training set and test set. We choose 500 normal traces randomly, and every trace contains 168 data to make up of 13850 data for training. For test set, we choose 100 traces that do not contain in the training set. The true rates and the false rates of the IDS are calculated on the test set.

In order to prevent the function from working in the plain region, we suggest the data should be transformed to be between 0.1~0.9 with the following formula.

$$y=\frac{0.9-0.1}{x_{\max}-x_{\min}}x+(0.9-\frac{0.9-0.1}{x_{\max}-x_{\min}}x_{\max}) \tag{5}$$

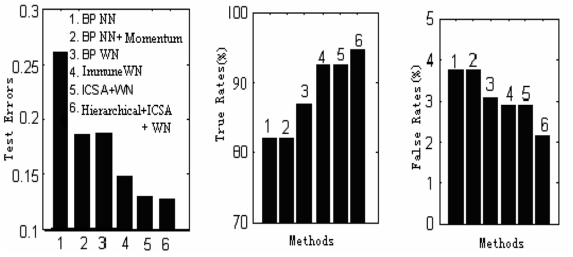
In the experiments, the clonal scale N_c is a given integer relating. Here N_c = 20. In the clonal selection algorithm, the probability of mutation is set to 0.5 and α 0.2. When the maximum iterative generation number equals to 1000 or not any improvements in the qualities of the network parameter values in the continuous t iterations (set t=40), the program stopped.

In order to express more clearly, we define the following parameters first: Detection Rate (the percentage of abnormal), False Positive (errors which normal data are identified as anomalous), False Negative (errors which intrusions are not identified), and False Rate (the sum of False Positives and False Negatives). The results of the following six algorithms are showed in Table 1, Fig. 3, Fig. 4 and Fig. 5. Each result is the statistical mean value of 50 runs. From Table 1 and Fig. 3, we can see that ICSA+WN and Hierarchical ICSA+WN have higher detection rate, lower false rate and lower test rate than the other four methods.

From Fig. 4, we can find that ICSA+WN and Hierarchical ICSA+WN have higher convergence speed than the other methods. Experiment results show that the method of ICSA based Wavelet Networks for Intrusion Detection is feasible and effective.

Table 1. The comparison of the six algorithms

Method	Test rate	Net error	Detection rate	False rate
BPNN	0.261	7.55%	82.1%	3.73%
BP+Moentum	0.186	7.26%	82.1%	3.73%
WN	0.187	5.21%	86.9%	3.06%
Immune+WN [14]	0.148	3.39%	92.5%	2.87%
ICSA+WN	0.129	3.10%	92.5%	2.87%
Hierarchical ICSA+WN	0.127	2.99%	94.7%	2.13%



(a) Test Error (b) Detection Rate (c) False Rate

Fig. 3. Comparison of the six algorithms mentioned above

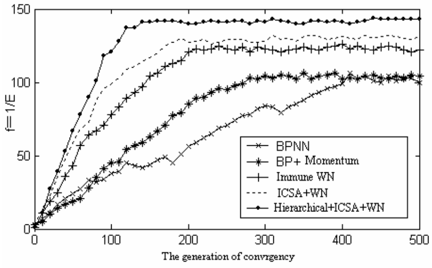


Fig. 4. The relationship between the object function and the number of generations

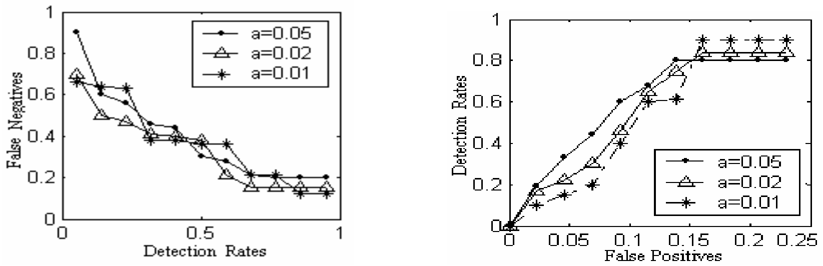


Fig. 5. The relationship between Detection Rates and False Negatives under different α (left), the relationship between Detection Rates and False Positives under different α (right)

Under the real conditions, however, it is impossible to find a training data that included all the normal actions. In this paper, we account the unsuitable number or proportions of system call sequences and we think this action abnormal, if it is larger than the given threshold α. Certainly, we wish to minimize False Rates (both False positives and False Negatives). But in most conditions, it is not realistic. Then we should try to find a balance point (i.e. threshold α) to obtain the best effects according to the practical questions and requests. From Fig.5, we can see the relationship between Detection rates, False Positives and False Negatives under different α .

References

1. Schaffer, J.D., Whitley, D., Eshelman, L.J.: Combinations of Genetic Algorithms and Neural Networks: A Survey of the State of the Art. In: Int. Workshop on Combinations of Genetic Algorithms and Neural Networks(COGANN-92), Los Alamitos (1992): 1-37
2. Mak, M.W., Cho. K.W.: Genetic Evolution of Radial Basis Function Centers for Pattern Classification. In: the proceedings of 1998 IEEE Int. Joint Conf. on Neural Networks (IJCNN'98). Anchorage. 1 (1998): 669-673
3. Yao, X.: Evolving Artificial Neural Networks. Proceedings of the IEEE. 87 (1999): 1423-1447
4. de Lacerda, E.G.M., de Carvalho, A.C.P.L.F., Ludermir, T.B.: Evolutionary Optimization of RBF Networks. In: Proceedings of Sixth Brazilian Symposium on Neural Networks. (2000): 219-224
5. Hofmann. A., Sick, B.: Evolutionary Optimization of Radial Basis Function Networks for Intrusion Detection. In: Proceedings of the International Joint Conference on Neural Networks. (2003) 20-24
6. Wang, L.N., Yu, G., Wang, G.R., Wang, D.: Method of Evolutionary Neural Network-Based Intrusion Detection. In: Proceedings of International Conferences on Info-tech and Info-net. Beijing. 5 (2001): 13-18
7. Wang, L., Pan, J., Jiao, L.C.: The Immune Algorithm. ACTA ELECTRONICA SINICA. (In Chinese) 28 (2000) 74-78
8. Zhang, Q.H., Benveniste, A.: Wavelet Networks. IEEE Transactions on Neural Networks. 3 (1992) 889-898.
9. Jiao, L.C., Du, H.F., etc.: The Prospect of the Artificial Immune System. ACTA ELECTRONICA SINICA (In Chinese). 31 (2003): 1540-1548.
10. Du, H.F., Jiao, L.C., Wang, S.A.: Clonal Operator and Antibody Clone Algorithms. In: Proceedings of the 2002 International Conference on Machine Learning and Cybernetics. Beijing, (2002) 506-510
11. Zheng, P.E., Ma, Y.H.: A New Hierarchical Genetic Algorithm for Training of RBF Neural Networks. Control and Decision (In Chinese). 15 (2000) 165-168
12. Warrender, C., Forrest, S., Pearlmutter, B.: Detecting Intrusions Using System Calls: Alternative Data Models. In: Gong, L., Reiter, M.K. (eds.): Proceedings of the 1999 IEEE Symposium on Security and Privacy. Oakland, CA: IEEE Computer Society Press. (1999) 133-145
13. Lee, W., Stolfo, S.J.: Data Mining Framework for Constructing Features and Models for Intrusion Detection Systems. PhD thesis. Columbia University. (1999)
14. Liu, F., Luo, L.: Adaptive Immune Wavelet Network based Intrusion Detection. Pattern Recognition and Artificial Intelligence (In Chinese). to appear. (2005)

Investigation of Evolving Populations of Adaptive Agents

Vladimir G. Red'ko[1], Oleg P. Mosalov[2], and Danil V. Prokhorov[3]

[1] Institute of Optical Neural Technologies, Russian Academy of Science,
Vavilova Str., 44/2, Moscow 119333, Russia
redko@iont.ru
[2] Moscow Institute of Physics and Technologies,
Institutsky per., 9, Dolgoprudny, Moscow region 141700, Russia
olegmos_@mail.ru
[3] Ford Research and Advanced Engineering, Ford Motor Company,
2101 Village Rd., MD 2036, Dearborn, MI 48124, U.S.A
dprokhor@ford.com

Abstract. We investigate an evolution model of adaptive self-learning agents. The control system of agents is based on a neural network adaptive critic design. Each agent is a broker that predicts stock price changes and uses its predictions for action selection. We analyzed different regimes of learning and evolution and demonstrated that 1) evolution and learning together are more effective in searching for the optimal agent policy than evolution alone or learning alone; 2) in some regimes the Baldwin effect (genetic assimilation of initially acquired adaptive learning features during the course of Darwinian evolution) is observed; 3) inertial switching between two behavioral tactics similar to searching adaptive behavior of simple animals takes place during initial stages of evolutionary processes.

1 Description of the Model

We investigate an evolution model of self-learning agents; the control system of the agent is based on a neural network adaptive critic design (ACD). The ACD includes two neural networks (NNs), model and critic. The model predicts the state of the environment for the next time step, and the critic is used to select actions on the basis of model predictions. These NNs can be optimized by both learning and evolution.

In comparison with other research of agents optimized by learning and evolution [1,2], our study concentrates on self-learning agents. Though our work is similar to that of Ackley and Littman [1], the control system of our agents is more advanced, because it is based on the well-investigated temporal difference algorithm [3]. As compared to other works on evolutionary optimization of NNs for reinforcement learning (see e.g. [4]), we pay the main attention to analysis of interaction between learning and evolution.

Agent Task. Inspired by [5], we consider an adaptive agent-broker. The agent capital $C(t)$ is distributed into cash and stocks. The fraction of stocks in the net capital of the agent is equal to $u(t)$. The environment is determined by the time series $X(t)$, t = 1,2,…, where $X(t)$ is the stock price at the moment t. The goal of the agent is to increase its capital $C(t)$ by changing the value $u(t)$. The capital dynamics is

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 337 – 342, 2005.

$$C(t+1) = C(t)\,[1 + u(t+1)\,\Delta X(t+1) / X(t)]\,, \tag{1}$$

where $\Delta X(t+1) = X(t+1) - X(t)$. Following [6], we use the logarithmic scale for the agent resource, $R(t) = \log C(t)$. The current agent reward $r(t)$ is defined by the expression: $r(t) = R(t+1) - R(t)$:

$$r(t) = \log\,[1 + u(t+1)\,\Delta X(t+1) / X(t)]\,. \tag{2}$$

For simplicity, we assume that the variable $u(t)$ takes only two values, $u(t) = 0$ (all in cash) or $u(t) = 1$ (all in stock).

Agent Control System. The agent control system is a simplified ACD that consists of two NNs: model and critic (see Fig.1).

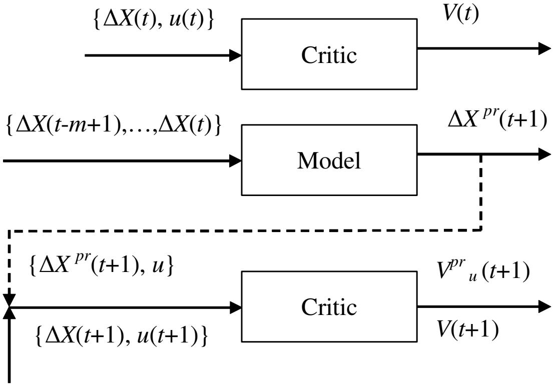


Fig. 1. Our ACD. The model predicts changes of the time series. The critic (the same NN is shown in two consecutive moments) forms the state value function $V(\mathbf{S})$ for the current state $\mathbf{S}(t) = \{\Delta X(t), u(t)\}$, the next state $\mathbf{S}(t+1) = \{\Delta X(t+1), u(t+1)\}$, and its predictions $\mathbf{S}^{\mathbf{pr}}{}_{\mathrm{u}}(t+1) = \{\Delta X^{pr}(t+1), u\}$ for two possible actions, $u = 0$ or $u = 1$.

The goal of the adaptive critic is to maximize utility function $U(t)$ [3]:

$$U(t) = \sum_{j=0}^{\infty} \gamma^j r(t+j), \qquad t = 1,2,... \tag{3}$$

where $r(t)$ is a reward obtained by the agent, and γ is the discount factor ($0 < \gamma < 1$). We suppose that the ACD state $\mathbf{S}(t)$ at moment t is characterized by two values, $\Delta X(t)$ and $u(t)$: $\mathbf{S}(t) = \{\Delta X(t), u(t)\}$.

The model predicts changes of the stock time series. The model output $\Delta X^{pr}(t+1)$ is based on m previous values of ΔX: $\Delta X(t-m+1),\dots,\Delta X(t)$, which are used as the model inputs. The model is implemented as a multilayer perceptron (MLP) with one hidden layer of tanh nodes and linear output. The critic is intended to estimate the state value function $V(\mathbf{S})$ (estimate of U in (3)). The critic is also a MLP of the same structure as the model.

At any moment t, the following operations are performed:

1) The model predicts the next change of the time series $\Delta X^{pr}(t+1)$.

2) The critic estimates the state value function for the current state $V(t) = V(\mathbf{S}(t))$ and the predicted states for both possible actions $V^{\mathrm{pr}}{}_{u}(t+1) = V(\mathbf{S^{pr}}_{u}(t+1))$, where $\mathbf{S^{pr}}_{\mathrm{u}}(t+1) = \{\Delta X^{pr}(t+1), u\}$, and $u = 0$ or $u = 1$.

3) The ε-greedy rule is applied [3]: with the probability 1 - ε the action corresponding to the maximum value $V^{pr}{}_{u}(t+1)$ is selected, and an alternative action is selected with the probability ε $(0 < \varepsilon << 1)$.

4) The selected action is carried out. The transition to the next time moment $t+1$ occurs. The current reward $r(t)$ is calculated in accordance with (2) and received by ACD. The value $\Delta X(t+1)$ is observed and compared with its prediction $\Delta X^{pr}(t+1)$. The NN weights of the model are adjusted to minimize the prediction error using the error backpropagation with $\alpha_M > 0$ as the model learning rate.

5) The critic computes $V(t+1)$. The temporal-difference error is calculated:

$$\delta(t) = r(t) + \gamma\, V(t+1) - V(t)\,. \tag{4}$$

6) The weights of the critic NN are adjusted to minimize the temporal-difference error (4) using its backpropagation and the gradient descent with $\alpha_C > 0$ as the critic learning rate.

Scheme of Evolution. An evolving population consists of n agents. Each agent has a resource $R(t)$ that changes in accordance with values of agent rewards: $R(t+1) = R(t) + r(t)$, where $r(t)$ is calculated in (2). At the beginning of any generation, initial resource of all agents is equal to zero.

Evolution passes through a number of generations, n_g=1,2,… The duration of each generation is T time steps. At the end of each generation, the agent having the maximum resource R_{max} (n_g) is determined. This best agent gives birth to n children that constitute a new (n_g+1)-th generation. The initial synaptic weights of both NNs form the agent genome **G**. The genome **G** does not change during agent life; however, temporary synaptic weights of the NNs **W** are changed during agent life via learning. At the beginning of (n_g+1)-th generation, we set for each newborn agent $\mathbf{G}(n_g+1) = \mathbf{G}_{\mathrm{best}}(n_g)$ + **mutations,** $\mathbf{W}_0(n_g+1)=\mathbf{G}(n_g+1)$. A normally distributed random value with zero mean and standard deviation P_{mut} is added to each synaptic weight at mutations.

2 Results of Simulations

The described model was investigated by means of computer simulations. We used two examples of model time series:

1) sinusoid:

$$X(t) = 0.5[1 + \sin(2\pi t/20)] + 1\,, \tag{5}$$

2) stochastic time series from [5, Example 2]:

$$X(t) = \exp[p(t)/1200]\,, \quad p(t) = p(t\text{-}1) + \beta(t\text{-}1) + k\,\lambda(t)\,, \quad \beta(t) = \alpha\beta(t\text{-}1) + \gamma(t)\,, \tag{6}$$

where $\lambda(t)$ and $\gamma(t)$ are two random normal processes with zero mean and unit variance, $\alpha = 0.9$, $k = 0.3$.

Some parameters were set to the same values for all simulations. Specifically, we set, $\gamma = 0.9$, $m = 10$, $\alpha_M = \alpha_C = 0.01$, $N_{hM} = N_{hC} = 10$, where N_{hM} and N_{hC} are numbers of hidden neurons of the model and critic. Other parameters (ε, P_{mut} , n, T) were set to different values, depending on the simulation, as specified below.

We analyze the following cases: 1) case L (pure learning); in this case we consider a single agent that learns by means of temporal difference method; 2) case E (pure evolution), i.e., evolving population without learning; 3) case LE, i.e., evolution combined with learning, as described above.

We compare the agent resource values attained during 200 time steps for these three cases of adaptation. For the cases E and LE, we set $T = 200$ (T is generation duration) and record the maximal value of agent resource in a population $R_{max}(n_g)$ at the end of each generation. For the case L, we have just one agent whose resource is reset $R(T(n_g\text{-}1)+1) = 0$ after the passing of every $T = 200$ time steps; the index n_g is incremented by one after every T time steps, i.e., $R_{max}(n_g) = R(T\,n_g)$. The plots R_{max} vs. n_g for the sinusoid (5) are shown in Fig. 2. In order to exclude the decrease of the value $R_{max}(n_g)$ due to the random choice of actions when applying the ε-greedy rule for the cases LE and L, we set $\varepsilon = 0$ after $n_g = 100$ for the case LE and after $n_g = 2000$ for the case L.

Under the optimal policy, the agent buys/sells stocks when stock price rises/falls. Analysis of agent behavior demonstrates that both pure evolution and evolution combined with learning are able to find the optimal policy. With this policy, the agent attains asymptotic value $R_{max} = 6.5$ (see Fig. 2). For the case L, the asymptotic value of R_{max} is only 5.4. Analysis reveals that the pure learning is able to find only the following satisfactory policy. The agent buys stocks when stock price rises (or falls by a small amount) and sells stocks when stock price falls significantly. So, the agent prefers to keep the capital in stocks.

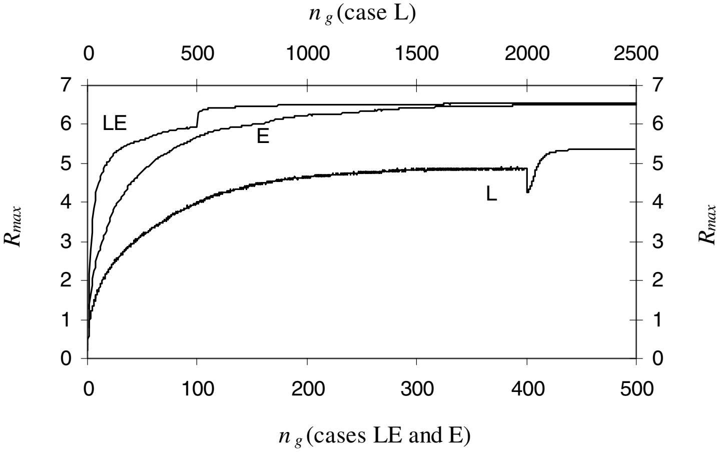


Fig. 2. The plots of $R_{max}(n_g)$ for the sinusoid (5). The curves LE, E and L correspond to the cases of evolution combined with learning, pure evolution and pure learning, respectively. Each point of the plots represents the average over 1000 simulations; $n = 10$, $T = 200$, $\varepsilon = 0.05$, $P_{mut} = 0.1$.

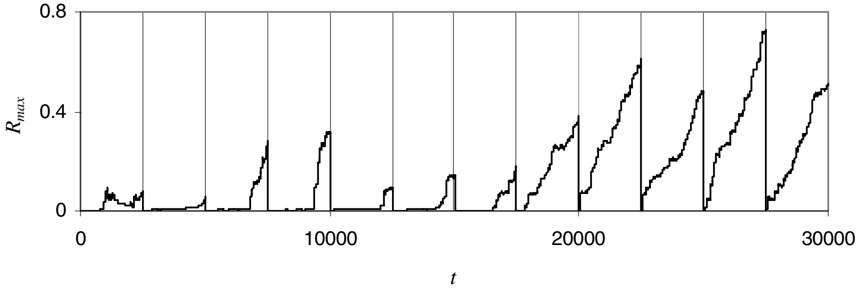


Fig. 3. The time dependence of the resource $R_{max}(t)$ of the best agent in the population; the stochastic time series with parameters $\varepsilon = 0.015$, $P_{mut} = 0.03$, $n = 10$, $T = 2500$; the case LE. The ends of generations are shown by vertical lines. During the early generations (generations 3 to 7), any significant increase of the agent resource begins only in the second half of the agent life. During the later generations (generations 9 to 12), the increase of the resource begins at the start of each generation, demonstrating that the advantageous policy becomes inherited.

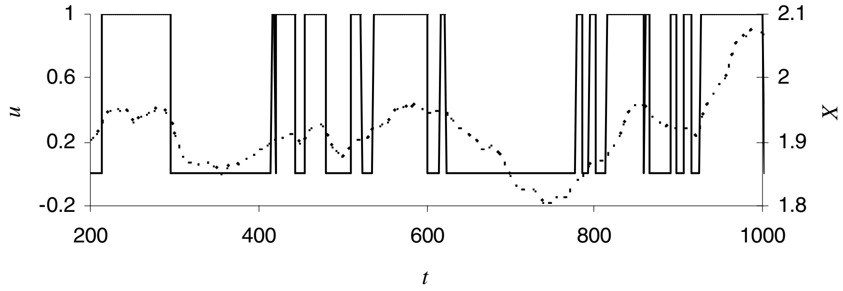


Fig. 4. Time dependence of action selection $u(t)$ for the best agent in the population (solid line) during the first stages of evolution (without learning) with parameters $P_{mut} = 0.1$, $n = 100$, $T = 200$. Time series $X(t)$ is also shown (dashed line). The agent has a rough policy that reflects only general features of changing environment.

Thus, pure learning is imperfect in our simulation, nevertheless, learning helps evolution to attain larger values of R_{max} faster (see curves E and LE in Fig.2). We also confirm that learning helps evolution to find a good policy by some other simulations. For example, Fig. 3 demonstrates that during initial stages of evolution (n_g = 2-7) a satisfactory policy is found only in the second half of the agent life (via learning). During later stages of evolution (n_g = 9-12) the agents exhibit a satisfactory policy right from the beginning of the generation. This phenomenon is known as the Baldwin effect [1,2,7], i.e., initially acquired features become inhered.

For the case of evolution alone, an interesting type of behavior is observed in the first stages of evolution. The agent has a rough policy that reflects only general features of changing environment (Fig. 4). The agent buys/sells stocks when the stock rises/falls significantly, and it ignores small and short-term variations of the stock price. There exists an inertia in switching between two tactics of behavior (sell stocks and buy stocks). This inertial behavior is very similar to that of simple animals, such as caddis fly larvae [8]; it helps an animal to react adaptively to only general large-scale patterns in environment.

Acknowledgments

The authors thank Dr. Valentin A. Nepomnyashchikh for consultations on caddis fly larvae behavior, as well as three anonymous reviewers for many helpful comments. This work is supported in part by the Russian Foundation for Basic Research, Grant No 05-07-90049.

References

1. Ackley, D., Littman, M.: Interactions between Learning and Evolution. In C. G. Langton, et al (Eds.), Artificial Life II. Reading MA: Addison-Wesley (1992) 487-509
2. Suzuki, R., Arita, T.: Interactions between Learning and Evolution: Outstanding Strategy Generated by the Baldwin Effect. Biosystems, 77 (2004) 57-71
3. Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. Cambridge: MIT Press (1998)
4. Braun, H., Ragg, T.: Evolutionary Optimization of Neural Networks for Reinforcement Learning Algorithms. In: ICML96, Workshop Proceedings on Evolutionary Computing and Machine Learning, Italy (1996) 38-45
5. Prokhorov, D., Puskorius, G., Feldkamp, L.: Dynamical Neural Networks for Control. In J. Kolen and S. Kremer (eds.) A Field Guide to Dynamical Recurrent Networks. IEEE Press, (2001) 23-78
6. Moody, J., Wu, L., Liao, Y., Saffel, M.: Performance Function and Reinforcement Learning for Trading Systems and Portfolios. Journal of Forecasting, 17 (1998) 441-470
7. Baldwin, J.M.: A New Factor in Evolution. American Naturalist, 30 (1896) 441-451
8. Nepomnyashchikh, V.A.: Selection Behaviour in Caddis Fly Larvae. In R. Pfeifer et al. (eds.) From Animals to Animats 5: Proceedings of the Fifth International Conference of the Society for Adaptive Behavior. Cambridge, USA: MIT Press (1998) 155-160

Enhancing Cellular Automata by an Embedded Generalized Multi-layer Perceptron

Giuseppe A. Trunfio

University of Calabria,
Center of High-Performance Computing, Rende (CS), Italy
trunfio@unical.it

Abstract. A hybrid approach combining Cellular Automata (CA) and Artificial Neural Networks (ANNs), capable of providing suitable dynamic simulations of some complex systems, is formalized and tested. The proposed method allows to incorporate in the CA transition function the available a priori knowledge of the interaction rules between the elementary system constituents. In order to effectively describe the remaining unknown local rules, an embedded ANN is exploited. The ANN component of the transition function is designed, on the basis of the available data about the emerging behavior of the system to be simulated, by an evolutionary strategy involving both the architecture and weights.

1 Introduction and Model Formalization

Computer simulation of the dynamic evolution of complex systems has become a fundamental tool for many scientific activities. To this end the Cellular Automata (CA) approach was successfully applied in different areas, including ecology, biology, geology, medicine, urban studies and many others. Actually, CA are good candidates for modelling complex dynamical systems whose evolution depends on the local interactions of their constituent parts. As it is well known, the main issue in CA modelling is often represented by the elicitation of the local interaction rules. In order to cope with the situation in which not enough prior system's knowledge is available, many approaches have been exploited. In particular some researchers have proposed, for specific applications, the use of an Artificial Neural Network (ANN) to model the local interactions (e.g. see [1]). In this paper a quite general method for embedding an ANN into the CA transition function is formalized and used in a test case. The proposed approach exploits an effective evolutionary procedure to design the ANN [2,3].

In order to allow the modelling of "real" systems, the adopted CA model is more general than the classical one. In particular, let $\mathcal{D} \in \mathbb{R}^d$ be a d-dimensional domain and $D = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_n\}$ its discretisation in n cells, each one representing a space site specified by a d-dimensional index vector $\mathbf{c}_i \in \mathbb{Z}^d$. Every relevant characteristic, relative to the space portion corresponding to a given cell, let be described as a scalar variable s_i belonging to a nonempty set $\mathcal{R}(s_i)$ [4]. Thus, to each cell $\mathbf{c}$ it corresponds a vector *state* $\mathbf{s} = \mathbf{s}(\mathbf{c}) = [s_1, \ldots, s_m]^T$ belonging to the

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 343–348, 2005.

set $\mathcal{R}(s_1) \times \cdots \times \mathcal{R}(s_m)$. A p-dimensional *sub-state* of a cell, with $p \leq m$, let be defined as the sub-vector $\mathbf{u}$ collecting only a subset of the variables included in $\mathbf{s}$. Given a generic set of cells $Y \subseteq D$, with $\mid Y \mid= k$, let us define the $p \times k$ matrix $\boldsymbol{\Omega}_{Y\mathbf{u}}$ whose columns are the vectors $\mathbf{u}(\mathbf{c}_j)$, $\forall\, \mathbf{c}_j \in Y$, as the *configuration* of Y with respect to the p-dimensional sub-state $\mathbf{u}$. In the following, $\boldsymbol{\Omega}_{Y\mathbf{u}}$ with $\mathbf{u} \equiv \mathbf{s}$ will be simply indicated as $\boldsymbol{\Omega}_{Y}$. The cell states of the set $D_A \subset D$ of *active cells*, simultaneously evolve in discrete steps. To each $\mathbf{c} \in D_A$ a set $N \subset D \setminus \{\mathbf{c}\}$ called *neighbourhood* is associated: the states of the cells belonging to N can influence the evolution of $\mathbf{s}(\mathbf{c})$. At each time-step the state of a cell is updated using its state and the neighbourhood configuration $\boldsymbol{\Omega}_N$ from the previous time-step, according to the *transition function*

$$\mathbf{s}^{(t+1)} = \varphi(\mathbf{s}^{(t)}, \boldsymbol{\Omega}_N^{(t)}) \qquad (1)$$

where the superscripts refer to the time step. Let us suppose that Eq. (1) results from the sequential application of n_t elementary local transformations

$$\mathbf{s}^{(t,j)} = \phi_j(\mathbf{s}^{(t,j-1)}, \boldsymbol{\Omega}_N^{(t)}), \quad j = 1...k-1 \qquad (2)$$

$$\mathbf{s}^{(t,k)} = \psi(\mathbf{s}^{(t,k-1)}, \boldsymbol{\Omega}_N^{(t)}, \mathbf{W}) \qquad (3)$$

$$\mathbf{s}^{(t,j)} = \phi_j(\mathbf{s}^{(t,j-1)}, \boldsymbol{\Omega}_N^{(t)}), \quad j = k+1...n_t \qquad (4)$$

where $\mathbf{s}^{(t,0)} = \mathbf{s}^{(t)}$, $\mathbf{s}^{(t+1)} = \mathbf{s}^{(t,n_t)}$. Functions ϕ_j incorporate the knowledge of some local interactions, whereas ψ uses an ANN with connection-weights $\mathbf{W}$.

1.1 Neural Component of the Transition Function

In this work the local transformation ψ is based on the generalized multi-layer perceptron shown in Fig. 1 [2]. The ANN's inputs are a p-dimensional sub-state $\mathbf{u}^{(t,k-1)}$ of the current cell, updated by the $k-1$ local transformations ϕ_j, and the neighbourhood configuration with respect to a r-dimensional sub-state $\mathbf{z}^{(t)}$, i.e. $\boldsymbol{\Omega}_{N\mathbf{z}}^{(t)} = [\sigma_{ij}^{(t)}]$, given by the previous time step. The output is a q-dimensional sub-state $\mathbf{v}^{(t,k)}$ of the cell. In order to account for their potentially different range of variations, the transformations $F_{y_i} : \mathcal{S}(y_i) \to [0,1]$ are preliminary applied to each scalar input y_i. Thus, the ANN is initialized as

$$\begin{aligned} x_{(j-1)\times r+i} &= F_{z_i}(\sigma_{ij}^{(t)}), \;\; i = 1\ldots r,\; j = 1\ldots b, \quad \text{with } b =\mid N \mid \\ x_{b\times r+i} &= F_{u_i}(u_i^{(t,k-1)}), \;\; i = 1\ldots p \end{aligned}$$

besides the bias unit $x_{n_i} = -1$, where $n_i = b \times r + p + 1$. Units x_i with $i > n_i$ can take inputs from units x_j with $j < i$ and are updated as follows

$$x_i = \frac{1}{1+e^{-y}}, \quad \text{for} \quad i = n_i + 1 \ldots n_i + n_h + q, \quad \text{where} \quad y = \sum_{j=1}^{i-1} w_{ij} x_j \qquad (5)$$

being n_h the number of hidden units. Finally from the units up to the $(n_i{+}n_h)$-th the outputs is the sub-state $\mathbf{v}^{(t,k)}$ of the cell

$$v_i^{(t,k)} = F_{v_i}^{-1}\left(x_{n_i+n_h+i}\right), \quad i = 1\ldots q \qquad (6)$$

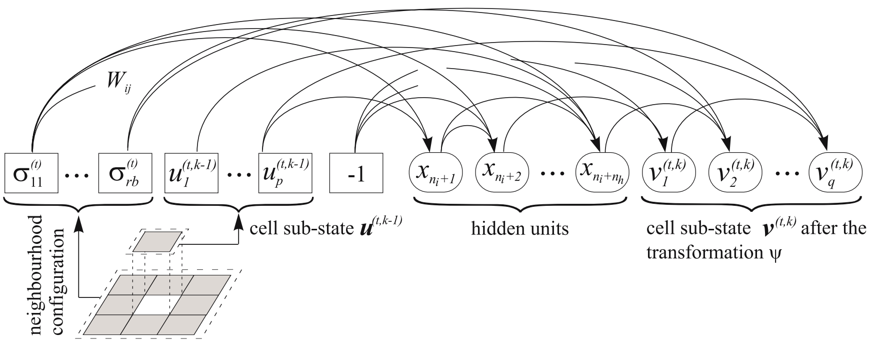

W_{ij}
$\sigma_{11}^{(t)}$ ··· $\sigma_{rb}^{(t)}$
$u_1^{(t,k-1)}$ ··· $u_p^{(t,k-1)}$
-1
x_{n_i+1}
x_{n_i+2} ··· $x_{n_i+n_h}$
$v_1^{(t,k)}$
$v_2^{(t,k)}$ ··· $v_q^{(t,k)}$
neighbourhood configuration
cell sub-state $u^{(t,k-1)}$
hidden units
cell sub-state $v^{(t,k)}$ after the transformation ψ

Fig. 1. The transformation ψ implemented as a generalized multi-layer perceptron

1.2 Design Phase

All the local transition functions (1), acting simultaneously on each cell, can be thought as an overall transition function Φ which acts on the entire automaton and gives the global configuration at the step $t+1$ as $\boldsymbol{\Omega}_D^{(t+1)} = \Phi(\boldsymbol{\Omega}_D^{(t)}, \mathbf{W})$. Thus the iterative application of the function Φ to the successive configurations, starting from an initial one $\boldsymbol{\Omega}_D^{(0)}$, leads to the dynamic process

$$\boldsymbol{\Omega}_D^{(0)} \xrightarrow{\Phi} \boldsymbol{\Omega}_D^{(1)} \xrightarrow{\Phi} \cdots \xrightarrow{\Phi} \boldsymbol{\Omega}_D^{(t)} \tag{7}$$

representing the automaton evolution. For a given set $Y \subseteq D$ we can write

$$\boldsymbol{\Omega}_Y^{(t)} = \Phi_Y^t(\boldsymbol{\Omega}_D^{(0)}, \mathbf{W}) \tag{8}$$

expressing the configuration of Y at the time step t as a function of the initial global configuration and the connection weights, being fixed the other automaton characteristics (i.e. the model structure). Supposing $\boldsymbol{\Omega}_D^{(0)}$ be assigned, we can think at (8) as a nonlinear relation, parameterized by $\mathbf{W}$, which relates the *input* $\boldsymbol{\Omega}_D^{(0)}$ and the *output* $\boldsymbol{\Omega}_Y^{(t)}$ at time step t. Clearly the automaton evolution (7) can be, in general, significantly influenced by the neural component (3) of the local transition function. Therefore the ANN must be characterized by proper architecture and connection weights, in order to correctly and efficiently reproduce the real system behavior starting from an input $\boldsymbol{\Omega}_D^{(0)}$. In particular, especially in this dynamic context, a very suitable feature of the ANN is its generalization capability with respect to the possible neighbourhood configurations: the number of such configurations being very high in general, the ANN used as transition function must very often cope with a number of unknown inputs during simulations. Another critical characteristic is related to the computational cost: since the ANN is invoked for every automaton cell and time step, the number of invocations per simulation may easily reach the order of many millions.

Hence, a little optimization in the ANN structure may result in a big economy of the required computational resources.

In this work we attempt to obtain the desirable ANN characteristics listed above, exploiting an evolutionary learning procedure, in which both the structure and connection weights are evolved. Let us suppose that a number n_e of experiments of the real system behavior has been performed, each one characterized by the initial automaton configuration and a series of consequent configurations

$$\bar{\boldsymbol{\Omega}}_D^{(i,j)}, j \in \{0, \tau_1, ..., \tau_{n_c(i)}\}, \quad i = 1...n_e \tag{9}$$

where τ indicates the time step in which a configuration is known and $n_c(i)$ is the number of known configurations from the i-th experiment. In general, the desired time granularity of the simulation is smaller than that resulting from the frequency of the available experimental configurations, i.e. $\tau_{i+1} > \tau_i + 1$, and hence no cell state transition examples are explicitly available. This makes appropriate to design the ANN using a supervised indirect approach, which in this work is represented by an evolutionary process driven by the global error between the emerging simulated behavior and the experimented one. To this end for a given set of cells $Y \subseteq D$ we can introduce a *misfit function* Θ giving a measure of the error as

$$e_Y^{(i,j)} = \Theta\left(\boldsymbol{\Omega}_Y^{(i,j)};\ \bar{\boldsymbol{\Omega}}_Y^{(i,j)}\right) = \Theta\left(\Phi_Y^j(\bar{\boldsymbol{\Omega}}_D^{(i,0)}, \mathbf{W});\ \bar{\boldsymbol{\Omega}}_Y^{(i,j)}\right) \tag{10}$$

The model error over the subset Y, also accounting for the use of a particular ANN in the transition function, can be computed introducing two proper aggregation (e.g. maximum, minimum or some kind of average) functions

$$\epsilon_Y = \mathop{\Gamma}_{i=1}^{n_e} \mathop{\Lambda}_{j=1}^{n_c(i)} e_Y^{(i,j)} \tag{11}$$

where Γ aggregates errors over experiments and Λ over configurations. In order to improve the generalization ability of the evolved networks, the cell set D is partitioned (e.g. trough a uniform random sampling) into a *training set* T and a *validation set* V. Thus, using the equations (10) and (11), for a given ANN the errors ϵ_T and ϵ_V can be computed performing n_e simulations. The adopted evolutionary strategy is similar to that proposed by Yao and You in [2], with the difference that in the present case the training of an individual is not based on a direct approach, being instead performed exploiting the evolutionary procedure based on the Cauchy mutation proposed in [3]. In the design algorithm, n_a sub-populations of n_w individuals (i.e. ANNs) are evolved, where each sub-population is composed by individuals with the same architecture and different connection weights. Since both architectures and weights are evolved, a direct encoding scheme is used, that is, the ANN is represented by the connectivity matrix $\mathbf{K}$, the connection weight matrix $\mathbf{W}$ and the hidden node vector $\mathbf{h}$. In particular, the evolution of the connection weights is performed on the basis of the training set, while the evaluation of the sub-population goodness is based on the validation set as follows: *(i)* as a result of the weights evolution procedure, the best sub-population individual (i.e. the one having the smaller ϵ_T) is available; *(ii)* the sub-population error is defined as the value of ϵ_V error of the so selected ANN. The procedure uses rank-based selection and five mutation: *(i)* connection weights change, which is obtained through a few generations of the evolutionary procedure mentioned above; *(ii)* node deletion; *(iii)* connection deletion; *(iv)* node addition; *(v)* connection addition. The mutations are attempted sequentially and if one mutation leads to a better ANN it is regarded as successfull and no other mutations are applied, promoting in this way compact ANNs [3].

2 A Preliminary Application

The proposed framework was applied to the simulation of the wildfire spreading in the complex orography represented in Fig. 2 and in presence of heterogeneous fuels. It is known that fires show a higher propagation velocity when they climb up an up-ward slope, whereas fires show a smaller velocity when they descend a downward slope. Besides, wind speed and direction as well as the fuel characteristics, greatly affect the spreading. In order to exploit the performance of the hybrid ANN-CA in this case, the data used for the learning phase were obtained as outcomes of the "standard" CA model described in [5], which is based on the knowledge of empirical relations experimentally derived. The ANN-CA used here is composed of squared cells and the neighbourhood pattern includes a double order of squares around the cell (i.e. $b =\mid N \mid=24$). The cell state is defined by the following variables: *(i)* the *cell altitude* z; *(ii)* the *wind speed components* w_x and w_y;*(iii)* the *fuel kind* f_k which can take 13 different values; *(iv)* the *potential spread rate* r_0, i.e. a real number representing the spread rate on a flat terrain and without wind; *(v)* the *fuel virtual temperature* t_v, assuming values in the interval $[0, 1]$; *(vi)* the *combustion state* c_s which takes one of the values "inflammable" and "burning". The first local transformation ϕ_1 is devoted to the computation of r_0: if $c_s \neq$"burning" $r_0 = 0$, otherwise it is computed using an empirical relation [5] based on the fuel kind characteristics defined by the variable f_k (i.e. the reaction intensity, the propagation flux ratio, the heat of pre-ignition, the ovendry bulk density and the effective heating number). Then, if c_s="inflammable", the neural component ψ is used for the computation of t_v, taking as input the sub-states $\mathbf{z} = [z, r_0]^T$ and $\mathbf{u} = [z, w_x, w_y]^T$, where each variable is rescaled to between 0.0 and 1.0 by a linear function. Finally, it is applied the transformation ϕ_2 defined as follows: if $t_v > 0.5$ then c_s="burning". The ignition cell is initialized setting $t_v = 1$ and c_s="burning". The ANN takes 52 inputs and gives a single scalar output. In the performed test the automaton was composed of 80 × 140 square cells of side 20 m, while the simulation time steps were 96. The ANN evolutionary design phase was based on the knowledge of four configurations (i.e., the initial one plus three) in two experiments characterized by different ignition cell and wind conditions (i.e., eight configurations in total). The functions Γ and Λ in equation (11) were defined as arithmetic means, while the error Θ was defined as $\Theta = 1- \mid B \cap \bar{B} \mid / \mid B \cup \bar{B} \mid$, where $B \subseteq D$ and $\bar{B} \subseteq D$ are the simulated and "real" set of cells with cs="burning", respectively. The

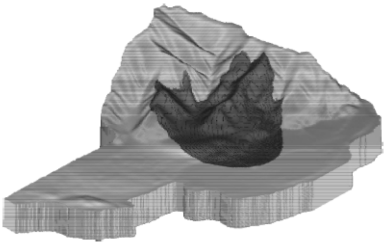


Fig. 2. The tested orography and fire spreading

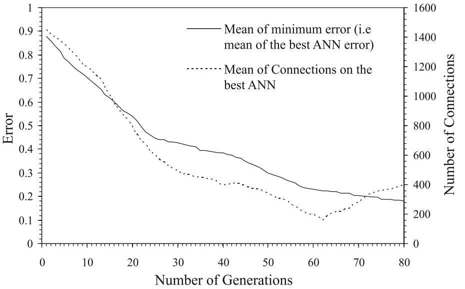


Fig. 3. Evolution of ANN's connections and accuracy

evolutionary strategy was based on a population of $n_a = 20$ sub-populations, each one composed by $n_w = 20$ individuals with the same architecture but different connection weights. Fig. 3 reports the results of the ANN design procedure averaged over 20 runs. All runs took less than 80 generation to finish, producing an ANN with an average error on the validation set of 0.18. The best ANN in terms of error was composed of 22 hidden units and 384 connections and had the error of 0.16 on the validation set. After the design phase, this ANN was exploited in 30 test cases randomly generated in terms of ignition cell and wind conditions, obtaining an average error on the final configuration of 0.24.

3 Conclusions and Future Work

The first application has shown that the proposed approach can become useful in improving the CA modelling abilities. Although the learning phase is quite time consuming, it needs to be carried out only once and nowadays parallel computing resources are very commonly available. Clearly, further insights are required in order to assess the effectiveness and the reliability of the method. Possible future developments may include the explicitation of global parameters (e.g., the cell or the time step sizes) and the use of neural network ensembles.

References

1. Yeh, A.G.O., Li, X.: Neural-network-based cellular automata for simulating multiple land use changes using GIS. Int. Journal of Geogr. Inf. Science **16** (2002) 323–343
2. Yao, X., Liu, Y.: A new evolutionary system for evolving artificial neural networks. IEEE Transactions on Neural Networks **8** (1997) 694–713
3. Yao, X., Liu, Y., Lin, G.: Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation **3** (1999) 82–102
4. Di Gregorio, S., Serra, R.: An empirical method for modelling and simulating some complex macroscopic phenomena by cellular automata. Future Generation Computer Systems **16** (1999) 259–271
5. Trunfio, G.A.: Predicting wildfire spreading through a hexagonal cellular automata model. Volume 3305 of Lecture Notes in Computer Science. (2004) 385–394

Intelligent Pattern Generation for a Tactile Communication System

C. Wilks and R. Eckmiller

Division of Neural Computation, Department of Computer Science,
University of Bonn, D-53117, Germany
{wilks,eckmiller}@nero.uni-bonn.de

Abstract. In this paper, we present an optimization method for a learning algorithm for generation of tactile stimuli which are adapted by means of the tactile perception of a human. Because of special requirements for a learning algorithm for tactile perception tuning the optimization cannot be performed basing on gradient-descent or likelihood estimating methods. Therefore, an Automatic Tactile Classification (ATC) is introduced for the optimization process. The results show that the ATC equals the tactile comparison by humans and that the learning algorithm is successfully optimized by means of the ATC.

1 Introduction

In this paper the generation of tactile stimulus/information assignments that can be used for communication is discussed. The challenge is to develop and optimize a learning algorithm that can be used for tuning tactile pattern in order to obtain the stimulus/information assignments.

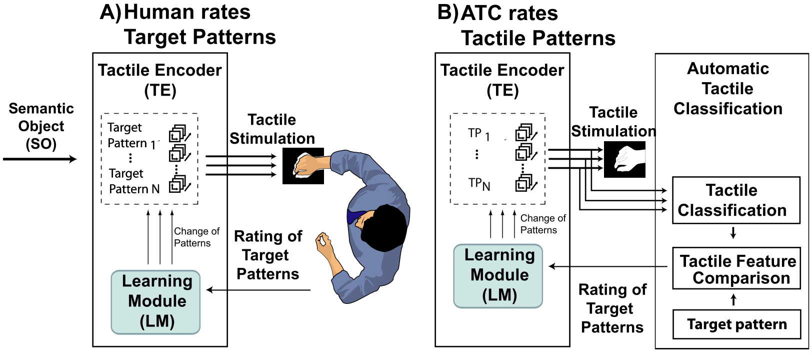


Fig. 1. Principle of learning induced tactile communication

The principal schema of tactile tuning is shown in figure 1A. It consists of the technical system (TS) and the human himself (HU). The tactile encoder (TE) generates different tactile patterns TP_1 - TP_N and stimulates the fingertips of the human. The human rates the evoked perceptions to each other and gives a feedback to the learning module LM. Depending on this feedback the LM changes the tactile patterns TP_1 -

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 349–356, 2005.

TP_N and presents the patterns again to the human. It is in this iterative way that the tactile patterns are generated according to the idea of the human. In this paper it is discussed how the LM changes the tactile patterns TP_1 - TP_N in order to adapt them to the human and how to minimize the iteration time. A detailed description of the setup is given in [1]

2 Specific Requirements for Learning Algorithms for Tactile Perception Tuning

- Upper bound of iteration time
  Due to loss of attention and concentration the human can only interact with the TS for a limited time period. Psychophysical tests recommend an interaction time of about 30 minutes as an upper bound. A human needs about 2s - 4s to rate a tactile pattern. Derived from this the upper bound for the number of iterations n is $n_{iteration} = \frac{30min}{3s} = 600$.
- Incomplete data
  Patterns used to develop the system are only a subset of all features of the tactile stimulator. So the learning algorithm has to be able to produce new kinds of tactile patterns that were not considered during the development of the system.

3 Methods

During the optimization a lot of different parameters have to be tested to find optimal parameters and therefore many more than 600 iterations are performed. When searching optimal parameters constant conditions are needed to reproduce a result and to compare different parameters. Both requirements, the great number of iterations and the constant conditions, cannot be achieved when the human is involved.

Therefore an automatic rating routine has to be implemented to optimize the learning algorithm. Once the learning algorithm is optimized the human can interact with the optimal learning algorithm.

The optimization of the learning algorithm for tactile tuning is divided into two phases:

1. Creation of an automatic tactile classification (ATC)
   The human rates given target patterns to each other and the ATC is adapted to reproduce these ratings.
2. Optimization of the learning algorithm.
   Figure 1B shows the setup of the optimization of the learning algorithm by means of the ATC. The ATC is subdivided into two subsystems, a Tactile Classification and a Tactile Feature Comparison.

4 Implementation

4.1 Automatic Tactile Classification

The basis of the automatic rating routine is a tactile pattern classification. Now the question is, how the tactile sense classifies tactile stimuli. The tactile neural information are processed by the somatosensory system of the human. It passes different parts of the tactile memory whereas the neural information are differently represented in the different parts of the memory [3]. The sensory registers are the input storage of neural information, where the information are only stored for some ms. The next part is the tactile working memory [4][5]. Here, the information loss depends on an internal rehearsal. If there is an internal representation and a rehearsal is possible, the information can be stored in the working memory much longer. In case of rehearsal the information can be passed to the third part of the memory model, the long-term memory.

P. Mahrer und C. Miles investigated the question of what kind of internal representation is used for tactile stimuli and found out, that the human applies a representation that is visual-spatial- or verbal-based [6]. Thus, the features of the tactile classification are derived from the visual-spatial and verbal features. Additional features of the tactile sense are added to the list of tactile classification features, shown in table 1.

Table 1. List of tactile classification features

visual-spatial	verbal	tactile
Movement of a stimulus,	Length of a stimulus,	Stimulus of different fingers,
Intensity of a stimulus,	Rhythm of a stimulus,	Division of one finger into 4 areas,
Activity of a stimulus		

The tactile stimulator consists of 40 tactile actuators with each actuator having two activation states, up(1) and down(0). The time resolution of a tactile pattern is 20 ms. At a given time t the tactile pattern is defined by the activity of the 40 actuators which are parameterized by time frames such as $a(t) = a_1^t \dots a_{40}^t, \quad a_i \in \{0, 1\}$, whereas a_i is the actuator activity, $p_x(i)$ and $p_y(i)$ are the x/y-position of the i-th actuator. From this parametrization a time series of the features of table 1 is calculated.

1) Center of movement $\overline{cm_x}$ and $\overline{cm_y}$ of a stimulus

$$\overline{cm_x} = \tfrac{cm_x}{n_a}, \quad cm_x(t) = \sum_{i=0}^{40} x_i, \begin{cases} x_i = p_x(i), & \text{if } a(t, p_i) = 1 \\ x_i = 0 & \text{otherwise} \end{cases}$$

$$\overline{cm_y} = \tfrac{cm_y}{n_a}, \quad cm_y(t) = \sum_{i=0}^{40} y_i, \begin{cases} y_i = p_y(i), & \text{if } a(t, p_i) = 1 \\ y_i = 0 & \text{otherwise} \end{cases}$$

n_a= number of active actuators ($a(t, p_i) = 1$)

2) Length of a stimulus

l = maximum number of time frames a(t) of a tactile pattern

3) Order of stimulation of fingers

The fingers are labeled from f_1(thumb) to f_5(little finger).

$$f_{finger}(t) = \sum_{i=1}^{5} f_i, \begin{cases} f_i = 2^i \text{ if finger i is stimulated at t} \\ f_i = 0 \text{ otherwise} \end{cases}$$

4) Rhythm of a stimulus
   A perception of a rhythm is evoked if the time frames of a tactile pattern change as follows:
   - N_1 constant time frames
   - change of the time frame
   - N_2 constant time frames
   - change of the time frame
   - ...
   - N_n constant time frames

   Therefore, the number of constant time frames is counted and stored in a time series $f_{rhythm}(t) = [N1..N_n]$.

5) Division of one finger into four areas
   The fingertips are divided into four spatial areas. Labeled from fingertip(4) down to the palm(1).
   $f_{area}(t) = \sum\limits_{i=1}^{4} f_i, \begin{cases} f_i = 2^i \text{ if finger area of one finger is stimulated} \\ f_i = 0 \text{ otherwise} \end{cases}$

6) Intensity of a stimulus
   The intensity of a tactile pattern corresponds to the number of active actuators per time.
   $intensity(t) = \sum\limits_{i=0}^{40} i_{count}, \begin{cases} i_{count} = 1 \text{ if } a(t, p_i) = 1 \\ i_{count} = 0 \text{ otherwise} \end{cases}$

7) Activity of a stimulus
   The activity of a tactile pattern is the difference between the intensity for t+1 and t.
   $activity(t) = intensity(t + 1) - intensity(t)$.

Two tactile patterns are compared by comparing the time series of each feature by means of the dynamic time wrapping algorithm (DTW)[7]. There are weighting coefficients of the features g_1 - g_7 which are defined by adaptation to real user data. The human and the ATC have to arrange 2 tactile patterns P_1 and P_2 with respect to similarity to a third pattern P_3. If the rating orders are not equal, an error counter is incremented. The weighting coefficients g_1- g_7 for the different features are set by minimizing this error value. The minimization is implemented by a systematic parameter search between $[0..10]$ with a step size of 0.5.

4.2 Learning Algorithm

A learning algorithm based on a gradient descent cannot be used because of the missing objective function. A statistical learning algorithm is not applicable due to the limited number of data. Thus, the learning algorithm consists of heuristic methods, such as evolutionary strategies, and the resulting evolutionary algorithm consists of operators, such as selection, recombination and mutation [8][9][10].

The main problem with using an evolutionary algorithm is to define the evolutionary operators. Theories which predicate the convergence of an algorithm (such as Schema theory, the Building-block theory or the Price theory) only exist for constraints like infinite optimization iterations and for toy problems. Therefore, it is impossible to calculate optimal evolutionary operators for tactile perception tuning and the evolutionary operators are optimized by means of the ATC.

A target pattern is introduced and the rating of the human is replaced by the ATC. Starting from initial patterns the learning algorithm and the ATC have to find target patterns. After a predefined number of optimization iterations the process is stopped and the convergence of the learning algorithm is rated. During this procedure different evolutionary algorithms can be compared and the evolutional operators can be optimized. To check the ATC a human has to compare the perception evoked by the found patterns of the evolutionary algorithm with the perception evoked by the target patterns.

5 Results

5.1 Automatic Tactile Classification

The minimized g_1 - g_7 coefficients are shown in table 2.

	g_1	g_2	g_3	g_4	g_5	g_6	g_7
Human a	1.0	4.5	1.0	0.0	0.0	2.0	1.5
Human b	0.5	4.5	2.0	0.0	0.5	0.5	4.5
Human c	2.5	5.0	0.0	0.0	1.0	5.0	2.0

Table 2. Minimized g_1-g_7 coefficients

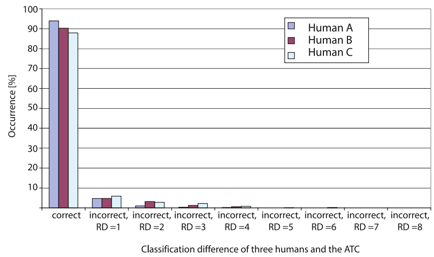


Fig. 2. Comparison of the human and the automatic tactile classification

The weighting coefficients g_1 - g_7 are different for each human, only g_4 equals zero. This means that the intensity of a tactile stimulus was not the crucial factor for the human tactile ranking.

Figure 2 shows the classification difference between three humans and the ATC. The humans were not familiar with the tactile device. The occurrence of correct (the tactile comparisons of the humans and ATC are matching) and incorrect classifications are represented on the y-axis. The incorrect classifications are subdivided into the rating difference (RD) between the two patterns P_1 and P_2 of the human, e.g RD = 1 - human rates P_1 and P_2 very similar, RD = 8 - human rates P_1 and P_2 not similar.

Figure 2 illustrates the results of 3 different humans. The human- and the ATC classification are the same in 88% (human c), 90% (human b) and 94% (human a).

Incorrect classifications are about 5% if RD equals 1 and about 1% up to 5% if RD equals 2. If RD is greater than 2 the incorrect classifications can be ignored.

This result shows that incorrect classifications are only observed if the human rates the tactile pattern very similar. Therefore, the tactile comparison of the ATC equals the tactile comparisons of the humans very well.

5.2 Tactile Pattern Generation

The learning algorithm has to generate tactile patterns that were not considered during the development of the system. Therefore, the evolutionary algorithm has to generate any possible solution for the tactile pattern. One way to obtain this is to start with an optimization algorithm only consisting of selection and mutation (continual improvement).

To study the effect of different selection and mutation mechanisms the following operators are used.

Stochastic Universal Sampling (SUS). The probability to select an individual is proportional to the fitness of the individual.

Rank-Based Selection (RS). A ranking of the individuals is executed and the three best individuals are marked. The probability to select one of the three individuals is the same. The selection probability does not depend on the absolute fitness value.

Mutation (M). The mutation is implemented by adding activities to and erasing activities from a tactile pattern. This contains the following actions:

- Increment, decrement and keep the number of time frames a(t). All operators are chosen with the same probability.
- Add or erase activities of actuators in N time frames. N is a random number but new activities must have a spatial neighborhood, e.g. activation of an actuator of a(t+1,p_i) must be next to activation of an actuator of a(t,p_i).

Spatial Mutation (SM). In addition to the first mutation M spatial filters like rotation, translation and time scaling are added and the amount of the spatial filters are randomly chosen. The order of the operators is randomly chosen, too.

Spatial Direction Mutation (SDM). SDM is a special variant of SM. The order of operators is no longer random but fixed. If a generation does not produce a better individual, the next operator is applied.

The convergence of the algorithm is measured by area A under the learning curve. The less is A, the better is the algorithm. A minimal statistic is made by repeating each run 10 times. A learning algorithm is evaluated for 15 different target patterns (tp) by $A_{algorithm} = \frac{1}{15}\frac{1}{10}\sum\limits_{tp=1}^{15}\sum\limits_{i=1}^{10} A_{i,tp}$.

Figure 3 shows the normalized $A_{algorithm}$ for different combinations of selection and mutation operators.

There are differences of 30% between the best and the worst algorithms. Comparing the two selection methods it can be observed that RS is better than the fitness proportional selection. Furthermore, all learning algorithms with spatial mutation have a better convergence than those without spatial mutation. The best algorithm consists of RS and SDM.

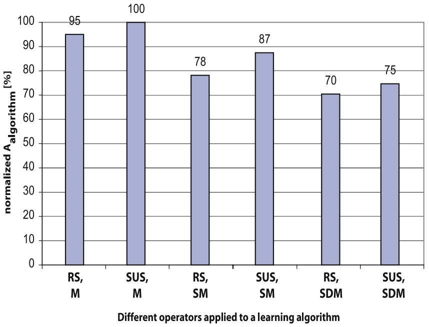


Fig. 3. Comparisons of different learning algorithm

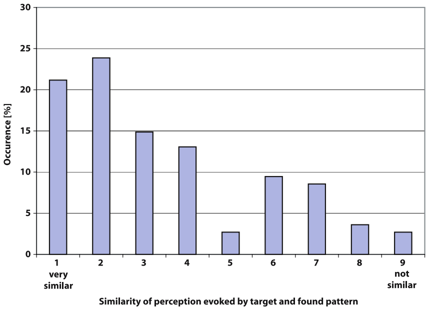


Fig. 4. Comparison of evoked perception of target and final patterns

5.3 Comparison of Final Patterns and Target Patterns

The learning algorithm and the ATC have found tactile patterns (final patterns) that are similar to the target pattern. Now it is analyzed if the evoked perception of the final patterns and the target patterns is also similar. If the ATC rates the patterns similar to the human, the final patterns and the tactile patterns should evoke the same perception. To prove this the human has to rate the final pattern and the target pattern with respect to similarity.

In figure 4 the results of the comparison of all three humans are added. The human has once again nine different classifications of similarity, 1-very similar and 9-not similar. The human rates about 20% as very similar and about 25% only one classification lower than very similar. For the classifications 3 to 9 the occurrence is constantly decreasing from 15% to 3%. Only classification number 5 varies with a low occurrence of 3%.

6 Discussion and Conclusion

- The evolutionary learning algorithm is successfully optimized in respect of the convergence of the learning algorithm (minimal iteration steps) and in respect of the tactile perception evoked by the final patterns (that are similar to the perception evoked by the target patterns).
- The ATC matches about 90% with the rating of three humans. These results are based on training data. The ability of generalization is proved by the tactile pattern tuning. During the tuning process many unknown tactile patterns are presented and have to be compared. The final tactile patterns are generated basing on these comparisons of unknown patterns. The similarity of the evoked perceptions of the final patterns and the perceptions of the target patterns shows the ability of generalization and that the human is successfully replaced by the ATC during optimization of the learning algorithm.
- A conspicuous result of comparing the final patterns to the target patterns is the low occupance of classification number 5. This could be caused by the fact that the

default position of the graphical user interface of the rating dialog is classification number 5. Starting from this position the human has to move the slider to the left (classification numbers 1-4) or to the right (classification numbers 6-9). Thus, the first classification strategy could be to rate similar patterns to the left and not similar patterns to the right. Afterwards, these rough classification is improved. The sum of the occurrence of classification numbers 1 to 4 is 73%, and of the classification numbers 6 to 9 it is 24%. Thus, the human rates 73% as similar and 24% as not similar.

References

1. C. Wilks, T. Schieder and R. Eckmiller, "Towards a Tactile Communication System with Dialog-based Tuning, " *Proceedings of the International Joint Conference on Neural Networks*, Portland, Oregon, pp. 1832-1837, 2003.
2. A. Pascual-Leone and R. Hamilton, "The Metamodal Organization of the Brain", C. Casanova and M. Ptito (Eds.), *Progress in Brain Research*, vol. 134, Amsterdam: Elsevier, pp. 1-19, 2001.
3. A. Baddeley, "The Fractionation of The Working Memory, " *Proceedings of the Natl. Acad. Sci. USA*, vol. 93, pp. 13468-13472, 1996.
4. J. A. Harris, I. M. Harris and E. Diamond, "The Topography of Tactile Working Memory," *The Journal of Neuroscience*, vol. 20(21), pp. 8262-8269, 2001.
5. C. Miles and H. Borthwick, "Tactile Short-term Memory Revisited," *MEMORY*, vol. 4(6), pp. 655-668, 1996.
6. P. Mahrer and C. Miles, "Recognition Memory for Tactile Sequences," *MEMORY* , vol. 10(1), pp. 7-20, 2002.
7. S. Salvador and P. Chan, "FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space, " *KDD Workshop on Mining Temporal and Sequential Data*, 2004.
8. D. Whitley, S. B. Rana, J. Dzubera and K. E. Mathias, "Evaluating Evolutionary Algorithms", *Artificial Intelligence* , vol. 85, no. 1-2, pp. 245-276, 1996
9. D. E. Goldberg, *The Design of Innovation*, Boston/Dordrecht/London: Kluwer Academic Publishers, 2002.
10. K. Sastrya and D. E. Goldberg, "Let's Get Ready to Rumble: Crossover Versus Mutation Head to Head" Tech. Rep. 2004005, Illinois Genetic Algorithms Laboratory (IlliGAL),1 2004

Self-organizing Map Initialization

Mohammed Attik[1,2], Laurent Bougrain[1], and Frédéric Alexandre[1]

[1] LORIA/INRIA-Lorraine, Campus Scientifique - BP 239 - 54506,
Vandœuvre-lès-Nancy Cedex - France
attik@loria.fr
http://www.loria.fr/~attik
[2] BRGM, 3 av Claude Guillemin - BP 6009 - 45060 Orléans Cedex 2 - France

Abstract. The solution obtained by Self-Organizing Map (SOM) strongly depends on the initial cluster centers. However, all existing SOM initialization methods do not guarantee to obtain a better minimal solution. Generally, we can group these methods in two classes: random initialization and data analysis based initialization classes. This work proposes an improvement of linear projection initialization method. This method belongs to the second initialization class. Instead of using regular rectangular grid our method combines a linear projection technique with irregular rectangular grid. By this way the distribution of results produced by the linear projection technique is considred. The experiments confirm that the proposed method gives better solutions compared to its original version.

1 Introduction

The Self-Organizing Map (SOM) is a widely used tool in exploratory data analysis for discovering groups and identifying interesting distributions in the underlying data. The most important characteristics of the SOM are the ability to extract topological structure hidden in data and the ability to visualize of complex data in a two or three dimensional display. As well as all the clustering methods, SOM converges to one numerous local minima. This technique is sensitive to initial conditions: number of clusters, initial cluster centers, number of iteration and neighborhood function. Learning speed and shapes of resultant clusters may differ with respect to these initial conditions [1].

In this work we interested only in cluster center initialization problem. In the literature two initialization approaches are found for clustering methods, namely supervised initialization approach and unsupervised initialization approach. The first one is a supervised selection which assumes that a small subset of samples data can be labeled in accordance with a tentative classification scheme. The second is generally preferred because of its better computational behavior, without the manual effort to label unlabeled subset of data [2].

We propose to group the unsupervised initialization methods for SOM in two classes: the *random initialization class* and the *data analysis based initialization class*:

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 357–362, 2005.

Random Initialization was found to be a preferable initialization approach for its simplicity. This approach follows a naive way to initialize the cluster centers. It is not necessarily the best approach, especially for producing a rapid convergence to a stable state [3,4,5,6]. It can be performed in a number of ways: (1) random component: each component of a reference vector is chosen randomly from the range of values observed in the data. (2) random selection: it uses sample data. It has the advantage that the samples automatically lie in the same part of the input space of data [7,8]. (3) random perturbation around mean values: it initializes the cluster centers by slightly perturbing the mean or the global centroid of the inputs.

In order to improve the SOM solution, multiple restarts (random initialization) can be used in the first step and in the second step. Morever, the model which present a better classification performance is selected [9,10]. We can also create the initialization diversities by using different initialization methods in the goal to select the better SOM solution.

Data Analysis Based Initialization uses some methods resulting from the statistical data analysis and data classification to initialize SOM. Several methods can be developed for clustering initialization. We can find in the literature two main methods for SOM initialization (1) for data classification: K-means based initialization involves three stages. In the first stage, we use the K-means algorithm to select $N \times M$ (i.e. the size of the feature map to be formed) cluster centers from a dataset. Then a heuristic assignment strategy is employed to organize the $N \times M$ selected data points into an $N \times M$ neural array so as to form an initial feature map. If the initial map is not good enough then it will be fine-tuned by the traditional Kohonen self-organizing feature map algorithm under a fast cooling regime in the third stage [11]. (2) for statistical data analysis: linear projection data based initialization combines a linear projection method with regular rectangular grid. The most known *linear projection method* is Principal Component Analysis (PCA). PCA is an unsupervised approach whose the perpose is to find the most salient features of a dataset. By an optimal linear transformation from a high dimensional space to a low dimensional subspace, PCA retains the most important relations of the sample data [12]. The clusters are initialized to lie in the same input space that is spanned by the first eigenvectors corresponding to the largest eigenvalues of the input data. The *reconstitution algorithm* is used to estimate the sample values corresponding to the empty cell of the regular grid [13].

Linear initialization method based on regular rectangular grid has defect that it does not take into account the distribution of linear projection results and also produces several empty cells which implies to use more reconstitution algorithm. *We propose to replace regular grid by irregular grid in order to improve the initialization by taking into account the distribution of linear projection results.*

Algorithm 1 LIG-Init

```
1- Project the data on the significant axis by using linear method;
2- Build the irregular rectangular grid by using algorithm (2);
3- For each cell containing data calculate its gravity center;
4- Estimate the gravity center for each empty cell by using equation (1).
```

2 Linear Method with Irregular Grids for Initialization

The proposed method is called *Linear method with Irregular Grids for Initialization (LIG-Init)*. It is a qualitative representation of geometric based on density distribution concept. It takes the form of a irregular grid partitioning the domain for each axis. We describe this method by Algorithm 1. The detail of the algorithm used by our method are detailed below:

- **Irregular Grid Algorithm.** To produce an irregular rectangular grid we propose an algorithm (Algorithm 2) based on the cumulative distribution function (cdf) called *CDF-Grid*. For each axis, this algorithm divides the cloud samples in an equal distribution. The intersections of the various axis give the *irregular rectangular grid*. We note that for a continuous function, the probability density function (pdf) which is the probability that the variate has the value x, can be expressed mathematically as: $\int_a^b f(x)dx = Pr[a \leq X \leq b]$

  and the cumulative distribution function (cdf) which describes the probability that a variate X takes on a value less than or equal to a value x, can be given by: $F(x) = \int_{-\infty}^{x} f(\mu)d\mu$.
- **Reconstitution Algorithm.** After linear projection on the rectangular grid, we can have empty cells i.e. cells without data. Reconstitution algorithm is used to estimate the sample values corresponding to this empty cell [14]. Suppose that $\mathbf{X}$ is represented by a cloud of n samples in $\mathbf{R}^p$. The best representation of the p variables in subspace of dimension q is generated by the orthonormal eigenvectors $v_1, v_2, \ldots, v_q$ of $\mathbf{XX}'$ associated with the eigenvalues $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_q$: (i) in $\mathbf{R}^p$, we have: $\mathbf{X}'\mathbf{X}\mu_\alpha = \lambda_\alpha \mu_\alpha$ and (ii) in $\mathbf{R}^n$: $\mathbf{XX}'\mathbf{v}_\alpha = \mu_\alpha \mathbf{v}_\alpha$. The transition formulas between two spaces, $\mathbf{R}^p$ and $\mathbf{R}^n$:

  $$\mathbf{u}_\alpha = \frac{1}{\sqrt{\lambda_\alpha}}\mathbf{X}'\mathbf{v}_\alpha \qquad \mathbf{v}_\alpha = \frac{1}{\sqrt{\lambda_\alpha}}\mathbf{X}\mathbf{u}_\alpha$$

  exact reconstitution is given by: $\mathbf{X} = \sum_{\alpha=1}^{p} \sqrt{\lambda_\alpha}\mathbf{v}_\alpha {\mathbf{u}'}_\alpha$
  and approximate reconstitution is given by:

  $$\mathbf{X} \simeq \mathbf{X}^* = \sum_{\alpha=1}^{q} \sqrt{\lambda_\alpha}\mathbf{v}_\alpha {\mathbf{u}'}_\alpha \tag{1}$$

Algorithm 2 CDF-Grid

```
1- Choose the number of clusters for each axis;
2- Build density distribution function for each axis;
3- Build the cumulative density distribution function for each density function;
4- Build the grid by applying the intersection enters the various axis.
```

3 Evaluation of Cluster Center Initialization

Evaluating the quality of SOM solution is a non-trivial and often ill-posed task. Generally, two approach are used for SOM solution evaluation:

- **Quality Measures Based on the Distance:** The classical evaluation measures for the quality of a classification are based on the intra-class inertia and the inter-class inertia [15]:
  - intra-class inertia, measures the distance between the closest members of the clusters:
  $$\frac{1}{|C|}\sum_{c\in C}\frac{1}{|c|}\sum_{d\in c}\|p_c - p_d\|^2 \tag{2}$$
  - inter-class inertia, measures the distance between the centroids of the clusters:
  $$\frac{1}{|C|^2 - |C|}\sum_{c_i\in C}\sum_{c_j\in C, c_j\neq c_i}\|p_{c_i} - p_{c_j}\|^2 \tag{3}$$

  ($|.|$ is the counting measure and $\|.\|$ is the Euclidean distance) where C represents the set of classes associated to the classification, d represents a class member and p_x represents the profile vector (center) associated to the element x.

  Thanks to these two measures, a clustering is considered as good if it possesses low intra-class inertia as compared to its inter-class inertia. However, these measures are often biased in several ways as well as, the sensitivity to the empty classes, that these last are regarded as intermediary classes by SOM.
- **Visualization Evaluation:** The most characteristic property of the SOM solution is the preservation of topology. The goal of visualization is verifying this characteristic. Generally, U-matrix and Sammon mapping are used for visualization. The Unified distance matrix (U-matrix) makes the 2D visualization of multi-variate data possible using SOM's code-vectors as data source. By U-matrix we can detect topological relations among neurons and infer about the input data structure. High values in the U-matrix represents a frontier region between clusters, and low values represent a high degree of similarities among neurons on that region [16]. Sammon mapping computes

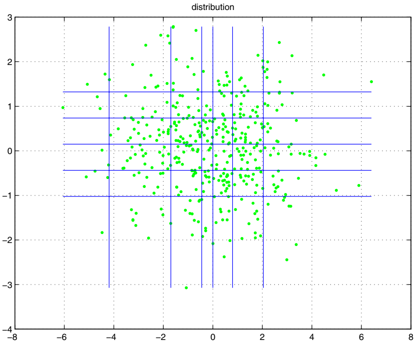


Fig. 1. Regular (black) and irregular (blue) grid for initialization

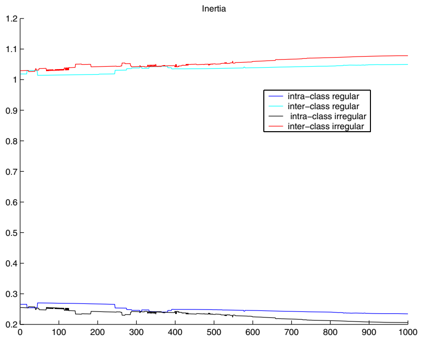


Fig. 2. Evaluation of Initialization

the location of neurons in two or three-dimensional space from their connection weights. It is useful to use this mapping for assisting the interpretation of the 2-D or 3-D distance map [17].

The figures (Fig. 1) and (Fig. 2) represent the comparison between regular and irregular grids for clusters initialization by using an artificial dataset generated randomly. For the same number of cells, irregular grid represents well the data distribution and shows that the irregular grid is clearly better than the regular grid for performance evaluation.

4 Conclusion

We presented in this work different initialization methods associated with the SOM cluster centers. We also discussed some difficulties that arise in the evaluation SOM solution. We shown how the initialization with repetition method based on diversity in different random initialization methods can be used to find the good solution. The model search involves a considerable increase in the computational cost, but in serious data mining, the major concern is the reliability of the results for making decision. We proposed an approach based on linear projection and irregular grid which takes into account the distribution of results produced by the linear projection method compared to using regular grid. It outputs a quasi-null number of empty cells. Accordingly, it gives a “good” initialization for SOM. Moreover, it deals with any dimensionality size of SOM, *i.e.* 2D, 3D, etc.

Acknowledgment

We would like to thank Abdessamad Imine, Jean Charles Lamirel and Shadi AL Shehabi for their helpful remarks.

References

1. Kohonen, T.: The Self-Organizing Map. Proceedings of the IEEE **78** (1990) 1464–1480
2. Milligan, G.W.: An examination of the effect of six types of error perturbation on fifteen clustering algorithms. Psychometrica **45** (1980) 325–341
3. Bradley, P.S., Fayyad, U.M.: Refining initial points for K-Means clustering. In: Proc. 15th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA (1998) 91–99 citeseer.ist.psu.edu/bradley98refining.html.
4. Pena, J.M., Lozano, J.A., Larranaga, P.: An empirical comparison of four initialization methods for the k-means algorithm. Pattern Recogn. Lett. **20** (1999) 1027–1040
5. Juan, A., Vidal, E.: Comparison of Four Initialization Techniques for the K-Medians Clustering Algorithm. In: Proc. of Joint IAPR Int. Workshops SSPR 2000 and SPR 2000. Volume 1876 of Lecture Notes in Computer Science., Alacant (Spain), Springer-Verlag (2000) 842–852
6. He, J., Lan, M., Tan, C.L., Sung, S.Y., Low, H.B.: Initialization of cluster refinement algorithms: A review and comparative study. In: Proceedings of International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary (2004) To appear.
7. Forgy, E.W.: Cluster analysis of multivariate data: efficiency vs interpretability of classifications. Biom **21** (1965) 768–769
8. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In Cam, L.M.L., Neyman, J., eds.: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Volume 1., Berkeley, CA, University of California Press (1967) 281–297
9. Strehl, A., Ghosh, J.: Cluster ensembles – a knowledge reuse framework for combining partitionings. In: Proceedings of AAAI 2002, Edmonton, Canada, AAAI (2002) 93–98
10. Ghosh, J.: Multiclassifier systems: Back to the future. In: MCS '02: Proceedings of the Third International Workshop on Multiple Classifier Systems, Springer-Verlag (2002) 1–15
11. Mu-Chun Su, T.K.L., Chang, H.T.: Improving the self-organizing feature map algorithm using an efficient initialization scheme. Tamkang Journal of Science and Engineering **5** (2002) 35–48
12. Ben-Hur, A., Guyon, I.: Detecting stable clusters using principal component analysis (2003) M. J. Brownstein and A. Kohodursky (eds.) Humana press.
13. Elemento, O.: Apport de l'analyse en composantes principales pour l'initialisation et la validation de cartes de kohonen. In: Septième journées de la Société Francophone de classification, Nancy, INRIA (1999)
14. Lebart, L., Morineau, A., Piron, M.: Statistique exploratoire multidimensionnelle. Dunod (1995)
15. Lamirel, J.C., Al Shehabi, S., Francois, C., Hoffmann, M.: New classification quality estimators for analysis of documentary information : application to patent analysis and web mapping. Scientometrics **60** (2004) 445–462
16. Ultsch, A.: Self-organizing neural networks for visualization and classification. In Opitz, O., Lausen, B., Klar, R., eds.: Information and Classification, Berlin, Springer (1993) 307–313
17. Jr., J.W.: A nonlinear mapping for data structure analysis. IEEE Transactions on Computers **C-18** (1969) 401–409

Principles of Employing a Self-organizing Map as a Frequent Itemset Miner

Vicente O. Baez-Monroy* and Simon O'Keefe

Computer Science Department,
University of York,
York YO10 5DD, United Kingdom
vicente(sok)@cs.york.ac.uk

Abstract. This work proposes a theoretical guideline in the specific area of Frequent Itemset Mining (FIM). It supports the hypothesis that the use of neural network technology for the problem of Association Rule Mining (ARM) is feasible, especially for the task of generating frequent itemsets and its variants (e.g. Maximal and closed). We define some characteristics which any neural network must have if we would want to employ it for the task of FIM. Principally, we interpret the results of experimenting with a Self-Organizing Map (SOM) for this specific data mining technique.

1 Introduction

Association Rule Mining (ARM), introduced by Agrawal et. al. [11], may be divided into two-subprocess activities: *a) finding frequent itemsets* and *b) the generation of association rules* to tackle the problem of discovering significant association rules among the items (attributes, components, etc.) of a collection of data (e.g. Basket shopping data) which must satisfy user constraints (e.g. Support and confidence thresholds). During the last decade, the first step of ARM, better known as Frequent Itemset Mining (FIM) [4], has been the motivation for the development of a large number of algorithms [5], of which the most frequently cited and better-known is Apriori [1]. The majority of the proposals are based on the use of diverse data structures (e.g. Hash tables, trees) and the exploitation of the Apriori anti-monotone property: *if any k pattern is not frequent in the database, its extensions, (k+1) super-patterns can never be frequent* to avoid the complexity of the problem.

At the same time, the technology of neural networks, which has been successfully employed for tasks of pattern recognition [2], already has been introduced for data mining problems such as classification and prediction [3,10], but it has not been applied seriously to problems such as FIM. This work analyzes the use of neural network technology for the problem of association rule mining, especially for the task of generating frequent itemsets. In particular, this work

* Supported by The Central Bank of Mexico (BANXICO) and The National Council of Science and Technology (CONACyT), Mexico, under grant No. 161407.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 363–370, 2005.

explores how a self-organizing map could be employed for FIM. The remainder of this paper is organized as follows: Section 2 describes the problem of Frequent Itemset Mining (FIM). Section 3 defines the minimal characteristics that a neural network must have if we want to use it for FIM. In Section 4, we present an analysis of the experiments using SOM for FIM. Some conclusions are presented in Section 5. We decide not to include information about Self-Organizing Maps (SOM) for space limitations but details of this technology can be found in [8].

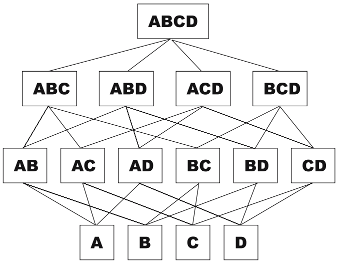


Fig. 1. Lattice representing the data space formed by a 4-itemset and its subsets

2 Problem Definition: Frequent Itemset Mining (FIM)

Let Γ be a set of items. An *itemset* I is any subset of items such as $I \subseteq \Gamma$. Let D be a set of transactions T. A transaction T is defined by the couple (tid, I) where tid is an unique transaction identifier and I is its corresponding itemset. A transaction $T = (tid, I)$ is said to support an itemset X only if $X \subseteq I$.

An itemset X is said to be *frequent* if its *support* s is greater than or equal to a given minimum threshold σ. The support value of an itemset can be defined by the number of elements (transactions) in the set $S = \{tid \mid (tid, I) \in D, X \subseteq I \}$. In other words, this support value, $s = \frac{\|S\|}{\|D\|}$, is the probability that X appears among the possible transactions contained in D.

The goal then is to mine a given database D in order to identify all the frequent itemsets (FI) based on a defined minimal support threshold σ. The complexity of this problem is the size of the search space (data space) that can be formed by a set of items Γ. This mining activity simply becomes unfeasible when the number of items grows to be large. For instance, a frequent k-itemset (k is the size) implies the presence of $2^k - 1$ frequent itemsets (all of them are combinations or subsets of the k-itemset). In figure 1 we show a lattice describing the search space formed by a 4-itemset in order to have a better idea of this.

3 A Hypothetical Neural Network for FIM

Neural networks have been used successfully for many different pattern recognition problems. If we consider the similarity between an itemset and a pattern definition, then the way in which neural networks can be applied becomes clear. FIM may be seen as the task of detecting patterns (itemsets) that occur with certain frequency in some dataset and that have to be identified and counted by some neural-network model. We have named this activity pattern counting.

Although the range of FIM algorithms in the literature is large, there are few works directly related to the hypothesis of employing a neural network for ARM. The work presented in [7] introduces neural network technology to the problem of ARM. In this work, the authors promote the use of an $n\ x\ p$ Hopfield network to discover frequent itemsets based on the premise that this neural model has been used previously for combinatorial optimization problems. Secondly, the use of SOM for this specific data mining problem was proposed in [14] but no analysis was presented and its authors only make an assumption that a SOM can be employed for FIM because of its clustering feature.

The characteristics of the current FIM algorithms, the definition of the problem itself and the lack of formal work on the use of neural networks for FIM, lead us to search for a model in the domain of the neural network technology that can fulfill the requirements to became a frequent itemset miner. We propose that a neural network must have the following characteristics to tackle this problem:

- A neural network, which supports unsupervised training must be considered initially for this data mining task because we do not know the number of different patterns (itemsets) than can be found in a dataset.
- The weight matrix obtained as a result of the training of the neural network must contain information from which we can obtain a reasonable support metric for any pattern (itemset) in our original data space.
- It is essential that this neural model deals with n-dimensional patterns with large n, particularly with data that could be formatted as binary arrays.
- This neural-network model must be able to learn new patterns without forgetting past knowledge. This characteristic will help to keep the support of the patterns updated and avoid using tree-party processes which are employed normally for this purpose [9].
- It will suggest which parts of the data lattice are important to mine depending on the threshold provided by the user.
- The size of the weight matrix formed must be much smaller than the size of the training dataset. This characteristic will allow us to run different mining exercises (e.g. Setting different support threshold values) on the neural network without having to rebuild or retrain it.

Employing neural networks gives the advantage of reusing the weight matrix for other mining reasons such as clustering or classification. In addition, the portability of the weight matrix makes the use of neural networks for distributed-data-mining processes realizable. Based on the number of requirements fulfilled and on the features that SOM has to form maps of clusters representing the data distribution of a dataset, the SOM is a suitable network for further investigation.

4 Experimentation and Analysis

In this section, we present the results obtained from training a SOM for FIM. For all these experiments, a batch-training-mode SOM was employed from the toolkit developed by [12,13]. The default training parameters[1] were used. Six synthetic binary datasets[2] were also created. It is important to state that even if we could have used real datasets for this experimentation, we decide not to employ them, because they normally do not include every possible combination of their items and the aim of this first experimentation is only to analyze how a SOM behaves when the distribution of the data is being controlled (e.g. One of our datasets was formed to represent all patterns that form a 4-itemset lattice as in Figure 1). Thus, we initially train the SOM using the synthetic datasets in order to identify whether there are any characteristics on the maps, formed by a training process with binary-input patterns, that could be used to discover either frequent, maximal or closed patterns. The maps shown in Figure 2 are the results of the SOM training process from which we make the following observations:

Formation of big clusters. In every case, the resulting map is composed of the formation of four large clusters which model the distribution of the patterns contained in each dataset created for this analysis. The presence of these four clusters can also be verified by the visualization of their corresponding U-matrix in which the absence of colour describes a group of nodes and the existence of colour defines a division between these groups. This U-matrix is the result of calculating the distance (e.g., Euclidean distance) among all the nodes in the map. The size of the maps and clusters depends directly on the number and definition of input patterns contained in the datases. Although parameters such as the radius defined for the process can affect the resulting map, the four-cluster formation observed in all cases will be always present due to the fact that the two-dimensional representation (compressing representation) of our input data (binary vectors) can be described by data points arranged closely to the four vertexes of a square figure.

Identification of winners and hit histograms. A competitive process is executed by all the nodes in the map during the training that controls the manner in which the modifications of the map will be made. This process,

[1] In a batch training mode, parameters such as the neighborhood function and the size of the radius have to be mainly defined. In this particular case, we have employed the Gaussian function to modify the neighborhood and set the radius to one for the entire process. More details on the variations of these parameters can be found in [12,13].

[2] Our synthetic data consists of six datasets which contain a)16, b)64, c)256, d)1024, e)4096 and d)65536 transactions respectively. Within each dataset, the transactions correspond to the all possible combinations of a)4, b)6, c)8, d)10, e)12 and d)16 items (attributes, columns). Thus, the total number of transactions is equal to 2^k, where k is the number of items used to form them. Binary vectors represent the content of each dataset.

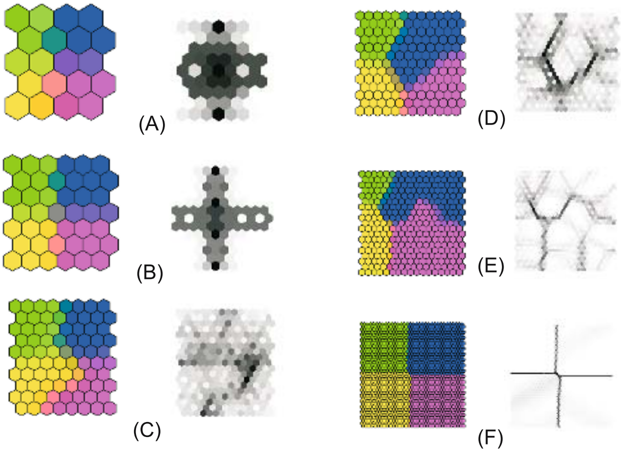


Fig. 2. Maps formed after training a SOM with all the possible combinations of: (a 4, (b 6, (c 8, (d 10, (e 12 and (f 16 items. In each column of maps, the maps on the right group their nodes based on the similarities of their components in which colours are used to visualize the formation of four clusters. The maps on the left (black and white maps) illustrate the corresponding U-Matrix which is formed from the results of calculating the distance metric among all the nodes of a map. The absence of colour in these maps means that the distance among the corresponding nodes is minimum. The dark sections define a division among either the clusters of the map or the bmus of a cluster. More techniques on how to understand the clusters formed by a SOM can be found in [6]. It is important to point out that all these maps are formed with a constant neighborhood size equal to one. Even if a modification of this parameter (neighborhood size or radius) can give different sizes of the clusters as a result, the formation of four clusters will always be present in the resulting map.

better known as the Best Matching Unit (BMU) selection, leads the training because it determines which nodes will be turned into winners. The components of each winner and its neighbourhood will be updated in order to approximate their values to those that form part of all input patterns that hit this node. With this feature, the SOM splits the input data universe into several winners that form clusters. A winner has the characteristic that it is chosen as a BMU by some input patterns during the training. This information can be seen by using hit histograms as in Figure 3, which represent the number of times that a node was chosen by input patterns. It is important to mention that this feature in the SOM resembles the support metric for an itemset defined for the problem of FIM.

Dependency among clusters (The grouping of input patterns). Inside of each of the winners an interesting characteristic occurs.This property refers to the existence of a dependency among all the input patterns that have chosen the same node as a winner. This dependency can be seen if a

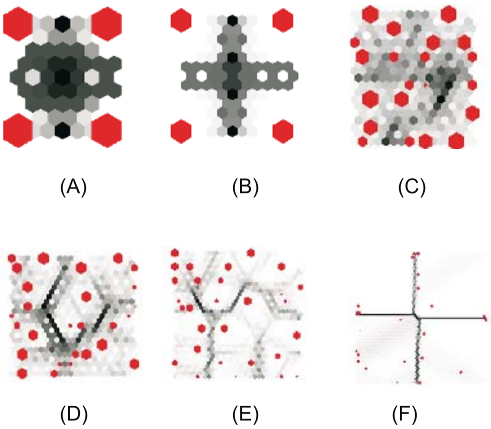


Fig. 3. Visualization of hit histograms on the U-Matrix for the detection of winners for the datasets containing: (a 4, (b 6, (c 8, (d 10, (e 12 and (f 16 items. The size of each red hexagon represents the total number of times that a node was hit by the input patterns. The number of hits among the winners can be different since the grouping of the input patterns is done by calculating the similarities among them by using a distance metric. Any node chosen by the input patterns is known as a Best Matching Unit(BMU) or winner.

tree data structure is built using the patterns accumulated in the winner with the following rules:

Let A and B be two binary patterns in the winner node X and $\vee$ be an OR bitwise operator such that the operation $A \vee B$ can give us the following dependencies:

- They are related directly only if $(A \vee B)$ gives either A or B as a result such that A is considered the parent of B, $(A \vee B) = A$, defining that $B \subseteq A$, otherwise A will be considered the child of B, $(A \vee B) = B$, defining that $A \subseteq B$.
- A partial or null dependency is defined when $(A \vee B)$ gives neither A nor B as a result, defining that a possible dependency among the clusters on the map exists.

Figure 4(b), shows a basic example of the hierarchical tree built by patterns of the dataset formed with 4 items that have hit the same winner within the cluster $C1$. A more complex structure can be formed if all of the elements from all of the clusters are considered to give a lattice such as the one illustrated in Figure 4(c). An important conclusion of this task is the relevant manner in which the SOM splits the data space so that a novel manner of traversing the lattice can be defined by this method. As a first thought, it could help to determine which group of patterns could be considered interesting when a support threshold is provided. This decision can

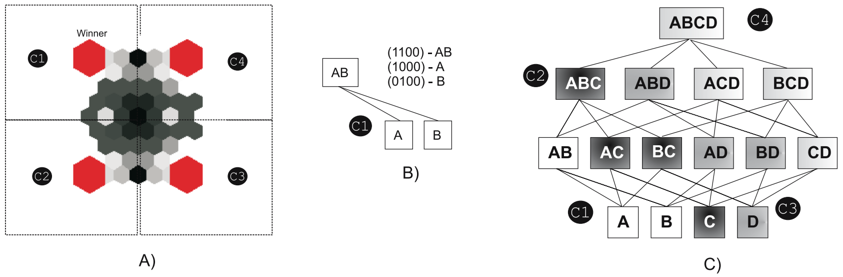


Fig. 4. Visualization of the dependency among the four clusters formed by training a SOM using the dataset containing all itemsets derived from a 4-itemset. Figure A illustrates the formation of the clusters with their winners. Each winner contains more than one input pattern. A basic hierarchical tree is shown in Figure B which has been formed with the input patters accumulated by the winner in the cluster $C1$ for instance. In figure C, a lattice is built using the distribution of the input patterns in the map. This lattice describes the dependency existed among the clusters. Four colours are used in the lattice to illustrate which input patterns share or belong to the same winner (The dependency among the clusters).

be made if a record of the support value of each itemset is stored and a modified version of the anti-monotone property of Apriori is employed such as: *If cluster is frequent, then some of its members called winners could be frequent also, depending on their hit histogram values.*Using this method, we will be identifying those sectors of the data space which are not relevant for our mining process.

5 Conclusions

This work outlines the use of neural network technology for the problem of association rule mining, especially for the task of generating frequent itemsets. In particular, this work explores how a self-organizing map could be employed for frequent itemset mining. This work reinforces the hypothesis that the use of neural network technology for the problem of association rule mining is feasible. Analysis of our experimentation with synthetic datasets has shown that a self-organizing map may become a frequent set miner. We also show a novel way of traversing the data space formed for this type of data ming task could be also implemented if we consider the dependency among the patterns accumulated in each winner node on the map.

References

1. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

2. Christopher M. Bishop. *Neural Networks for Pattern Recogntion.* Oxford University Press, 1995.
3. Mark W. Craven and Jude W. Shavlik. Using neural networks for data mining. *Future Generation Computer Systems*, 13(2–3):211–229, November 1997.
4. B. Goethals and M. Zaki. Fimi'03: Workshop on frequent itemset mining implementations, 2003.
5. B. Goethals and M. J. Zaki. Advances in frequent itemset mining implementations: Report on fimi'03, 2003.
6. M. Sulkava J. Vesanto. Distance matrix based clustering of the self-organizing map. *Lecture Notes in Computer Science, ICANN 2002*, 2415:951–956, 2002.
7. T. El-Ghazawi LIAL Ecole Centrale de Lille France K. Gaber, M.J. Bahi. Parallel mining of association rules with a hopfield type neural network. In *12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'00)*, pages 90–93. IEE, 2000.
8. Teuvo Kohonen. *Self-Organizing Maps.* Springer-Verlag, Berlin, 1995.
9. Sau Dan Lee and David Wai-Lok Cheung. Maintenance of discovered association rules: When to update? In *Research Issues on Data Mining and Knowledge Discovery*, 1997.
10. Hongjun Lu, Rudy Setiono, and Huan Liu. Effective data mining using neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):957–961, December 1996.
11. T. Imielinski R. Agrawal and A. N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., 26–28 1993.
12. J. Vesanto. Data mining techniques based on the self-organizing map. Master, Helsinki University of Technology, May 1997.
13. J. Vesanto. Som-based data visualization methods. *Intelligent Data Analysis*, 3(2):111–126, August 1999.
14. S. Yang and Y. Zhang. Self-organizing feature map based data mining. In *International symposium on Neural Networks (ISNN 2004), LNCS 3173*, pages 193–198. Springer-Verlag, 2004.

Spatio-Temporal Organization Map: A Speech Recognition Application

Zouhour Neji Ben Salem, Feriel Mouria-beji, and Farouk Kamoun

Cristal Laboratory: Artificial Intelligence Unit, National School of Computer Sciences,
University Campus of Manouba, Tunisia
{zouhourbensalem, Farouk.kamoun}@hotmail.com,
Feriel.beji@ensi.rnu.tn

Abstract. The temporal dimension is very important to be considered in many cognitive tasks involving a decision making or a behavior in response to spatio-temporal stimuli, such as vision, speech and signal processing. Thus, the capacity of encoding, recognizing, and recalling spatio-temporal patterns is one of the most crucial features of any intelligent system either artificial or biologic. If some connexionnist or hybrid model integrates the temporal data as spatial input, few other models take them into account together internally either in training or in architecture. Temporal Organization Map TOM is one of the latest types. In this paper, we propose a model gathering saptio-temporal data coding, representation and processing based on TOM map, and yielding to a Spatio-Temporel Organization Map (STOM). For spatio-temporal data coding, we use the domain of complex numbers to represent the two dimensions together. STOM architecture is the same as TOM, however, training is ensured by the spatio-temporal Kohonen algorithm to make it able to manage complex input.

1 Introduction

Most connectionists or even hybrid models process spatial and temporal information within two methods: 1- **Extrinsic:** temporal information is coded with spatial one at the level of inputs. Here we have three cases: either time is introduced to the model as spatial data and this is used in serial-parallel transformation [18] and TDNN [6, 12], or spatial information is encoded in temporal form, we found this in spiking neuron models [7], or the two dimensions are grouped together in a spatio-temporal coding approach, which is the case of STAN model[11] 2- **Intrinsic:** models take into account internally the two dimensions at the same time. Here also we have two cases: either temporal dimension is supported by the whole architecture of the model, yet the training remains spatial. This case enters in the category of recurrent network [16], or they are embedded at the basic unit of the model, which is represented by TOM map [5]. TOM map could be seen as an interesting alternative for spatio-temporal representation facing models of classical inspiration like ANN or hybrid one. In fact, TOM offers a better understanding of the self-organization map and the structure of cortical signal, because its design and functioning are inspired from cortical column and area [1, 2] of the human cortex. So within a biological frame work, TOM reveals how dynamic connectionist networks can self-organize to embed spatial signal and their temporal context in order to realize a meaningful representation of dynamic

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 371 – 378, 2005.

phenomena. While TOM encodes internally time and space, training processes separately these two dimensions. Indeed, it is done on two steps: first step for fixing the spatial masks using a self-organization algorithm. The second step for training the temporal occurrence (context) of learned masks with a specific temporal algorithm. Thus, temporal training is realized when the spatial one is accomplished. Therefore, input of different learning steps does not interact spatio-temporally inside the map yielding a passive input representation. At this level, several questions could be raised: is it possible to learn spatial and temporal dimension at the same time? Or at least, could temporal learning put back in cause or refine certain aspects of spatial learning? We think that with the integration of spatio-temporal training, one will certainly have more significant learning of spatio-temporal events. As consequence of training, TOM inputs are given to the map sequentially, the spatial then the temporal data. With spatio-temporal training, STOM inputs embed space and time together. This fact is done by the spatio-temporal coding approach presented by Gilles Vaucher [11]. This approach is proposed to introduce a spatio-temporal coding for the STAN. It is based on complex numbers because they are the only numbers offering two degree of freedom encoding the two dimensions. Using this approach and with a spatio-temporal training, STOM will be an extrinsic and intrinsic model at the same time. STOM is tested in speech recognition for two aims. First, to see if the map can correctly conserve the topology of the human auditory cortex [9, 10]. Thereby, high frequency signals are neighborly encoded with respect to their time occurrence; the same phenomena must be observed for low frequency signals. Second, to see the performance of STOM regarding to TOM, since TOM was applied to this application.

2 The Spatio-Temporal Coding Approach

The big amount of information that stochastic signal transports, is enclosed in a space of representation (amplitude x time).

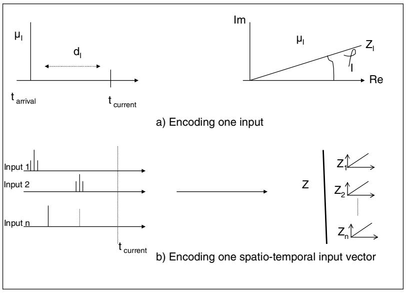


Fig. 1. The Spatio-Temporal coding approach

The correlation between time and space at the level of input data could be represented by the spatio-temporal coding approach proposed by Gilles Vaucher. This ST approach takes its roots from the work undertaken by neurobiologists to model passive electric properties of the dendrites trees [17]. The spatio-temporal coding approach (Fig.1.) was introduced for the aim to provide the classical artificial neurons the capacity of processing sequences in asynchronous manner, leading to the emergence of STANN (Spatio Temporal Artificial Neural Networks) [14, 15]. It consists on adding the delay; at the level of input; to introduce temporal information. It improves the work of 'Integrate and Fire' neuron [4] using the field of complex numbers. The spatio-temporal coding is done as following: Each input produces a train of impulses, for each impulse or spike I characterized by its amplitude μ_I and the temporal delay d_I which separates the current instant of time from the time at which the impulse has been occurred, a complex number z_I is assigned. z_I Contains μ_I as its module and ρ_I as its phase as following:

$$\begin{aligned} z_I &= \mu_I e^{i\rho_I} \\ \rho_I &= \arctan(\ \mu_t d_I\ ) \end{aligned} \tag{1}$$

To incorporate the concept of ‘Integrate and Fire’ neuron into the approach, the amplitude of one input is attenuated with the passage of time. This attenuation is measured between $t_{current}$ and $t_{arrival}$ as following:

$$\mu_I = \mu_{I_0} e^{-\mu_s d_I} \tag{2}$$

Where μ_{I_0} is the initial amplitude of the input, μ_s, μ_t are two constants allowing the control of the characteristics of the model.

3 The Spatio-Temporal Organization Map

3.1 TOM Architecture

From the technical point of view, TOM is a cortical map[3] whose goal is to learn special signs and their temporal sequences in asynchronous manner. TOM is composed of Super Unit (SU) (Fig. 2) having an invariant architecture, different kind of inputs and different small interconnected units for the activity propagation through the map. Each SU has three kinds of connections. First, feedforword connections determine how spatial stimuli without additional temporal relation are represented by SU. Second, lateral or local connections determine the spatial topology of the map and represent the neighbored function. Third, temporal or intra-map connections link small units inside the SU. Unit activation depends on if the suitable stimulus of the SU is presented and if its contextual activity is correct, that is at least one unit is activated in its Receptor Field (RF). The later is represented by the intra-map links. The connections between different units of different SU define the temporal topology inside the map and constitute its Short Term Memory. This STM is ensured by a decreasing activation of units, while the SU has a binary activation. In fact, if the input corresponds to the stimuli coded by the SU, and if its temporal context is right (one unit is activated in the RF), then the corresponding unit is activated and the SU ‘fire’.

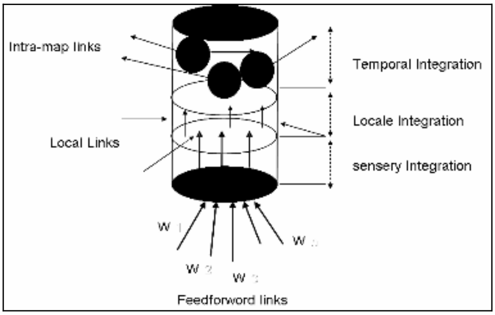


Fig. 2. The TOM Super Unit

3.2 STOM Architecture

STOM has the same architecture as TOM, however, the SU (Fig. 3) will receive spatio-temporal input coded using the approach cited in section 2. Therefore, to represent spatio-temporal data inside SU two cases could be mentioned depending on the application used.

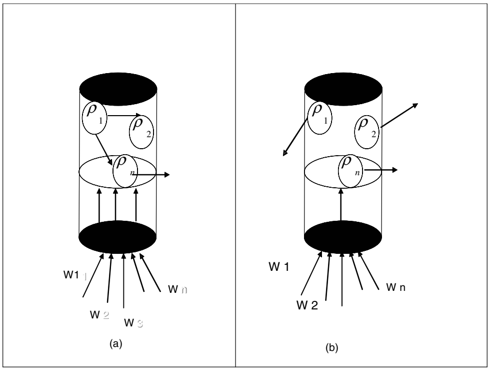


Fig. 3. STOM Super Unit

1st case: The SU code any input equals or near to the amplitude w_1 at the phase φ_1, w_2 at the phase φ_2,..., w_n at the phase φ_n (figure 2 a). Therefore, the weighting vector is:

$$W = \begin{pmatrix} w_1 e^{-i \quad {}_1} \\ w_2 e^{-i \quad {}_2} \\ .... \\ w_n e^{-i \quad {}_n} \end{pmatrix} \tag{3}$$

2nd case: The SU code a known spatial representation w_1, w_2,.., w_n which could occur at the phase φ_1, φ_2,..., φ_n. The weighting vector is, therefore:

$$W = \begin{pmatrix} w_1 e^{-i \quad {}_j} \\ w_2 e^{-i \quad {}_j} \\ \cdots\cdot \\ w_n e^{-i \quad {}_j} \end{pmatrix} \mathit{for}\ j = 1,..,n \tag{4}$$

The function and activation of each STOM SU remain identical to those of TOM SU.

4 Spatio-Temporal Training

TOM training is done in two steps and works with the decision 'Winner takes all' and Kohonen algorithm. Applying the spatio-temporal complex coding approach to entries, we will have input vectors in C^n, thus we need to extend the Kohonen algorithm to deal with complex numbers. We use for that, the Spatio-temporal Kohonen proposed by Mozayyani [13]. ST-Kohonen works in the same manner as Kohonen; however, the winner is chosen according to the hermitienne distance instead of the Euclidian one:

$$\delta(X, W_i) = \|X - W_i\| = \sqrt{{}^t(X - W_i)\overline{(X - W_i)}} \tag{5}$$

The adaptation rule for ST-Kohonen is the same as the one presented in Kohonen, yet we manage complex numbers instead of reels. With the application of ST-Kohonen to STOM, we will update three types of connections. Feedforward and lateral connections are updated using the same rule as kohonen algorithm. The neighborhood function is fixed as prior. The ntra-map connections are binary connections used to create and update the RF of one unit. They are modified using the same algorithm proposed for TOM [5]. Different approaches have been developed to embed spatio-Temporal information in the SOM algorithm: The most direct way is to include time-delayed version of the input vectors in the input layer or to add a second layer of processing neurons that captures the spatial dynamics of the first layer [21]. Other research has focused on Dynamic Leaky Integrator, first on the level of processing neurons [22], later on the level of input vectors [23]. These works address the problem of spatio-temporal pattern classification: neurons code whole sequences of stimuli and allow distinguishing the learned sequence. The ambiguity of a presented spatial pattern is resolved by incorporating network memory representing preceding spatial patterns. Contrarily, the aim of STOM is to be an intrinsic and extrinsic model that is to embed Spatio-Temporal relation at the level of inputs, architecture and processing. This aim permits the model to generate first spatio-temporal feedforward weight vectors that capture the essential invariances and serve as reasonable templates for further stimulus and second a more meaningful spatio-temporal representation of these templates. Another spatio-temporal algorithm is available. The training presented by Wiemer [8] is an algorithm that extends the common self-organization map (SOM) from the processing of purely spatial signals to the processing of purely spatio-temporal signals. The algorithm aims at transforming the temporal interval between two consecutive stimuli into distance representation. This transformation is done at the level of processing and not inputs. Thus, time is not conserved and treated as a dimension a part but transformed into spatial distance, which does not match STOM architecture and coding approach.

5 Experimentation and Results

For reasons mentioned in the introduction, the application used in this paper is the speech recognition. The map is trained and tested with isolated words of TI-Digit database. The number of speakers contributing to the application is 110 distributed as 55 men and 55 women pronouncing the ten digits and the word 'oh'. We use the Mel Frequency Cesptral Coefficients (MFCC) vectors [20] to represent speech signal. To permit the spatio-temporal coding, we suppose that all coefficients for one MFCC vector have the same time of occurrence. Thus, complex MFCC vectors will have the following form:

$$MFCC_{comp} = \left\{ \begin{matrix} c_1 e^{- \; j} \\ c_2 e^{- \; j} \\ .... \\ c_{12} e^{- \; j} \end{matrix} \right\} \tag{5}$$

The proposed speech recognition model is composed of one STOM map connected to 11 neurons representing the digits to be recognized. The map contains 30 (5x6) spatio-temporal super units. μ_s, μ_t are both chosen inverse to the temporal window according to Baig [15]. To compare TOM and STOM we have used a ST-Kohonen algorithm with an STOM SU architecture shown by the figure 3 b), because and as mentioned above, we suppose that all coefficients of one MFCC vector have the same time of occurrence. The result is shown in the following table:

Table 1. Digits Recognition

	0	1	2	3	4	5	6	7	8	9	oh
0	106	0	0	0	0	0	0	0	0	0	4
1	0	110	0	0	0	0	0	0	0	0	0
2	0	0	110	0	0	0	0	0	0	0	0
3	0	0	0	110	0	0	0	0	0	0	0
4	0	0	0	0	108	2	0	0	0	0	0
5	0	0	0	0	0	110	0	0	0	0	0
6	0	0	0	0	0	0	110	0	0	0	0
7	0	0	0	0	0	0	1	109	0	0	0
8	0	0	0	0	0	0	0	0	110	0	0
9	0	0	0	0	0	0	0	0	0	110	0
oh	7	0	0	0	0	0	0	0	0	0	103

As shown in the above table, the recognition rate reaches the level 100% except for the 0 (resp 4, 7 and ‘oh’) where 4 (resp 2, 1 and 7) cases were confused to ‘oh’ (resp 5, 6 and 0). The recognition step has reached with TOM map the rate of 95% but

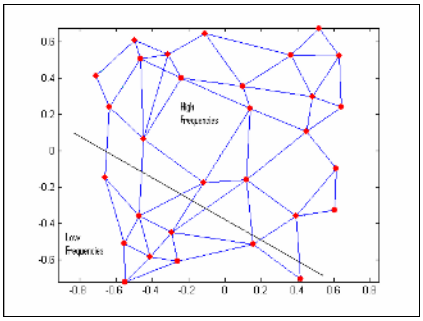


Fig. 4. Spatio-temporal topology of STOM

never 100%. Moreover, STOM has preserved the topology of the input, in fact we have taken randomly two weights of STOM weighting vector and we have plotted them (Fig. 4). We have observed that digits with high frequencies are represented by SU closely near each other. The same result is obtained for low frequencies.

We know that Hidden Markov Model (HMM) is one of the most powerful statistical tools available for designing the automatic speech recognition [19]. However, this model does not have any biological foundation. So regarding to HMM, STOM gives the same result (reaches a learn rate of 100% for the majority of digits) yet, with a biological plausibility.

6 Conclusion

The Spatio-Temporal Organization Map proposed in this paper is an extension of TOM map to spatio-temporal domain. It's an intrinsic and extrinsic model because it embeds the two dimensions together at the level of input architecture and processing. For more conformity to the cortex, neurons proposed in the model could be replaced by an STOM map and further research could be focused on the feedforward and backward links between two or more STOM maps. Furthermore, the activation of one Super Unit could be complex instead of real, which express more the spatio-temporal representation and allows a more meaningful interaction between Super Units.

References

1. F. Alexandre.F, 'A functional design of cortex: the cortical column' visual and motor aspects. PhD thesis. University of Nancy 1990.
2. VB. Mountcastle, 'Perceptual Neurosciences: the cerebral cortex.' *Havard university press, Cambridge Massachustts and London England 1998.*
3. HC. William, 'Columns and modules'. *MIT encyclopedia of the cognetive sciences 1998.*
4. A. R. R. Casti, A. Omurtag, A. Sornborger, E. aplan, B. Knight, J. Victor, L. Sirovich , 'A population study of integrate and fire or burst neur*on'*, *Neural Computation, Volume 14 Issue 5, May 2002.*

5. S. Durand, ‘Learning speech as acoustic sequences with the unsupervised model TOM’. In NEURAP, 8th inter conf on neural networks and their applications. Marseille france 1995.
6. P. Haffner, A. Waibel, ‘A Multi-State time delay neural networks for continuous speech recognitio*n’. In adv in neural information prong system 4. p 135-143. San Mateo, CA 1992.*
7. W. Gestner, W. Kistler, ‘Spiking neuron models’. *Cambridge university Press 2002.*
8. J. C. Wiemer, ‘The Time-Organized Map (TOM) algorithm: extending the self-organizing map (SOM) to spatiotemporal signals’. *Neural Computing, 15, May 2003*
9. R. Fitch, S. Miller, P. Tallal, ‘Neurobiology of speech perception.’ *Annual review of neuroscience 20. p 331-353 1997.*
10. L. Aitkin, ‘The auditory cortex, structural and functional bases of auditory perception’. *Chapman and Hall 1990.*
11. G.Vaucher, ‘A la recherche d'une algerbre neuronale spatio-temporal’. *P.hD. Supelec, Renne University 1996.*
12. M. R. Ashouri, ‘Isolated word recognition using a high-order statistic and time delay neural network’, *Proc of the 1997 IEEE Signal Processing Workshop on H-O-Ss, july 1997*).
13. N. Mozayyani, V. Alanou, J.F Derfus, G. Vaucher, ‘A spatio-temporal data coding applied to kohonen maps’. *Inter conf on Artificial Neural Natwork. October 1995, 2, p 75-79.*
14. N. Mozayyan, ‘Introduction au codage spatio-temporel dans les architectures classiques des réseaux de neurones artificiels.’ *PhD, Suplec, campus universitaire de rennes, juillet 1998.*
15. A.R Baig, ‘Une approche méthodologique de l'utilisation des STAN applique a la reconnaissance visuelle de la parole’. *PhD, Suplec, campus universitaire de rennes, avril 2000.*
16. P. Dorrizzi, J.M Duval and Hebar. H,‘Utilisation des réseaux récurrent pour la prévision de la consommation électrique’. *In Proc.Neural Networks and their application-France 1992.*
17. G. Vaucher, ‘A Complex-Valued spiking machine’, *ICANN 2003, p 967-976.*
18. S. Thorpe, ‘Spiking arrival times: Ahighly efficient coding scheme for neural networks’. *In Parallel Processing in Neural System, Elseiver Press, 1990.*
19. Y. Zhang ‘Speaker independent isolated word’, Technical report, university of western, Australia, January 2002.
20. Calliope. La parole et son traitement automatique. *Masson, Paris, Milan , Barcelone*, 1989.
21. Kanzing, J, ‘Time delayed self-organizing maps’, *IJCNNp 331-336, 1990.*
22. Chappell, G and Taylor, M, ‘The Temporal Kohonen Map’, *Neural Networks, Volume 6, p 441-455.*
23. Koskela. T, Vastra. M, Heikkonon. J and Kaski. K,’Time series prediction using recurrent SOM with local linear models’, Int *Journal on knowledge-based Intelligent Engineering System, volume 6, p 60-68.*

Residual Activity in the Neurons Allows SOMs to Learn Temporal Order

P. Campoy and C.J. Vicente

DISAM - Universidad Politcnica Madrid
campoy@etsii.upm.es

Abstract. A novel activity associated to the neurons of a SOM, called Residual Activity (RA), is defined in order to enlarge into the temporal domain the capabilities of a Self-Organizing Map for clustering and classifying the input data when it offers a temporal relationship. This novel activity is based on the biological plausible idea of partially retaining the activity of the neurons for future stages, that increases their probability to become the winning neuron for future stimuli. The proposed paper also proposes two quantifiable parameters for evaluating the performances of algorithms that aim to exploit temporal relationship of the input data for classification. Special designed benchmarks with spatio-temporal relationship are presented in which the proposed new algorithm, called TESOM (acronym for Time Enhanced SOM), has demonstrated to improve the temporal index without decreasing the quantization error.

Keywords: Time sequence learning, SOM, intrinsic dimensionality.

1 Introduction

In most real problems the time sequence of the input data is relevant for its correct classification and therefore also relevant in the training phase. The aim of standard SOMs can be seen as a reduction of the dimensionality of the input data into an output map of a reduced dimension, generally a bi-dimensional map but as the intrinsic dimensionality of the input data is rarely of dimension two, different output maps can be produced by a particular SOM. Since in most real applications every stimulus is related to the preceding and ulterior one, it is plausible that two input data that occur one after the other should be considered similar to each other. If this assumption is to be learned by a SOM, it yields to the fact that two subsequent input data (i.e. similar) have to activate neighbor neurons in the output space, implying the similarity of the input data.

2 Previous Related Works

The idea of integrating time information into Self Organizing Maps has already been undertaken by some authors using several approaches. Some of the earlier attempts were based on the addition of time information into the input data

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 379–384, 2005.

themselves [5], [7], where exponential averaging and tapped delay lines were tested. Other later approaches consist on using layered or hierarchical SOM in which a second map tries to capture the spatial dynamics of the input moving through the first map [4], [6] that increases the dimensionality of the network size. As an alternative to the leaky integration methods discussed above, Voegtlin [8] proposes a recursive SOM by adding time-delay feedback. Biologically plausible local communication techniques have increasingly been used in neural networks, as proposed by Ruwish [1] where the learning algorithm includes an active medium using diffusion that can be implemented in analog hardware. Also using the reactive-diffusion mechanism Principe and Euliano [2] propose the SOMTAD architecture, whose key concept is the activity diffusion of the neurons through the output map, including an auto-feedback that provokes that the activity decreases A new structure of the same authors [3] is the GASTAS algorithm that suppress the imposition of a predefined (spatial) neighborhood structure on the output space by creating an auxiliary connection matrix trained in the learning phase. An alternative to incorporate network memory representing spatial patterns is introduced by Wiemer [9] in the TOM algorithm, where the temporal stimulus relations is used to generate feed-forward weight vectors that capture essential invariance and serve as reasonable templates for further stimulus processing and to arrange these templates in a topographically meaningful way.

3 Equations of the Learning Algorithm

The best matching criterion proposed in this paper takes into account not only the distance in the input space of the input instance to each neuron (i.e. the "spatial" similarity), but also the Residual Activity containing information about the time since it learned a previous input data (i.e. the "temporal" similarity). Let first define the diverse activities associated with each neuron and their equations:

- Instantaneous Activity $IA_i(k)$ of neuron $i-th$ at time k. This activity is provoked at each neuron depending on its neighborhood to the input instance.
- Vicinity Activity $VA_i(k)$ of neuron $i-th$ at time k. This activity is provoked at each neuron depending on its neighborhood to the winning neuron at time k. the weights of the neurons at each step of the learning phase.
- Residual Activity $RA_i(k)$ of neuron $i-th$ at time k. This activity is provoked at each neuron depending on its neighborhood to the previous winning neurons and depending on the period of time since it lays within the neighborhood of the previous winners, calculated as:

$$RA_i(k) = 1 - \prod_{j=k-1}^{k-t_m} [1 - DF(k,j) VA_i(j)] \tag{1}$$

where $DF(k, j)$ is the damping function at time $k-th$ of the Vicinity Activity produced at previous time $j-th$. This DF is aimed to be stronger (closer to 0) when the previous time $j-th$ is far away from the present time $k-th$ of the learning phase and thereafter the value of the temporal horizont t_m is not relevant. One possible equation to evaluate DF is:

$$DF(k,j) = e^{-\frac{(k-j)^2}{2\sigma_t^2(k)}} \tag{2}$$

where σ_t indicates the temporal propagation of the VA at instant $j-th$ into the future instant $k-th$ when the RA is being evaluated and it typically decreases over the time.

- Total Activity $TA_i(k)$ of neuron $i-th$ at time k. This activity is composed of the Instantaneous Activity and the Residual Activity of each neuron at each time of the training phase. The Total Activity represents therefore a composition of the "spatial similarity" to the input instance and the "temporal similarity" to previous data. In classical SOM learning algorithm the Total Activity is exactly the same as the Instantaneous Activity. In the new proposed learning scheme, the Total Activity can be calculated by:

$$TA_i(k) = 1 - [1 - IA_i(k)][1 - RA_i(k)] \tag{3}$$

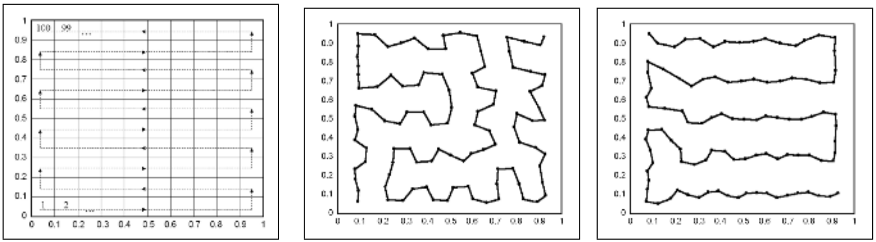
The winning neuron at each stage is therefore that one having the highest value of the Total Activity calculated accordingly to equation 3. The Vicinity Activity is calculated then using a well known type of function which decreases over the distance in the output map to the winning neuron, such as:

$$VA_i(k) = e^{-\frac{-d_i^2(k)}{2\sigma_v^2(k)}} \tag{4}$$

where $d_i(k)$ is the distance in the output map of neuron $i-th$ to the winning neuron in moment $k-th$ and where $\sigma_v(k)$ indicates the spatial propagation of Vicinity Activity around the winning neuron, where the value of $\sigma_v(k)$ typically decreases over the time. Then the winning neurons and its neighbors learn the new instance of the input data in the same way as in the standard SOM.

4 Results

In order to be able to visually compare the results of both algorithms, a bidimensional benchmark has been designed where the input data suffer a high time correlation. The input data have two dimensions $x(k) = [x_1(k) \quad x_2(k)]^T$ that cover a whole unitary square, as seen in figure 1(a), but at any given time k the input instance $x(k)$ can only lay on the neighborhood of the preceding input instance $k-1$, The first input instance has a uniform probability function over a 0.1x0.1 square on the bottom-left corner. Recursively the probability function at each time k is a uniform distribution area 0.1x0.1 square adjacent to the probability square associated to the previous input data at instant $k-1$. These adjacent squares move over the time following the winding line of figure 1(a) until the $100-th$ input instance in the upper-left corner. The described benchmark



(a) Iinput data (b) SOM output map (c) TESOM output map

Fig. 1. Visual results of SOM versus TESOM algorithms. Nodes represent the final value of the neurons and links are their neighbour relationship in output map.

has the remarkable feature that any point within the unitary square has the same probability to become an actual piece of input data, but at the same time the input data presents a well defined time order.

In any one of the obtained results the output map obtained by a classical SOM follows the sequential order of the input data, as it could be expected. When using same benchmark for testing the here proposed TESOM algorithm, in which a damping Residual Activity remains in the activated neurons for future learning stages, the obtained results can be seen in figure 1(c) for the same values of the common parameters of SOM. In order to quantify the results obtained by the new proposed algorithm TESOM for classifying input data with temporal relationship, the following two indexes are proposed for this evaluation propose:

1. Quantization Error (QE). This index is intended to measure how well the input data are represented by the winning neurons in the execution phase, in other words this index quantifies how well the output map is distributed over the input space, calculated by:

$$QE = \sum_i (x(i) - w_w(i))^2 \tag{5}$$

   $x(i)$ is input data and $w_w(i)$ are the weights of the winning neuron.

2. Time Enhancement Index (TEI). This index measures how close are the wining neurons to the previous winning one in the output map, when the input data follows the learned sequential order. The probability of every position increment in the output space of the winning neuron is firstly evaluated using the test data:

$$p(\Delta(i)) = \frac{f(\Delta(i))}{N-1} \tag{6}$$

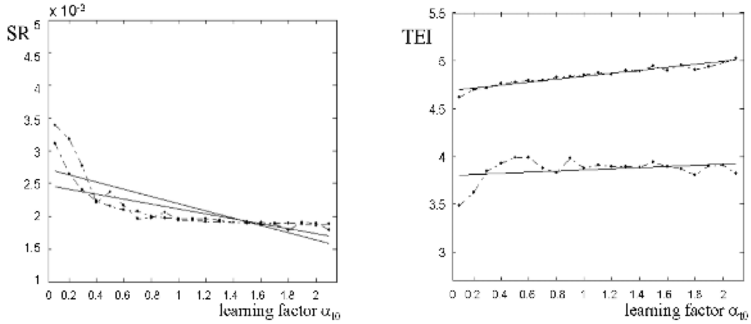
   $f(\Delta(i))$ denotes the number of times (i.e. frequency) the value $\Delta(i)$ is the distance in the output space of two consecutive winning neurons.
   N is the number of instances in the test data set.

Now the Time Enhancement Index *TEI* is calculated as the Entropy of the above calculated frequency distribution by:

$$TE = -\sum_{i=-n}^{n} p(\mathit{\Delta}(i))log_2(p(\mathit{\Delta}(i))) \tag{7}$$

where n is the number of neurons in the output map.
It is interesting to note that the obtained *TEI* denotes the number of bits necessary to transmit the information of the present winning neuron related to the previously winning one.

The results obtained using SOM and TESOM depend on the values selected for the tunable parameters σ_{v0} and α_0, common for both algorithms, and on the value of σ_{t0} in the case of the TESOM algorithm. The experiments on the benchmark have been carried out for a wide range of mentioned three parameters. Since in the case of TESOM the results are also affected by σ_{t0}, a good value of this parameter has to be selected depending on the values of σ_{v0} and α_0, that means that the time influence (parameterized by σ_{t0}) is related to the spatial influence (parameterized by σ_{v0}), as it could be expected.



(a) *QE* of SOM and TESOM for $\sigma_{v0} = 0.5$

(b) *TEI* of SOM and TESOM for $\sigma_{v0} = 0.5$

Fig. 2. *QE* and *TEI* of TESOM and SOM versus α_0

As it can be seen in figure 2 the results obtained by TESOM have smaller *TEI* than those of the SOM algorithm for every value of σ_{v0} and α_0 within the whole range of these parameters, while at the same time the results of both algorithms are very similar in terms of *QE*. Therefore it can be conclude that the TESOM algorithm yields to a good distribution of the neurons all over the training data, as good as the SOM does, but accomplishing simultaneously the goal that consecutive input data activate neighbor neurons, that is measured by the *TEI* index defined above.

5 Discussion and Conclusions

This paper demonstrates through a practical benchmark and quantitative indexes that the novel idea of a Residual Activity in the neurons can improve SOM algorithms for mapping purposes when the input data present a sequential order (i.e. the input data are time related). This Residual Activity increases at each neuron each time it is activated by the input instance and it decreases over the time. This Residual Activity is used in the learning phase to increase the Total Activity of each neuron and therefore to increase its probability to become the winner and therefore to learn the incoming input instance.

The results of the TESOM algorithm have been tested using a specially designed benchmark with a spatio-temporal relationship in the input data. The presented results have demonstrated that the TESOM algorithm distributes the output map over the input data as well as the standard SOM does, but it has the additional advantage that the neurons in the output map follow the same sequential order than the input data do, this is the feature that is to be exploited in the execution phase for better classification. These two parameters have allowed to quantify the results of section 4 that clearly demonstrate that the TESOM and SOM are equally distributed over the input data space (i.e. same QE) and that the TESOM has always lower values of TEI.

References

1. Matias Bode Dietmar Ruwisch and Hans-Georg Purwins. Parallel hardware implementation of kohonen's algorithm with an active medium. *Neural Networks*, 6:1147–1157, 1993.
2. Neil R. Euliano and Jose C. Principe. Spatiotemporal self-organizing feature maps. In *International Joint Conference on Neural Networks, Proceedings of*, volume 4, pages 1900–1905, 6 1996.
3. Neil Euliano Jose Principe and Shayan Garani. Principles and networks for self-organization in space-time. *Neural Networks*, (15):1069–1083, 2002.
4. Jari Kangas. Phoneme recognition using time-dependent versions of self-organizing maps. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 101–104, 1991.
5. T. Kohonen. The hypermap architecture. *Artificial Neural networks*, pages 1357–1360, 1991.
6. P. Morrasso. Self-organizing feature maps for cursive script recognition. *Artificial Neural networks*, pages 1323–1326, 1991.
7. G. Vauche N. Mozayyani, V. Alanou J.F. Dreyfus. A spatio-temporal data-coding applied to kohonen maps. *Proceedings of the International Conference on Artificial Neural Networks*, 2:75–79, 1995.
8. Thomas Voegtlin. Recursive self-organizing maps. *Neural Networks*, 15(8-9):979–991, 10-11 2002.
9. Jan C. Wiemer. The time-organizer map algorithm: Extending the self-organizing map to spatiotemporal signals. *Neural Computing*, (15):1143–1171, 2003.

Ordering of the RGB Space with a Growing Self-organizing Network. Application to Color Mathematical Morphology

Francisco Flórez-Revuelta

U.S.I. "Informática Industrial y Redes de Computadores", Universidad de Alicante,
P.O. Box 99, E-03080 Alicante, Spain
florez@dtic.ua.es
http://www.ua.es/i2rc

Abstract. Mathematical morphology is broadly used in image processing, but it is mainly restricted to binary or greyscale images. Extension to color images is not straightforward due to the need of application to an ordered space with an infimum and a supremum. In this paper a new approach for the ordering of the RGB space is presented. The adaptation of a linear growing self-organizing network to the three-dimensional color space allows the definition of an order relationship among colors. This adaptation is measured with the topographic product to guarantee a good topology-preservation of the RGB space. Once an order has been established, several examples of application of mathematical morphology operations to color images are presented.

1 Introduction

Mathematical morphology (MM) has been broadly used in image processing. MM must be applied to a set provided with an order and with a supremum and an infimum pertaining to that order [1]. This is the reason why MM has been mainly applied to binary or grayscale images. Extension to color images is not straightforward, because those two requirements are missing in color spaces, where there is not an order relationship among colors.

Several techniques have been developed to extend MM to color images, getting partial or total orderings of different color spaces (RGB, HSI, YIQ, CIELAB,...). In marginal ordering [2] MM operations are applied to each image channel, recombining the partial results to get the final image. This method is not generally valid because different or new colors can appear due to this separate filtering.

Another strategy is to treat the color at each pixel as a vector. Order is established by reducing each multivariate color to a ranked single value [3], [4], [5], [6].

2 Ordering of the RGB Space with a Self-organizing Network

Self-organizing neural networks carry out a reduced representation of a vector space in order to achieve some goals [7]: quantization error minimization, entropy maximization, feature mapping, clustering,…

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 385 – 390, 2005.

Adaptation of a network of less dimensionality than the input space causes the creation of folders given that the network tries to preserve its topology. In case of a one-dimensional network the final result is similar to a Peano space filling curve [8].

This effect will allow the ordering of the three-dimensional RGB space with a linear self-organizinf network. A modification of the Growing Cell Structure model [9] will be used, setting two neurons at points representing black and white colors in the RGB space.

2.1 Ordering Growing Self-organizing Network

In the RGB space, each color is represented by three components, each one of them taking real values in the range $[0,1]$.

The ordering neural network is composed of:

- a set $\mathcal{A}$ of nodes (neurons). Each neuron $c \in \mathcal{A}$ has an associated reference vector $w_c \in [0,1]^3 \subset \mathbb{R}^3$. The reference vectors can be regarded as positions in the RGB space of their corresponding neurons,
- a set $\mathcal{C}$ of edges (connections) between pairs of neurons. Those connections are not weighted and their purpose is to define the topological structure.

The learning algorithm to approach the network to the RGB space is as follows:

1. Initialize the set $\mathcal{A}$ with three neurons

$$\mathcal{A} = \{c_0, c_1, c_2\} \tag{1}$$

where $c_0 = (0,0,0)$, $c_1 = (1,1,1)$ and c_2 is randomly chosen from the RGB space. Neurons c_0 y c_1 will be the extremes of the network, placed in the black and white colors respectively.

2. Connection set $\mathcal{C}$ is initialized joining the extremes to neuron c_2, such that the network is one-dimensional:

$$\mathcal{C} = \{(c_0, c_2), (c_1, c_2)\} \tag{2}$$

3. Generate at random an input signal $\xi \in [0,1]^3$ with a uniform distribution.
4. Determine the nearest neuron (winner neuron) s:

$$s(\xi) = \boldsymbol{arg\ min}_{c \in \mathcal{A}} \|\xi - w_c\| \tag{3}$$

5. Add the squared distance between the input signal and the winner neuron to an error variable $\mathcal{E}_s$:

$$\Delta\mathcal{E}_s = \|\xi - w_s\|^2 \tag{4}$$

6. Adapt the reference vectors of s and its topological neighbors (neurons connected to s) towards ξ by a learning step ε_s and ε_n, respectively, of the total distance:

$$\Delta w_s = \varepsilon_s\left(\xi - w_s\right), \quad s \in \mathcal{A} - \left\{c_0, c_1\right\} \tag{5}$$

$$\Delta w_n = \varepsilon_n\left(\xi - w_n\right), \quad \forall n \in \mathcal{N}_s - \left\{c_0, c_1\right\} \tag{6}$$

where $\mathcal{N}_s$ denotes the set of neighbors of s.

7. Every certain number λ of input signals generated, insert a new neuron as follows:
   - Determine the neuron q with the maximum accumulated error:
     $q = \arg\max_{c \in \mathcal{A}} \mathcal{E}_c$
   - Insert a new neuron r between q and its further neighbor f:
     $$w_r = 0.5\left(w_q + w_f\right) \tag{7}$$
   - Insert new edges connecting the neuron r with neurons q and f, removing the old edge between q and f.
   - Decrease the error variables of neurons q and f multiplying them with a constant α. Set the error variable of r with the mean value of q and f.
8. Decrease all error variables by multiplying then with a constant β.
9. If the net size is not yet achieved, go to step 3.

Figure 1 shows the result of the adaptation of different networks to the RGB space and the sequence of reference vectors from black to white color.

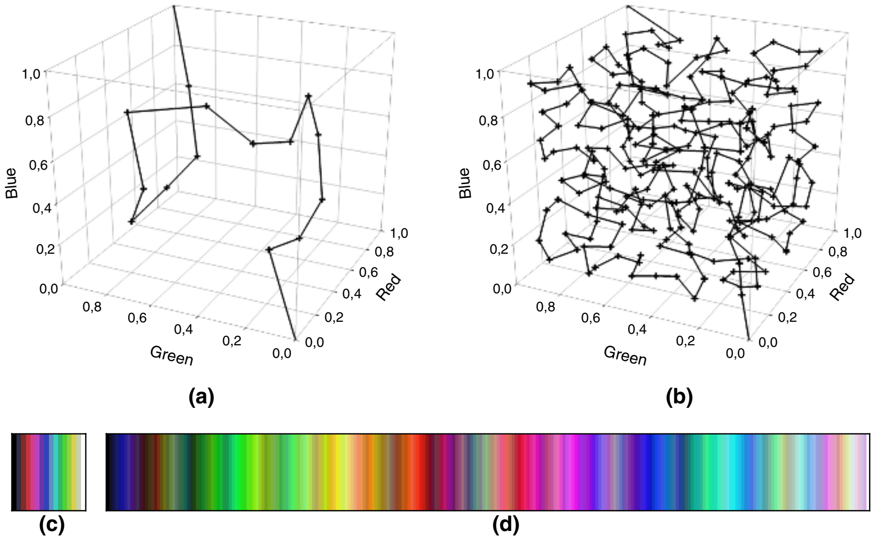


Fig. 1. Adaptation of a growing self-organizing network to the RGB space: (a) 16 and (b) 256 neurons; and their corresponding color orderings: (c) 16 and (d) 256 reference vectors

2.2 Color Order

A mapping of the RGB space onto the neural network is obtained once the learning process is finished :

$$\phi:[0,1]^3 \rightarrow \mathbb{N}, \xi \in [0,1]^3 \rightarrow \phi(\xi) \in \mathbb{N} \tag{8}$$

where $\phi(\xi)$ is obtained from:

$$\left\| w_{\phi(\xi)} - \xi \right\| = \min_{i/c_i \in A} \left\| \xi - w_c \right\| \tag{9}$$

However, two different colors can be mapped to the same neuron. This fact establishes a preorder relationship, not being able to ensure the uniqueness of infimum or supremum in a mathematical morphology operation.

In order to avoid this problem, color order is defined as:

$$\xi_1 < \xi_2 \quad if \quad \begin{cases} i < j \quad / \quad c_i = \phi(\xi_1), c_j = \phi(\xi_2) \\ c = \phi(\xi_1) = \phi(\xi_2) \quad \wedge \quad \left\| \xi_1 - w_c \right\| < \left\| \xi_2 - w_c \right\| \end{cases} \tag{10}$$

Even so, two colors could be at the same distance from the reference one. To solve this problem, one possible criterion is choosing the first point found in the calculation of the infimum or the supremum [10] (figure 2).

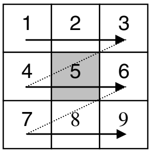


Fig. 2. Calculation route with a structuring element of size 3x3

3 Application to Color Mathematical Morphology

Once a color order ϕ has been defined, thereby allowing the choice of a supremum and infimum from a set of vectors; one can define the basic morphological operators.

The erosion of a digital image $I:\mathbb{N}^2 \rightarrow \mathbb{R}^3$ at point (x,y) by structuring element B is

$$\varepsilon_{\mathcal{B}}^{I(x,y)} = \inf_{(s,t)\in\mathcal{B}} \phi\left[I(x+s,y+t)-b(s,t)\right] \tag{11}$$

and the corresponding dilation is

$$\delta_{\mathcal{B}}^{I(x,y)} = \sup_{(s,t)\in\mathcal{B}} \phi\left[I(x-s,y-t)+b(s,t)\right] \tag{12}$$

Usually, when been applied to images, $b(s,t)=0, \quad \forall (s,t) \in B$, getting the so-called function-and-set-processing (FSP) filters:

$$\varepsilon_{\mathcal{B}}^{I(x,y)} = \inf_{(s,t)\in\mathcal{B}} \phi\left[I(x+s,y+t)\right] \tag{13}$$

$$\delta_{\mathcal{B}}^{I(x,y)} = \sup_{(s,t)\in\mathcal{B}} \phi\left[I(x-s,y-t)\right] \tag{14}$$

From these elementary operations there can be defined a wide set of morphologic operations as opening, closing, gradients or top-hats.

4 Results and Experimentation

One million networks have been adapted, taking the following learning parameters: size=256 neurons, ε_1=0.1, ε_2=0.01, λ=10000, α=0, β=0. Quality of the adaptation can change due to the random behaviour of the learning process. So, topology-preservation has been calculated with the topographic product [11].

The best result has been employed for obtaining the reduced ordering of the RGB space. The examples in this section use an image of the painting "Le chanteur" by Joan Miró and a square structuring element of size 5x5. Figure 3 shows the results of applying different color morphological operations.

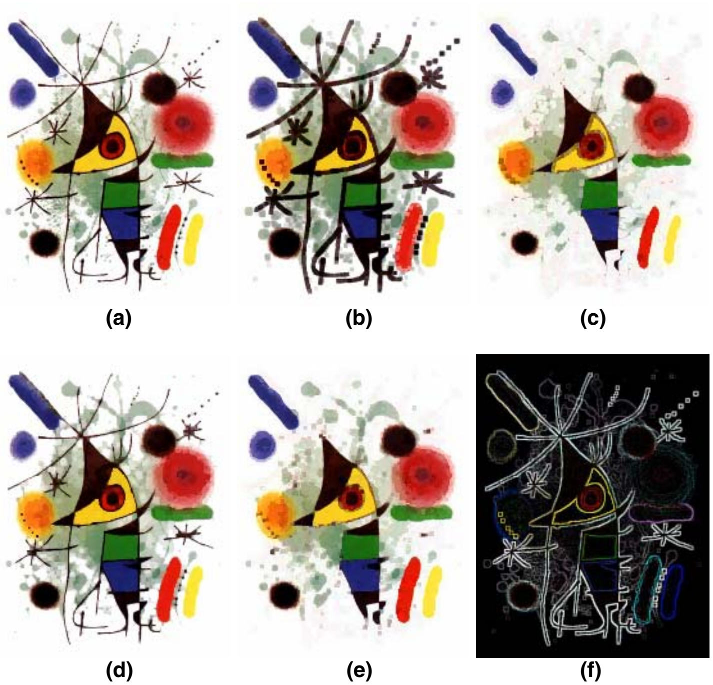


Fig. 3. (a) Original image, (b) erosion, (c) dilation, (d) opening, (e) closing and (f) gradient by erosion

5 Conclusions and Current Works

In this paper a new method for the ordering of the RGB space is presented, mainly directed to its later application to mathematical morphology. This ordering is better, in sense of topology-preservation, than other methods; and can be easily implemented in a parallel hardware architecture in order to ensure a fast image processing.

Nowadays, works are focused on the extension of this approach to other color spaces as HIS or YIQ; studying the right learning parameters to be used in each case. Next, ordering will be carried out to other image feature spaces, for instance, to textures.

Outcoming orderings will be applied to different image processing applications as hand gesture recognition, robotics and automatic visual inspection, in which my research laboratory is involved.

Acknowledgements

This work has been partially funded by the Spanish Ministry of Science and Technology under project DPI2002-04434-C04-01.

References

1. Serra, J.: Image Analysis and Mathematical Morphology Volume 2: Theoretical Advances. London: Academic Press (1988)
2. Barnett, V.: The ordering of multivariate data. Journal of the Royal Society A, vol 139 (1976) 318-355
3. Comer, M.L., Delp, E.J.: Morphological Operations for Color Image Processing. Journal of Electronic Imaging, Vol. 8, No. 3 (1999) 279-289
4. Chanussot, J., Lambert, P.: Total ordering based on space filling curves for multivalued morphology. In Proceedings of the International Symposium on Mathematical Morphology (ISMM'98) (1998) 51-58
5. Peters II, R.A.: Mathematical morphology for angle-valued images. In Non-Linear Image Processing VIII. SPIE volume 3026 (1997)
6. Hanbury, A., Serra, J.: Mathematical Morphology in the HLS Colour Space. In Proceedings of the 12th British Machine Vision Conference (2001) 451-460
7. Fritzke, B.: Some Competitive Learning Methods. Draft Paper (1997)
8. Kohonen, T.: Self-organizing maps. Springer (1995)
9. Fritzke, B.: Growing Cell Structures – A Self-organizing Network for Unsupervised and Supervised Learning. Neural Networks, Vol. 7, No. 9 (1994) 1441-1460
10. Ortiz, F.G.: Procesamiento morfológico de imágenes en color. Aplicación a la reconstrucción geodésica. Ph. D. Thesis, Universidad de Alicante (2002)
11. Bauer, H.-U., Pawelzik, K.R.: Quantifying the Neighborhood Preservation of Self-Organizing Feature Maps. IEEE Transactions on Neural Networks, 3(4) (1992) 570-578

SOM of SOMs: Self-organizing Map Which Maps a Group of Self-organizing Maps*

Tetsuo Furukawa

Kyushu Institute of Technology, Kitakyushu 808-0196, Japan
furukawa@brain.kyutech.ac.jp
http://www.brain.kyutech.ac.jp/~furukawa

Abstract. This paper aims to propose an extension of SOMs called an "SOM of SOMs," or SOM^2, in which the mapped objects are self-organizing maps themselves. In SOM^2, each nodal unit of the conventional SOM is replaced by a function module of SOM. Therefore, SOM^2 can be regarded as a variation of a modular network SOM (mnSOM). Since each child SOM module in SOM^2 is trained to represent a manifold, the parent SOM in SOM^2 generates a self-organizing map representing the distribution of the group of manifolds modeled by the child SOMs. This extension of SOM is easily generalized in the case of SOM^n, such that "SOM^3 as SOM of SOM^{2}s." In this paper, the algorithm of SOM^2 is introduced, and some simulation results are reported.

1 Introduction

The self-organizing map (SOM) as introduced by Kohonen has provided us with a powerful tool for data mining, classification, analysis, visualization, etc.[1]. In the case of labeled training samples, SOM should be one of the best techniques available for visualizing the distribution of each class. In such a case, an SOM shows how the sample vectors of each class are distributed in the high dimensional data space by transforming them to low dimensional map space while preserving their topological relationships. This means that, if the map is generated appropriately, a distance between two points in the map space signifies either similarity or difference between the corresponding two vectors in the data space. Though this characteristic of SOMs is effective in many applications, some cases require the visualization of such relationship *between the distributions of classes*, i.e., to what degree two class distributions are similar or different. Unfortunately, however, an SOM does not provide such information. An SOM provides a map of data vectors, but it is not a map of class distributions.

The aim of this paper is to propose a method of mapping classes which can represent the relationships between their distributions. In other words, the mapped objects of an SOM are no longer vectors but class distributions that form manifolds in the data space. Since the distribution of each class, i.e., each manifold, can be represented by a basic SOM, all the classes can be modeled by a group of basic SOMs. Thus, the method involves the generation of a self-organizing map of a group of self-organizing maps, that is, an "SOM of SOMs."

* This work was supported by a COE program (center #J19) granted by MEXT of Japan.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 391–396, 2005.

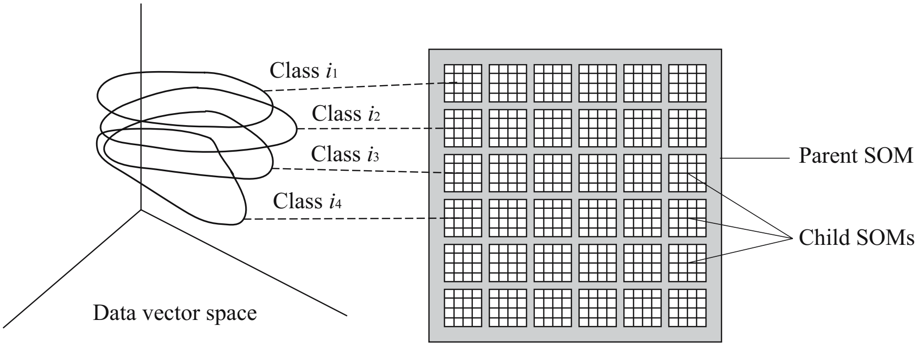


Fig. 1. The architecture and the scheme of SOM^2 as an "SOM of SOMs"

Previously, we have proposed a generalization of an SOM called a "modular network SOM" (mnSOM), in which each nodal vector unit is replaced by a function module of a neural network[2,3]. Such a generalization was first proposed by Kohonen as an "Operator Map"[4]. However, even further generalization is possible by adopting the idea of a modular network because various types of trainable architectures, including the case of the SOM itself, can be chosen as modules. Therefore, an "SOM of SOMs" can be realized by SOM-module-mnSOM as an example of a generalized SOM.

2 Architecture and Algorithm of SOM^2

The architecture of an "SOM of SOMs" is simple — just a replacement of vector units by the basic SOMs, as is shown in Fig.1. In this paper, let us abbreviate this architecture as "SOM^2." Each child SOM learns to represent a manifold in the data space, whereas the parent SOM is expected to generate a map of the manifolds represented by the child SOMs. The learning processes of both levels of SOMs progress in parallel. As in the case of the basic SOM, the algorithm of SOM^2 consists of three processes: the *competitive process*, the *cooperative process*, and the *adaptive process*.

Now suppose that there are I classes, each of which has J sample data. Further, let D_i be the ith dataset, which consists of $\{\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,J}\}$, and let us assume that SOM^2 consists of K child SOMs, each of which has L codebook vectors $W^k = \{\mathbf{w}^{k,1}, \ldots \mathbf{w}^{k,L}\}$. (Here, the superscripts represent the indexes of SOM^2, while the subscripts represent the indexes of classes or data vectors.)

In the *competitive process*, each class dataset is first picked up one by one. Then the child SOM that minimizes the total quantization error is chosen as "the best matching map (BMM)" (i.e., the winning child SOM) of the class. The BMM of the ith class k_i^* is defined as follows.

$$k_i^* = \arg\min_k E_i^k = \arg\min_k \sum_{j=1}^{J} e_{i,j}^{k*} \tag{1}$$

Here, E_i^k denotes the total quantization error of the kth child SOM for the ith class, which is the sum of the square error $e_{i,j}^{k*}$ between $\mathbf{x}_{i,j}$ and the best matching unit (BMU) of the kth child SOM. Thus,

$$e_{i,j}^{k*} = \left\| \mathbf{w}^{k,l_{i,j}^{k*}} - \mathbf{x}_{i,j} \right\|^2 \tag{2}$$

$$l_{i,j}^{k*} = \arg\min_{l} \left\| \mathbf{w}^{k,l} - \mathbf{x}_{i,j} \right\|^2 \tag{3}$$

Here, $l_{i,j}^{k*}$ is the index of BMU of the kth child SOM for the data vector $\mathbf{x}_{i,j}$. This process is repeated for all the classes.

In the *cooperative process*, the learning rates $\{\Phi_i^k\}$ and $\{\phi_{i,j}^l\}$ are calculated by using the neighborhood functions as follows.

$$\Phi_i^k = \frac{g[d(k, k_i^*), T]}{\sum_{i'=1}^{I} g[d(k, k_{i'}^*), T]} \tag{4}$$

$$\phi_{i,j}^l = \frac{h[d(l, l_{i,j}^{**}), T]}{\sum_{j'=1}^{J} h[d(l, l_{i,j'}^{**}), T]} \tag{5}$$

Here, $g[\cdot,\cdot]$ and $h[\cdot,\cdot]$ are the neighborhood functions of the parent and the child SOMs, respectively, which shrink with the calculation time T. $d(\cdot,\cdot)$ refers to the distance between two nodes in the map space, while $l_{i,j}^{**}$ denotes the index of the BMU of the BMM, i.e., $l_{i,j}^{**} = l_{i,j}^{k_i^* *}$.

In the *adaptive process*, all the codebook vectors of the entire child SOMs are innovated by using the learning rates Φ_i^k and $\phi_{i,j}^l$. Now suppose that $V_i = \{\mathbf{v}_i^1, \ldots, \mathbf{v}_i^L\}$ is a set of codebook vectors only for the ith class. In other words, V_i represents the tentative estimation of the ith manifold. By adopting the algorithm of the batch learning SOM, $\{\mathbf{v}_i^l\}$ are defined as follows.

$$\mathbf{v}_i^l = \sum_{j=1}^{J} \phi_{i,j}^l \mathbf{x}_{i,j} \tag{6}$$

Each child SOM is innovated so as to be the weighted interpolation of the estimated manifolds, i.e., the interpolation of $\{V_i\}$ with the learning rates Φ_i^k. Thus,

$$\mathbf{w}^{k,l} = \sum_{i=1}^{I} \Phi_i^k \mathbf{v}_i^l \tag{7}$$

By combining (6) and (7) together, the adaptive process is formulated as follows.

$$\mathbf{w}^{k,l} = \sum_{i=1}^{I} \Phi_i^k \left\{ \sum_{j=1}^{J} \phi_{i,j}^l \mathbf{x}_{i,j} \right\} = \sum_{i=1}^{I} \sum_{j=1}^{J} \Phi_i^k \phi_{i,j}^l \mathbf{x}_{i,j} \tag{8}$$

Equation (8) has a recursive structure like a Russian doll, in which the adaptation algorithm of the basic SOM is nested into itself. Therefore it is easy to extend to the case of SOM^n, such as SOM^3 as the SOM of SOM^2, through further nesting.

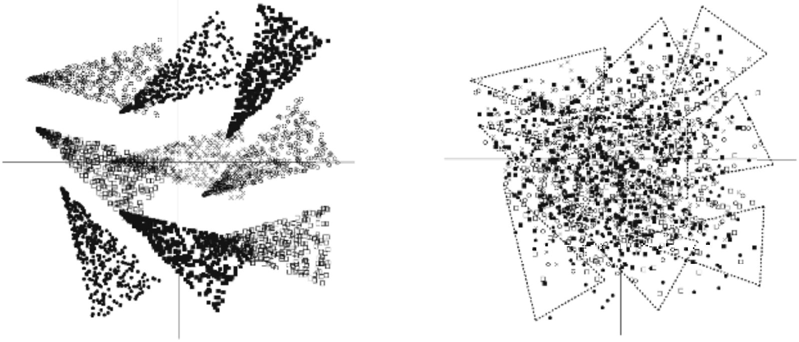
SOM^2 is expected to work in the following way. If the number of classes is larger than the number of child SOMs, i.e., $I > K$, then each child SOM is expected to be assigned to one or more classes as their BMM. In such a case, each child SOM learns the data vectors of the assigned classes in such a way that they can represent their average distribution. On the other hand, in the case of $I < K$, I out of K child SOMs become BMMs, each of which learns the distribution of the corresponding class, whereas the rest of $(K - I)$ child SOMs, which are not BMMs, are trained to represent the intermediate distributions by interpolation of the given classes. Since the number of class depends on the task, SOM^2 is expected to work well in both cases. These situations were reproduced in this study in the first simulation using artificial manifolds (Fig. 2 and Fig. 3).

3 Simulations and Results

3.1 A Case of Artificial Datasets

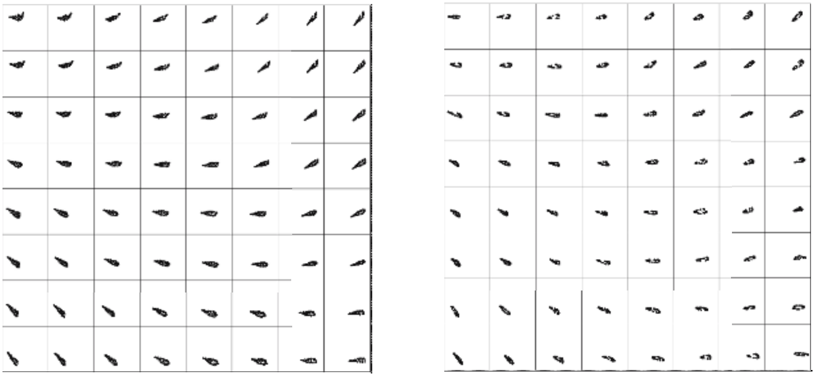
Two sets of artificial manifolds shown in Fig. 2 were used to validate the ability of SOM^2. In the first set (Fig. 2a), there was a small number of classes ($I = 9$), while each class had a large number of data vectors ($J = 400$) that were sampled randomly. The shapes of the manifolds were all congruent triangles, the positions and orientations of which were changed gradually. On the other hand, the second set (Fig. 2b) had a large number of classes ($I = 400$), each of which consisted of a very small number of samples ($J = 4$). The shapes of the manifolds were all congruent triangles just as in the first case, some of which are shown in Fig. 2b with dashed triangles. Unlike the first case, however, it is difficult to recognize the shapes of the manifolds due to the deficiency of the samples. Furthermore, the manifolds of the second set overlapped with each other in such a way that the sample vectors were distributed evenly over the area without forming clusters. In addition, the positions and orientations of the manifolds were changed in the same manner as in the first case. The number of the child SOMs was fixed in both cases ($K = 8 \times 8$), while the number of codebook vectors was 6×6 for each child SOM. Thus, $I < K$ in the first case, and $I > K$ in the second case.

Fig. 3 shows the simulation results. In the figure, the dots plotted in each box indicate the map generated by the corresponding child SOM. In the case of the first manifold set (Fig. 3a, corresponding to Fig. 2a), every child SOM represented the triangle shape well. Furthermore, the positions and orientations varied gradually in such a way that a continuous map of the manifolds was formed successfully in the parent SOM. This result also shows that the unknown intermediate manifolds were appropriately estimated by interpolation. The result was almost the same in the case of the second manifold set (Fig. 3b, corresponding to Fig. 2b). The parent map was well organized with good continuity, i.e., the positions and orientations of the child maps varied continuously. SOM^2 also succeeded in estimating the distributions, even though the number of samples per class was very small. Though not shown in the figure, the child maps were aligned so that the codebook vector of each child SOM with the same index corresponded to a congruent point of each manifold. For example, the BMUs which corresponded to the apex of the triangles had the same index in the child SOMs. Therefore it is possible to



(a) (b)

Fig. 2. The two sets of artificial manifolds used in the first simulation



(a) (b)

Fig. 3. The maps of the manifolds generated by SOM^2 from the dataset of Fig. 2

observe how the manifold gradually varied its shape by tracing the codebook vectors with the same index.

3.2 Classification of 3D Objects from 2D Images by SOM^2

The task of the second simulation was to generate a map of 3D objects from 2D projected images. Put more clearly, the task was to make a self-organizing map of 3D objects from a set of photo albums, each of which contains several photographs of one object from various viewpoints. Since a set of 2D images of a 3D object projected from different viewpoints form a manifold in the high dimensional data space, the distribution of the 2D images can be modeled by a basic SOM. Therefore an assembly of basic SOMs such as SOM^2 would be expected to represent a group of 3D objects. Please notice that SOM^2 does not know how 3D objects can be reconstructed from their 2D images.

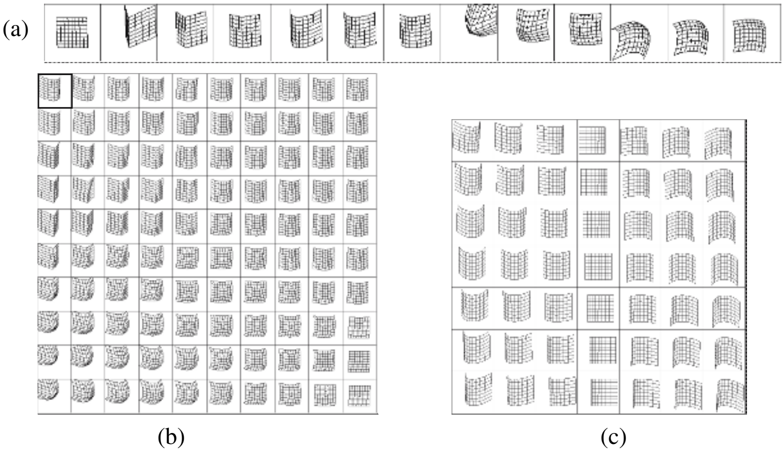


Fig. 4. A map of 3D objects generated by SOM^2 from 2D projected images

The 13 objects shown in Fig. 4a were used in the simulation, and 49 2D images from various viewpoints were prepared for each object. The objects are assumed to be flexible grids, and each data vecotor consisted of a set of coordinates of the lattice points on a 2D image. Fig. 4b is the map of the 3D objects generated by the parent SOM, while a map of a child SOM (corresponding to the thick box in Fig. 4b) is presented in Fig. 4c. The parent map was generated successfully, showing a good continuity of varying 3D shapes. This result also means that SOM^2 created 2D images of unknown intermediate 3D objects by interpolating between the given objects. Furthermore, all child SOMs were aligned with each other in such a way that all codebook vectors with the same index were assigned to the images taken from the same viewpoint.

4 Conclusion

In this paper we have proposed an extension of SOMs called SOM^2. SOM^2 provides a method for the topological mapping of classes by comparing their distributions instead of comparing individual vectors. The simulation results suggest that SOM^2 will be a powerful tool for class visualization and analysis. Further extensions of SOM^2 are currently being developed for datasets without class labels.

References

1. Kohonen, T.: Self-Organizing Maps, 3.ed., Springer (2001)
2. Tokunaga, K., Furukawa, T., Yasui, S.: Modular network SOM: Extension of SOM to the realm of function space. Proc. of WSOM2003 (2003) 173–178
3. Furukawa T., Tokunaga K., Kaneko S., Kimotsuki K., Yasui, S.: Generalized self-organizing maps (mnSOM) for dealing with dynamical systems. Proc. of NOLTA2004 (2004) 231–234
4. Kohonen, T.: Generalization of the Self-organizing map. Proc. of IJCNN93 (1993) 457–462

The Topographic Product of Experts

Colin Fyfe

Applied Computational Intelligence Research Unit,
The University of Paisley,
Scotland
colin.fyfe@paisley.ac.uk

Abstract. We create a new form of topographic map which is based on a nonlinear mapping of a space of latent points. The mapping of these latent points into data space creates centres which are equivalent to those of the standard SOM. We relate this mapping to the Generative Topographic Mapping, GTM. We then show that it is rather simple and computationally inexpensive to grow one of these maps and that a probabilistic interpretation of these maps facilitates our investigation of alternative algorithms.

1 Introduction

We create a new form of self-organising map which is based on an assumption (implicit in the SOM [4], but made explicit here) that the data have been generated by a set of underlying causes. Thus we call this mapping the Topographic Product of Experts (ToPoE).

While this network uses a gradient ascent method to learn parameters, it is closer to the Generative Topographic Mapping [1] (which uses the EM algorithm to optimise parameters) than the SOM and so we review this method in order to highlight the differences between the two generative methods.

2 The Topographic Product of Experts

We envisage that the underlying structure of the data can be represented by K latent points, $t_1, t_2, \ldots, t_K$. To allow local and non-linear modeling, we map those latent points through a set of M basis functions, $f_1(), f_2(), \ldots, f_M()$. This gives us a matrix Φ where $\phi_{kj} = f_j(t_k)$. Thus each row of Φ is the response of the basis functions to one latent point, or alternatively we may state that each column of Φ is the response of one of the basis functions to the set of latent points. One of the functions, $f_j()$, acts as a bias term and is set to one for every input. Typically the others are gaussians centered in the latent space. The output of these functions are then mapped by a set of weights, W, into data space. W is $M \times D$, where D is the dimensionality of the data space and is the sole parameter which we change during training. We will use $\mathbf{w}_i$ to represent the i^{th} column of W and Φ_j to represent the row vector of the mapping of the j^{th} latent point. Thus each basis point is mapped to a point in data space, $\mathbf{m}_j = (\Phi_j W)^T$.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 397–402, 2005.

We may update W either in batch mode or with online learning. To change W in online learning, we randomly select a data point, say $\mathbf{x}_i$. We calculate the responsibility of each latent point for this data point using

$$r_{ij} = \frac{exp(-\gamma d_{ij}^2)}{\sum_k exp(-\gamma d_{ik}^2)} \tag{1}$$

where $d_{pq} = ||\mathbf{x}_p - \sum_k(\phi_{qk}.w_k)||$, the euclidean distance between the p^{th} data point and the projection of the q^{th} latent point (through the basis functions and then multiplied by W). If no weights are close to the data point (the denominator is zero), we set $r_{ij} = \frac{1}{K}, \forall j$.

We calculate $m_{kd} = \sum_{m=1}^{M} w_{md}\phi_{km}$, the projection of the k^{th} latent point on the d^{th} dimension in data space and then use this in the update rule

$$\varDelta_n w_{md} = \sum_{k=1}^{K} \eta\phi_{km}(x_d - m_{kd}^n)r_{kn} \tag{2}$$

so that we are summing the changes due to each latent point's response to the data points. Note that, for the basic model, we do not change the $\varPhi$ matrix during training at all. It is the combination of the fact that the latent points are mapped through the basis functions and that the latent points are given fixed positions in latent space which gives the ToPoE its topographic properties.

3 Comparison with the GTM

The Generative Topographic Mapping (GTM) [1] is a probabilistic model which treats the data as having been generated by a set of latent points. We also have a set of K latent points which are mapped through a set of M basis functions and a set of adjustable weights to the data space. The parameters of the combined mapping are adjusted to make the data as likely as possible under this mapping. The GTM is a probabilistic formulation so that if we define $\mathbf{y} = \mathbf{\Phi W} = \mathbf{\Phi(t)W}$, where $\mathbf{t}$ is the vector of latent points, the probability of the data is determined by the position of the projections of the latent points in data space and so we must adjust this position to increase the likelihood of the data. More formally, let

$$\mathbf{m}_i = \mathbf{\Phi}(\mathbf{t}_i)W \tag{3}$$

be the projections of the latent points into the feature space. Then, if we assume that each of the latent points has equal probability

$$p(\mathbf{x}) = \sum_{i=1}^{K} P(i)p(\mathbf{x}|i) = \sum_{i=1}^{K} \frac{1}{K}\left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} \exp\left(-\frac{\beta}{2}||\mathbf{m}_i - \mathbf{x}||^2\right) \tag{4}$$

where D is the dimensionality of the data space. i.e. all the data is assumed to be noisy versions of the mapping of the latent points. It is the combination of the fact that the latent points are mapped through a smaller number of basis functions and that the latent points are given fixed positions in latent space which gives the GTM its topographic properties.

In the GTM, the parameters W and β are updated using the EM algorithm though the authors do state that they could use gradient ascent.

Hinton [3] investigated a product of K experts with

$$p(\mathbf{x}_n|\Theta) \propto \prod_{k=1}^{K} p(\mathbf{x}_n|k) \tag{5}$$

where Θ is the set of current parameters in the model. Hinton notes that using Gaussians alone does not allow us to model e.g. multi-modal distributions, however the Gaussian is ideal for our purposes. Thus our base model is

$$p(\mathbf{x}_n|\Theta) \propto \prod_{k=1}^{K} \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} \exp\left(-\frac{\beta}{2}||\mathbf{m}_k - \mathbf{x}_n||^2\right) \tag{6}$$

If we allow latent points to have different responsibilities depending on the data point presented, we have:

$$p(\mathbf{x}_n|\Theta) \propto \prod_{k=1}^{K} \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} \exp\left(-\frac{\beta}{2}||\mathbf{m}_k - \mathbf{x}_n||^2 r_{kn}\right) \tag{7}$$

where r_{kn} is the responsibility of the k^{th} expert for the data point, $\mathbf{x}_n$. Thus all the experts are acting in concert to create the data points but some will take more responsibility than others.

4 Simulations

Figure 1 shows the result of a simulation in which we have 20 latent points deemed to be equally space in a one dimensional latent space, passed through 5 Gaussian basis functions and then mapped to the data space by the lineaer mapping W which is the only parameter we adjust. We generated 60 two dimensional data points from the function $x_2 = x_1 + 1.25\sin(x_1) + \mu$ where μ is noise from a uniform distribution in [0,1]. We use 10000 iterations of the learning rule (randomly sampling with replacement from the data set) with $\beta = 2, \gamma = 20, \eta = 0.1$. The final placement of the projections of the latent points is shown by the asterisks in the figure and we clearly see the one dimensional nature of the data. We have similar results when we use a batch method, presenting all the data and not updating the weights till we have accumulated all the changes.

We have similar experiments with higher dimensional data and grids e.g. with 400 latent points arranged in a two dimensional grid (20×20) and 5×5 basis functions.

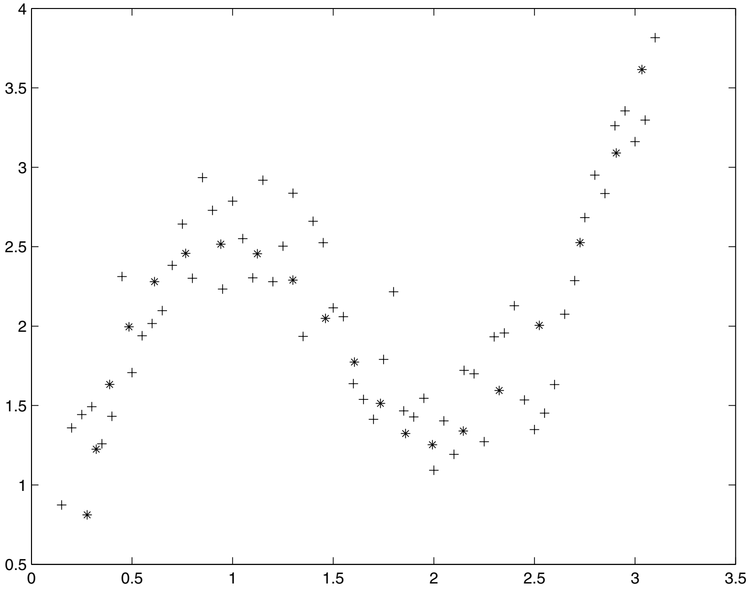


Fig. 1. The projections of 20 latent points into data space is shown by the asterisks. The training data are the other points.

5 Discussion

In this section we consider two extensions of the basic mapping but first we discuss one property of the method.

5.1 Projections

As a visualisation technique the ToPoE has one advantage over the standard SOM: the projections of the data onto the grid need not be solely to the grid nodes. If we project each data point to that node which has highest responsibility for the data point, we get a similar quantisation as the SOM. However if we project each data point onto $\mathbf{p}_k * r_{kn}$, we get a mapping onto the manifold at intermediate points. Figure 2 shows the responsibilities which 20 latent points have for 60 data points (arranged in ascending order of their position along the manifold) and the subsequent re-projection of the latent points to the data space when taking these responsibilities into account.

5.2 Growing ToPoEs

One advantage of this method is that we can easily grow a net: we train a net with a small number of latent points and then increase the number of latent points. We spread the latent points evenly in latent space between -1 and 1:

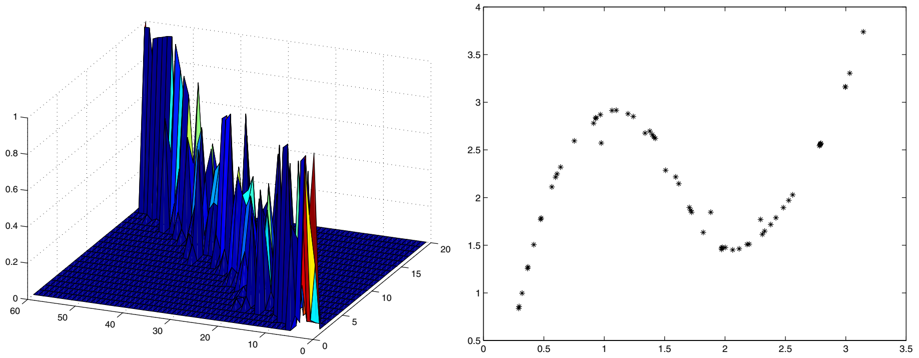


Fig. 2. Left: the responsibilities of the 20 latent points for 60 data points which are arranged in approximately increasing distance along the manifold. Right: the reprojection of the 60 data points onto the manifold.

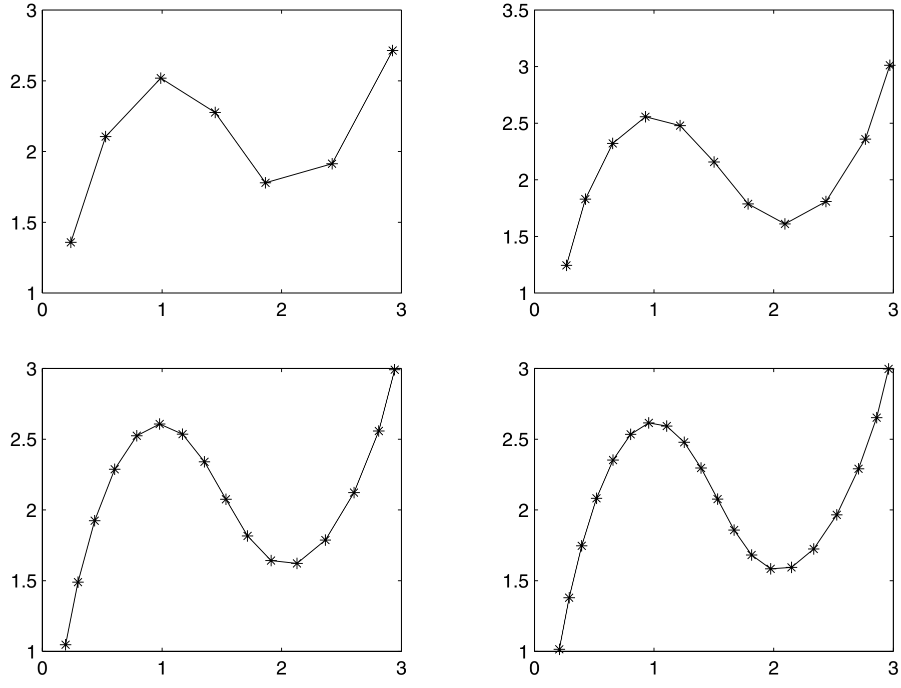


Fig. 3. The growing map. Top left : 7 latent points. Top right: 11 latent points. Bottom left: 16 latent points. Bottom right: 20 latent points.

thus 5 points are positioned at [-1,-0.5,0,0.5,1] while when we grow to 6, these are positioned at [-1,-0.6,-0.2,0.2,0.6,1]. Note that we have to recalculate the Φ matrix but need not change the W matrix of weights which can simply go on learning. An example is shown in Figure 3 in which we use 5 basis functions (+

a bias term) and increase the number of latent points from 7 to 20. The mapping becomes increasingly smooth.

5.3 Different Noise Models

An alternative model based on a Laplacian distribution is

$$p(\mathbf{x}_n) = \left(\frac{\beta}{2\pi}\right)^{\frac{D}{2}} \exp\left(-\frac{\beta}{2}\sum_{k=1}^{K}(||\mathbf{m}_k - \mathbf{x}_n||_1 r_{kn})\right) \tag{8}$$

where $||.||_1$ signifies the 1-norm [2]. In this case, we derive the learning rule

$$\Delta_n w_{md} = \sum_{k=1}^{K} \eta\phi_{km}\text{sign}(x_d - p_{kd}^n) r_{kn} \tag{9}$$

where $\text{sign}(t) = 1,$ if $t > 0$ and $\text{sign}(t) = -1,$ otherwise.

While this rule may be more appropriate for rather more kurtotic noise than in the above simulations, it can be used with data which is corrupted by Gaussian noise or even uniform (and hence far from kurtotic) noise. Simulations on exactly the same data as used for Figure 1 have shown similar convergence to that achieved in that figure.

References

1. C. M. Bishop, M. Svensen, and C. K. I. Williams. Gtm: The generative topographic mapping. *Neural Computation*, 1997.
2. C. Fyfe and D. MacDonald. Epsilon-insensitive hebbian learning. *Neurocomputing*, 47:35–57, 2002.
3. G. E. Hinton. Training products of experts by minimizing contrastive divergence. Technical Report GCNU TR 2000-004, Gatsby Computational Neuroscience Unit, University College, London, http://www.gatsby.ucl.ac.uk/, 2000.
4. Tuevo Kohonen. *Self-Organising Maps.* Springer, 1995.

Self Organizing Map (SOM) Approach for Classification of Power Quality Events

Emin Germen[1], D. Gökhan Ece[2], and Ömer Nezih Gerek[3]

Anadolu University, Turkey
{egermen, dgece, ongerek}@anadolu.edu.tr

Abstract. In this work, Self Organizing Map (SOM) is used in order to classify the types of defections in electrical systems, known as Power Quality (PQ) events. The features for classifications are extracted from real time voltage waveform within a sliding time window and a signature vector is formed. The signature vector consists of different types of features such as local wavelet transform extrema at various decomposition levels, spectral harmonic ratios and local extrema of higher order statistical parameters. Before the classification, the clustering has been achieved using SOM in order to define codebook vectors, then LVQ3 (Learning Vector Quantizer) algorithm is applied to find exact classification borders. The k-means algorithm with Davies-Boulding clustering index method is applied to figure out the classification regions. Here it has been observed that, successful classification of two major PQ event types corresponding to arcing faults and motor start-up events for different load conditions has been achieved.

1 Introduction

Nowadays, the increasing demands to the highly sensitive electronic devices in broad kind of area and the considerably big investments to the technological equipments require a good quality of power. The Power Quality (PQ) concept hence started to take attraction of researchers and technologically related people. One of the most common PQ events of voltage sag is the drop off effective value of the voltage between 10%-90% of its nominal value. The distribution system faults or switching on large loads have significant effects on the voltage waveform and disturbances can be observed as voltage sags. Also arcing faults or starting of motors in the energy grid can cause similar abnormalities on the quality of power. From the point of view of maintenance of the complex systems, the events possessing imperfections have to be classified and the related precautions have to be taken into consideration. In the previous work on PQ event detection and classification problem, wavelets were used for the detection purpose and FFT (spectral harmonic analyses) was used for the discrimination of event types. Due to the lack of other discriminative parameters, the event discrimination success was arguably limited [1].

Since PQ term broadly refers to pure sinusoidal waveform, it conglomerates different research areas such as power engineering, signal processing and neural networks.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 403 – 408, 2005.

Each discipline has its own contribution in order to either delineate the principle couses of the defections in the waveform or classify the possible reasons of them [1],[2].

Kohonen's Self Organizing Map (SOM) is a neural network which projects the higher dimensional input vector space onto one or two-dimensional array in a nonlinear fashion [3]. It is a valuable non-parametric pattern clustering and classification technique which can be adapted to wide spectrum of fields of science and technology. The SOM network is composed of "neurons", their associated "weights" (roughly, the cluster centroids) and an algorithm for updating the weights, given incoming data. Actually the neurons are the codebook vectors connected in planar lattice structure. The codebook vectors without any neighborhood information constitute a vector quantizer, however the organization of the neurons in two-dimensional lattice with a neighborhood formation gives us a clustering information between the input data set.

In this work a classification technique based on SOM has been used in order to discriminate the arcing fault type events from motor start-up events. Although the two PQ event types look similar from the point of view of voltage sag type waveform, the causes and the results are quite different. Here, in order to discriminate those types of voltage waveforms causing disturbance in the power quality, the data has been acquired from experiments carried on low voltage system by introducing arcing fault and induction motor start up events. The key point after getting the voltage waveforms is determining and extracting the features from data which can be used in the cluster analyses.

In section 2, the proposed method to develop a multi-dimensional feature vector is introduced in detail. Here the time window that contains not only the instant data but, pre and post events will be analyzed and put into the consideration by using a well formulated signal processing methods including local wavelet extrema, short-time spectral harmonics, and local higher order statistical parameters extrema. In section 3, the SOM is introduced and the method to classify the events is explained. In section 4, the results showing the excellent classifications of the events are shown and the discussion on the results is given.

2 PQ Events and Determination of Feature Vectors

The feature vector we have used in this work consists of scalar numbers obtained by three major methods; wavelets, spectrum analysis, and higher order statistical parameters. The instrumentation in our experimental system acquires the voltage and current waveforms and their 50 Hz. Notch filtered versions at a sampling rate of 20 KHz for each waveform. We selected local estimation window size as twice the fundamental period length, which corresponds to a size of 800 samples.

The feature vector used has a length of 19. The first eight numbers inside the feature vector correspond to the wavelet transform extrema for the four-level decomposition of voltage waveform using the Daubechies-4 (db4) orthogonal wavelet. These four levels depict time-frequency localized signatures at different frequency resolutions. The extrema are simply the maximum and the minimum transform values around the instance of a PQ event. It was previously shown by several authors that the transform domain values exhibit high energy at or around PQ event instances. Usually simple thresholding of these coefficient magnitudes is enough to detect the existence of a PQ vent. However, classification between different classes of PQ events, more waveform signatures, as well as sophisticated classifiers are required. In order to verify the usefulness of several decomposition level transform coefficients, we have taken both the minimum, and the maximum values corresponding to four decomposition levels. At decomposition levels higher than four, the time resolution is decreased beyond a factor of 32, which is below the desired time-resolution level.

The ninth coefficient of the feature vector was selected according to a classical spectral analysis. We have evaluated the signal energy exactly at the line frequency (50 Hz), proportioned it to the remaining spectral energy at all other frequencies, and took its reciprocal:

$$V_9 = \frac{\int_{-\infty}^{2\pi 50^-} \Phi_v(\omega)d\omega + \int_{2\pi 50^+}^{\infty} \Phi_v(\omega)d\omega}{\Phi_v(\omega)\big|_{\omega=2\pi 50}} \tag{1}$$

where *v* is the feature vector and v_9 corresponds to its 9♦♒ element, and Φ_v (ω) is the power spectral density of the voltage waveform, *v*(*t*). The remaining ten coefficients are obtained from the higher-order statistical parameters of 50 Hz. notch filtered voltage waveforms including the local central cumulants of order 2, 3, and 4, and skewness and kurtosis. For this parameter estimation, it is reasonable to keep the estimation window size a small integer multiple of the fundamental period length. The selected local estimation window size as twice the fundamental period length is statistically long enough to accurately estimate statistical parameters, and short enough to accurately resolve time localization.

The motivation behind selecting higher order statistical parameters is that, each PQ event can be modeled as a noise contribution over the voltage waveform. In fact, in [4], it was shown that the power system voltage waveform can be modeled as a combination of a pure sinusoid and noise components imposed upon to that sinusoid. The sinusoidal component is not the informative part in terms of an event detection or classification, but its existence greatly perturbs local statistical parameters. On the other hand, the noise component contains valuable information in case of PQ events and transients. For that purpose, during the data acquisition, we removed the 50 Hz sinusoid of the voltage waveform using Frequency Devices❄💣 ASC-50 programmable filter adjusted to a very sharp (20♦♒ order Elliptic) 50 Hz notch filter. Under event-free operation conditions, the output waveform of the filter can be modeled as Gaussian. This model is pretty accurate, because the voltage waveform may be noise-corrupted due to the ambient conditions such as EMI generating loads running on or near the system. From the central limit theorem, the combination of independent random sources adds up to a Gaussian process as the number of sources grows.

3 SOM as a Classifier

Kohonen's SOM is Neural Network, which projects the data vectors $\Lambda \in \mathbb{R}^n$ belong to higher dimensional input space *n* into *m* many codebook vectors of size *n* organized in a two dimensional lattice structure. SOM provides two fundamental issues: the first is the clustering of data and the second is the relationship between the clusters. The clustering is an unsupervised learning period which can be formulated as:

$$M_i(k) = M_i(k-1) + \alpha(k) \cdot \beta(i,c,k)(\Lambda(k) - M_i(k-1)) \quad \forall i \;\; 1 \le i \le m \tag{2}$$

where $\alpha(k)$ is the learning rate parameter which is changed during the adaptation phase and $\beta(i,c,k)$ is the neighborhood function around c where c is the Best Matching Unit index which can be found during training as:

$$c = \arg\min_i \left\| \Lambda(k) - M_i(k) \right\| \tag{3}$$

The relationship between clusters can be seen in the planar surface by checking the distances between the codebook vectors. Although it is difficult to deduce exact relationship between those, since the codebook vector size is much greater than the planar surface size of 2, this gives us an insight about the classification regions.

In this work the feature vectors $\Lambda \in \mathbb{R}^n$ where n = 19 has been used to train the SOM of 5x5 neurons connected in hex-lattice structure. The whole data set is obtained by 120 experiments. The 60 experiments have been carried out for arcing faults. The 30 experimental data of them have been obtained only for inductive and resistive loads. The other 30 experiments have been conducted by introducing adjustable speed drives connected to the experimental settings in order to test the quality of classification. Another 60 data also obtained from the experiments of motor start-up events in similar manner data obtained for arcing faults. Consequently, a data set of 4 classes is formed as:

Class 1: Arcing fault with adjustable speed drives load.
Class 2: Arcing fault without adjustable speed drives load.
Class 3: Motor start-up with adjustable speed drives load.
Class 4: Motor start-up without adjustable speed drives load.

The set of 120 data is divided into two in order to obtain the training and the testing sets. For training set, 20 different random data have been selected from each class. The remaining 40 data are used to test the results. After codebook vectors obtained using SOM training algorithm, the map is partitioned into subspaces to discriminate the classification regions by Learning Vector Quantization (LVQ3) algorithm. LVQ algorithm is the classification algorithm based on adjusting the Gaussian borders between the centers of possible classification regions represented by codebook vectors obtained in SOM[3]. In literature there are several ways of implementation of LVQ. In essence, the algorithms attempts to move the codebook vectors to positions that reflect the centers of clusters in the training data in supervised manner. Actually the aim is finding the Gaussian borders between the codebook vectors belonging to different classes by decreasing the miss-classification ratio. In LVQ1 algorithm the training has been done in roughly, however in LVQ2 and LVQ3 algorithms, much adequate approaches have been developed for well tuning.

The LVQ3 algorithm can be explained as:

$$\begin{aligned} M_i(k+1) &= M_i(k) - \mu(k)\left(\Lambda(k) - M_i(k)\right) \\ M_j(k+1) &= M_j(k) + \mu(k)\left(\Lambda(k) - M_j(k)\right) \end{aligned} \tag{4}$$

where M_i and *Mj* are the two closest codebook vectors to *Λ(k)*, whereby *Λ(k)* and *Mj* belongs to the same class, while *Λ(k)* and *Mi* belong to different classes respectively; furthermore *Λ(k)* must fall zone of a *window* defined as;

$$\min\left(\frac{d_1}{d_2} \quad \frac{d_2}{d_1}\right) > s \ \text{ where } \ s = \frac{\text{1-window}}{\text{1+window}} \tag{5}$$

where d_1 and d_2 are the distance between codebook vectors M_i - *Λ(k)* , and M_j - *Λ(k*). Also it is necessary to have:

$$\begin{aligned} M_i(k+1) &= M_i(k) + \varepsilon(k)\mu(k)\left(\Lambda(k) - M_i(k)\right) \\ M_j(k+1) &= M_j(k) + \varepsilon(k)\mu(k)\left(\Lambda(k) - M_j(k)\right) \end{aligned} \tag{6}$$

where M_i and M_j are the two closest codebook vectors to *Λ(k)*, whereby *Λ(k)* and M_j *a*nd M_i belong to same classes. The *μ(k)* and *ε(k)* parameters are learning rates in the algorithm.

4 Classification Results

There are several techniques to visualize the results of SOM in literature. A well known one is the U-matrix. This method identifies distances between neighboring units and thus visualizes the cluster structure of the map. Note that the U-matrix visualization has much more rectangles that the component planes. This is because in U-matrix, not the codebook vectors but distances between the vectors are shown. High values indicate large distance between neighboring map units, and identify possible cluster borders. Clusters are typically uniform areas of low values. Refer to colorbar to see which colors mean high values in Fig. 1. In the map, there appear to be two clusters. A rough inspection on the U-matrix which is given in Fig. 1 gives idea about possible two classification regions after training. Here the dark regions show the close connections in the component plane which represent the clusters and the light areas comprises the cluster borders.

In order to delineate the exact borders, after LVQ3, Davies-Boulding clustering index method is used. Actually this method is a k-means clustering which denotes the classification borders between the clusters.

Testing the classes found after SOM and LVQ3 methods, using the 40 point test set, it has been observed that 100% correct classification could be obtained (Fig1). Here, the same analysis has been done with different map size, and similar classification results have been obtained. The method for clustering and classification and the feature vectors obtained from voltage waveform are observed to comprise an adequate match to identify types of PQ defections of motor start-up and arcing fault type events.

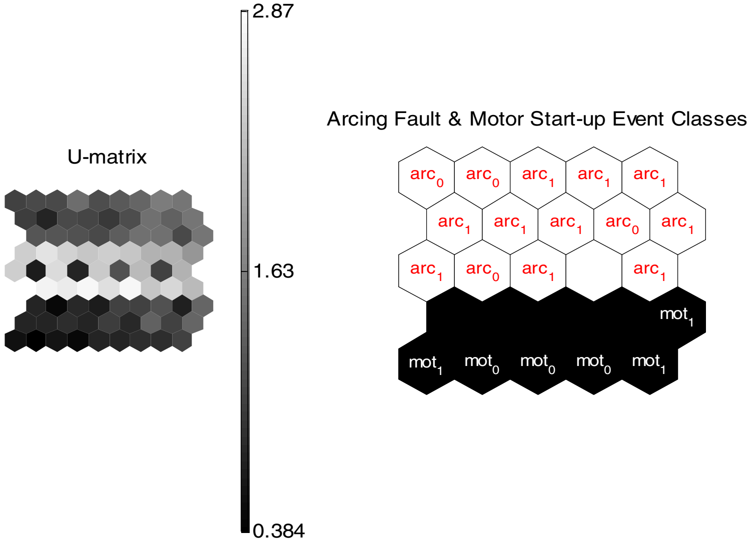


Fig. 1. 5x5 Map formation after SOM training and LVQ3. Here arc$_0$ represent arcing fault with adjustable speed drives and arc$_1$ indicates inductive and resistive load without adjustable speed drives. Similarly mot$_1$ and mot$_0$ represent motor start-up events.

References

1. Wael R., Ibrahim A., Morcos M. M., Artificial Intelligence and Advanced Mathematical Tools for Power Quality Applications: A Survey, IEEE Trans. on Power Delivery, Vol. 17, No. 2, April 2002.
2. Wang M., Mamishev A. V., Classification of Power Quality Events Using Optimal Time–Frequency Representations–Part 1: Theory, IEEE Trans. on Power Delivery, Vol. 19, No. 3, July 2004.
3. Kohonen T., The Self Organizing Map, Proceedings of IEEE, 78, 9, (1990), 1464-1480
4. Yang H.T., Liao C. C., A De–Noising Scheme for Enhancing Wavelet–Based Power Quality Monitoring System, IEEE Trans. on Power Delivery, Vol. 19, No. 1, January 2004.

SOM-Based Method for Process State Monitoring and Optimization in Fluidized Bed Energy Plant

Mikko Heikkinen[1], Ari Kettunen[2], Eero Niemitalo[2], Reijo Kuivalainen[3], and Yrjö Hiltunen[1]

[1] Department of Environmental Sciences,
University of Kuopio,
P.O. Box 1627, FIN-70211 Kuopio, Finland
{mikko.heikkinen, yrjo.hiltunen}@uku.fi
www.uku.fi
[2] Foster Wheeler Energia Oy,
P.O. Box 201, FIN-78201 Varkaus, Finland
{ari.kettunen, eero.niemitalo}@fwfin.fwc.com
www.fwc.com
[3] Department of Energy and Environmental Technology,
Lappeenranta University of Technology,
P.O. Box 20, FIN-53851 Lappeenranta, Finland
reijo.kuivalainen@lut.fi
www.lut.fi

Abstract. Self-organizing maps (SOM) have been successfully applied in many fields of research. In this paper, we demonstrate the use of SOM-based method for process state monitoring and optimization of NOx emissions. The SOM was trained using a dataset from a fluidized bed energy plant. Reference vectors of the SOM were then classified by K-means algorithm into five clusters, which represented different states of the process. One neuron in each cluster was defined optimal based on the NOx emission of the process. The difference between reference vectors of the optimal neuron and the neuron in each time step could be used for determination of reasons of non-optimal process states. The results show that the SOM method may also be successfully applied to process state monitoring and optimization of NOx emissions.

1 Introduction

New environmental legislation sets need to improve plant availability with increasing efficiency in electricity production. One of the main issues in combustion of fossil fuels is the minimization of emission. Due these reasons, the need for advanced monitoring and analyzing systems increases even those new combustion technologies provide environmental sound solutions for utilization of variety of fuels.

Archived process data is an important resource for the knowledge management of the process and it can be used for the optimization and improvement of productivity. Recent applications have demonstrated that artificial neural networks can provide an efficient and highly automated method for modeling industrial data [1, 2]. In particular, studies, which use standardized protocols, are most likely to benefit from

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 409 – 414, 2005.

automated ANN analysis [1, 2]. Self-organizing maps [1, 3-5] have been also successfully applied in many areas of research and are thus a tool for process optimization. The SOM method offers an efficient means of handling complex multidimensional data, which is typically the situation in industrial applications.

In this study we have constructed a self-organizing map by a data set of a fluidized bed energy plant and then used it for process state monitoring and optimization of NOx emissions.

2 The Process and the Data

The main components of a typical circulating fluidized bed (CFB) boiler are shown in Figure 1. These consist of a combustion chamber, a separator and a return leg for recirculation of the bed particles.

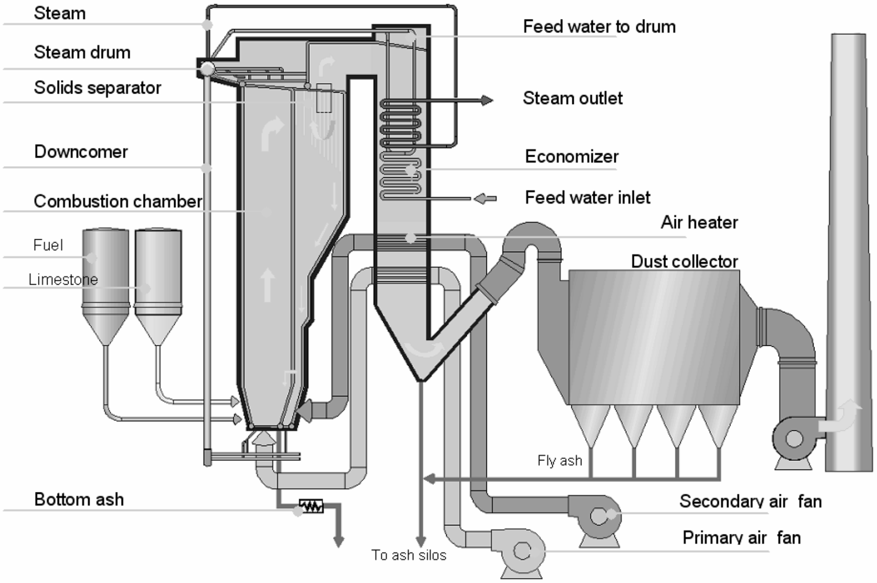


Fig. 1. Foster Wheeler's Compact CFB boiler with a centrifugal separator joined to the combustion chamber without expansion joints. The separator is fabricated with flat walls constructed from conventional water-cooled membrane panels and covered with a thin refractory lining.

Combustion takes place in fluidized bed, which is typically sand mixed with fuel ash and possible sorbent material for sulfur capture. The bed material is fluidized by injecting primary air from the bottom of the combustion chamber. Circulating fluidized bed boilers use high fluidizing velocities, so the particles are constantly held in the flue gases, and pass through the main combustion chamber into a separator, from which the larger particles are extracted and returned to the combustion chamber, while the finer particles are removed from the flue gases by an electrostatic precipitator or baghouse located downstream of the boiler's convection section.

Due to the large heat capacity of the bed, the combustion is stable and supporting fuels such as oil or gas are needed only during start-up. The intense turbulence of the circulating fluidized bed ensures good mixing and combustion of fuel. Combustion typically takes place at about 850 – 900 ^{o}C bed temperature resulting lower NO_X emission compared with conventional boilers. The temperature range is also optimal for use of sorbent material for sulfur retention. From the view point of emission optimization formation of flue gas emissions is a complex process and it is affected by number of process variables.

The raw data used in this study was extracted from databases of utility scale CFB boiler, which uses lignite as main fuel and limestone for sulfur capture. The time resolution of the data set was 15 minutes. The size of complete data matrix was 10 000 x 46 (10 000 rows, 46 variables in columns). Variables were selected by process experts.

3 Computational Methods

3.1 SOM

Self-organizing maps (SOMs) are an artificial neural network methodology, which can transform an n-dimensional input vector into a one- or two-dimensional discrete map. The input vectors, which have common features, are projected to the same area of the map e.g. (in this case described as "neurons"). Each neuron is associated with an *n*-dimensional reference vector, which provides a link between the output and input spaces. During learning, the input data vector is mapped onto a particular neuron (best matching unit, BMU) based on the minimal *n*-dimensional distance between the input vector and the reference vectors of the neurons. Then the reference vectors of the activated neurons are updated. When the trained map is applied, the best matching units are calculated using these reference vectors. In this unsupervised methodology, the SOM can be constructed without previous *a priori* knowledge [1].

The data were coded into 46 inputs for the SOM. All input values were variance scaled. The SOM having 256 neurons in a 16x16 hexagonal arrangement was constructed. The linear initialization and batch training algorithms were used in the training of the map. A Gaussian function was used as the neighborhood function. The map was taught with 10 epochs and the initial neighborhood had the value of 3. The SOM Toolbox [7] was used in the analysis under a Matlab-software platform (Mathworks, Natick, MA, USA).

3.2 K-Means

The K-means algorithm was applied to the clustering of the map. The K-means method is a well-known non-hierarchical cluster algorithm [6]. The basic version begins by randomly picking *K* cluster centers, assigning each point to the cluster whose mean is closest in a Euclidean distances sense, then computing the mean vectors of the points assigned to each cluster, and using these as new centers in an iterative approach.

3.3 Optimization and Subtraction Analysis

For each neuron the reference vectors, which represent the common features of the data in each neuron, are defined during the training of the map. Therefore the components of the reference vectors vary in different parts of the map. An optimal neuron for the whole map or a cluster can be determined using one or more components of the reference vectors. In this paper, we were interested in control of the NOx concentration, thus the optimal neurons of each cluster were simply the neurons, where the NOx component of the reference vectors was smallest.

In the subtraction analysis, reference vectors of two neurons are subtracted from each other. This method can be used for identification of any differences in factors between corresponding subgroups of two neurons. So the difference between reference vectors of the optimal neuron and the best matching unit, when for example in this case the NOx concentration is high, can indicate the reasons for high emission.

4 Results and Discussion

The SOM was obtained by training a self-organizing network with the data of a fluidized bed energy plant. The map and the five clusters calculated by the K-means method are shown in Figure 2. These clusters represent different states of the process. The brief descriptions of the clusters are also illustrated in Figure 2.

An optimal neuron for each cluster was determined based on information of the NOx emission, i.e. the neuron, where the NOx emission was smallest, was specified in each cluster. Figure 2 shows also these optimal neurons.

The difference between reference vectors of the optimal neuron and the neuron in each time step could be used for determination of reasons of non-optimal process

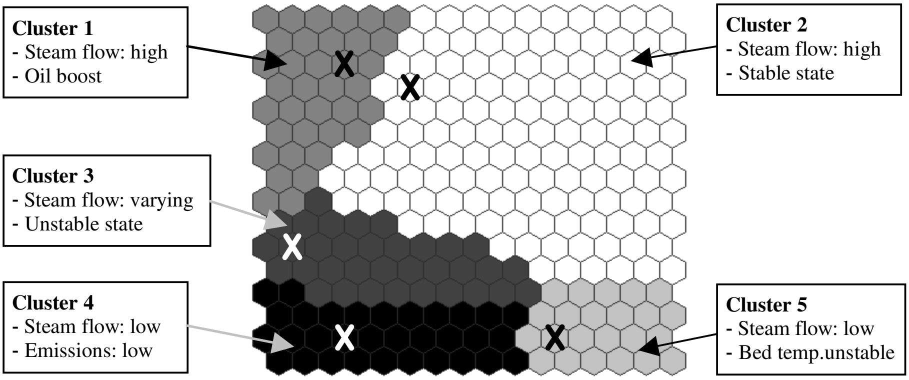
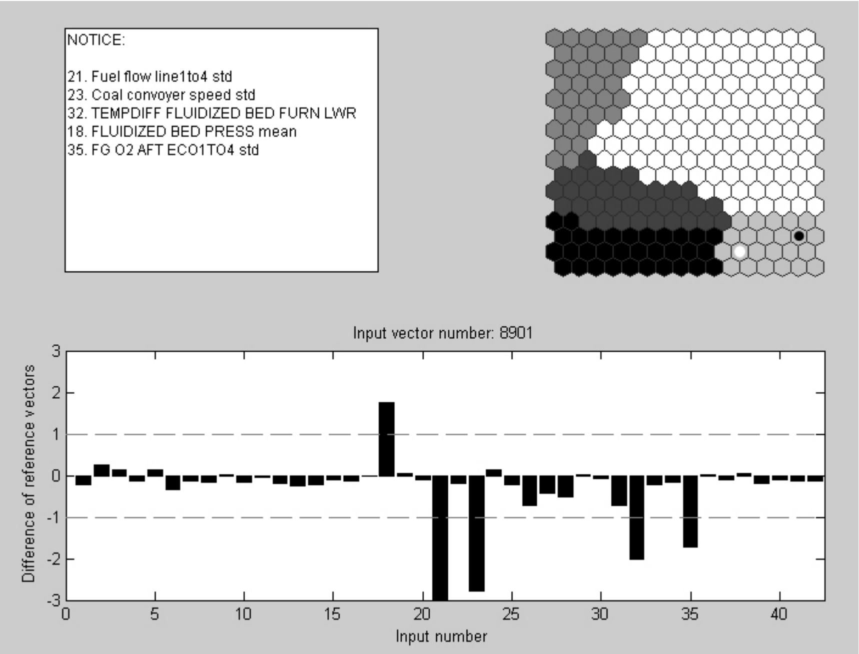


Fig. 2. SOM using the data of a fluidized bed energy plant. The background colors visualize the five clusters of the map. Short descriptions for each cluster are also shown. The optimal neurons in each cluster are marked by X.



NOTICE:

21. Fuel flow line1to4 std
23. Coal convoyer speed std
32. TEMPDIFF FLUIDIZED BED FURN LWR
18. FLUIDIZED BED PRESS mean
35. FG O2 AFT ECO1TO4 std

Fig. 3. The interface of the prototype system. (Upper left corner) Possible problems of the process transformed into text format. (Upper right corner) The optimal neuron (white mark) and the best matching unit of an input vector (black mark) visualized on the SOM. (Below) Difference of reference vectors of these two neurons.

states. A prototype system was made under a Matlab-software platform, which exploits the SOM method and also these differences between reference vectors. Figure 3 illustrates the interface of this prototype system, which contains three parts: (1) The optimal neuron and the best matching unit of an input vector are visualized on the SOM, (2) The differences between reference vectors are shown by bar graph representation and (3) the information of these differences is transformed into text format showing possible reasons for problems in the process.

5 Conclusion

Our SOM-based method seems to have several benefits: (i) it is easy to use and to be included in any kinds of applications, (ii) it does not need much processor and memory capacity, and (iii) it is quite generic and automatic, which means that it can be applied to optimization using almost any type of optimization or cost function.

The SOM analysis provides an efficient and automated method for data analysis in the process industry. The present study shows that this kind of data-driven approach is a fruitful way of developing the process state monitoring and optimization in an energy plant.

References

1. Kohonen, T.: Self-organizing Maps. Springer-Verlag, Berlin Heidelberg New York, (2001)
2. Haykin, S., Neural Networks: A Comprehensive Foundation, Upper Saddle River, NJ: Prentice Hall, (1999)
3. Kaartinen, J., Hiltunen, Y., Kovanen, P. T., Ala-Korpela, M.: Classification of Human Blood Plasma Lipid Abnormalities by 1H Magnetic Resonance Spectroscopy and Self-Organizing Maps. NMR Biomed. 11 (1998) 168-176
4. Hyvönen, M. T., Hiltunen, Y., El-Deredy, W., Ojala, T., Vaara, J., Kovanen, P. T., Ala-Korpela, M.: Application of Self-Organizing Maps in Conformational Analysis of Lipids. Journal of the American Chemical Society, Vol. 123. (2001) 810-816
5. Heikkinen, M., Kolehmainen, M., Hiltunen, Y.: Classification of process phases using Self-Organizing Maps and Sammon's mapping for investigating activated sludge treatment plant in a pulp mill. Proceedings of the Fourth European Symposium on Intelligent Technologies and their implementation on Smart Adaptive Systems. (2004) 281-297
6. MacQueen. Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Volume I: Statistics. University of California Press, Berkeley and Los Angeles (1967) 281-297
7. Homepage of SOM toolbox, http://www.cis.hut.fi/projects/somtoolbox/

A New Extension of Self-optimizing Neural Networks for Topology Optimization

Adrian Horzyk

University of Science and Technology, Department of Automatics,
Mickiewicza Av. 30, 30-059 Cracow, Poland
horzyk@agh.edu.pl

Abstract. The paper introduces a new extension of the ontogenic Self-Optimizing Neural Networks (SONNs) [4] making possible to optimize a neural network (NN) topology for a whole training data (TR) set at once. The classical SONNs optimize a NN topology only for subnetworks related to trained classes. The described SONN extension enables to optimize topology for all classes at once. Moreover, this extension makes possible to compute a minimal SONN topology for given TD which can be sometimes insufficient in view of generalization. The SONN extension computes better discrimination coefficients and automatically develops the topology that reflects all well-discriminative data features into the NN topology in order to achieve a good generalization property. Furthermore, the SONN extension can also automatically reduce the input dimension space of any TD and automatically recognize and correctly classify inverted inputs (especially important for image classification). All extended SONN computations are fully automatic and deterministic. There is no need to use any parameters given by user. The SONNs are free from many training problems, e.i. initiation, convergence, overfitting. The extended SONNs can be also used to unsupervised training.

1 Introduction

Artificial neural networks (ANNs) are very modern computational tool used to solve many difficult problems. However, many types of ANNs are known today it is still necessary to develop new better adjustable and faster adaptable NNs. Today, an important group of neural networks (NNs) that are able to adapt their structure and parameters to different problems establishes ontogenic NNs. The described ontogenic Self-Optimizing Neural Networks (SONNs) develop the NN topology and adjust the weights in the self-adapting process. The SONN adaptation process demands all TD to be known and accessible before the adaptation process is initiated. The adaptation process can be very fast thanks this feature. The SONNs adapt their structure and adjust the weights to given TD. The ANN technology can sometimes lead to produce specialized NNs which can even better or more effective solve some problems than NNNs [2,4,5]. The SONNs gradually project the most discriminative and well-differentiating features of the TD into the structure and parameters (weights) of the NN. The classical SONNs develop

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 415–420, 2005.

k subnetworks for k trained classes using k-classificator methodology. The extended SONNs build up a single NN for all trained classes. Such a NN is even smaller and more effective than the classical one. This way of SONN adapting is very similar the way of the NNNs remembering and classification. The extended SONN development stops the process of adaptation exactly when all TD are correctly classified. The described SONN extension makes possible to compute even minimal NN topology that correctly classifies all TD. Such minimal SONN topology can be often insufficient for satisfactory generalization because there is no redundant connections that can sometimes help to classify better especially the noised and corrupted inputs. Moreover, the SONNs automatically reduce the input dimension space of the TD. Furthermore, the SONNs automatically recognize inversion of the input TD, classifying the inverted data correctly and returning information about the inversion of the input data. The SONN adaptation process can be continued even after all TD are correctly classified increasing the SONN knowledge about TD details. The SONN adaptation process is deterministic and free from many training problems: initial topology and parameters establishment, redundant or insufficient NN structure, local minima and conversion, stopping condition, overfitting etc. [1,3,4].

2 Mathematical Background of SONN Extension

The ontogenic Self-Optimizing Neural Networks (SONNs) develop irregular multilayer partially connected NN topology with rare inter- and supralayer connections (Fig. 1). The extended SONNs build up a single better-optimized topology for all classes and all TD in comparison to the classical SONNs [4] that construct separate subnetworks for each class. The SONNs are created in a complex optimizing process after the statistical analysis of the TD and probabilistic estimation of all TD features taking into account quantity of training samples (TSs) representing individual classes. The SONNs use the TD $U = \left\{\left(u^1, C^{m_1}\right), \ldots, \left(u^N, C^{m_N}\right)\right\}$ consisting of input vectors $u^n = [u_1^n, \ldots, u_K^n]$ (where features $u_k^n \in \{-1, 0, +1\}$, continuous features have to be quantified) and the adequate class $C^{m_n} \in \left\{C^1, \ldots, C^M\right\}$. The relevance of the features *1, ..., K* for any input vector $u^n \in C^{m_n}$ can be estimated in view of any given TD set. The SONN adaptation and optimization processes use the discrimination coefficient $d_k^n \in [0;1]$ that defines how well the feature k discriminate between the training sample n of the class m and the training samples of the other classes [4] $\forall_{m\in\{1,\ldots,M\}}\ \forall_{u^n\in C^m}\ \forall_{n\in\{1,\ldots,Q\}}\ \forall_{k\in\{1,\ldots,K\}}$:

$$d_k^n = \begin{cases} \frac{\hat{P}_k^m}{(M-1)\cdot Q^m} \sum\limits_{h=1\&h\neq m}^{M} \left(1-\frac{\hat{P}_k^h}{Q^h}\right) & if\ u_k^n = +1 \\ 0 & if\ u_k^n = 0 \\ \frac{\hat{N}_k^m}{(M-1)\cdot Q^m} \sum\limits_{h=1\&h\neq m}^{M} \left(1-\frac{\hat{N}_k^h}{Q^h}\right) & if\ u_k^n = -1 \end{cases} \tag{1}$$

$$\forall_{m\in\{1,\ldots,M\},k\in\{1,\ldots,K\}}\ \hat{P}_k^m = \sum_{u^n\in C^m} x_k^n \quad \wedge \quad \hat{N}_k^m = \sum_{u^n\in C^m} y_k^n$$

$$\forall_{m\in\{1,...,M\}}\ Q^m = \|\{u^n \in U \vee C^m \& n \in \{1,...,Q\}\}\|$$

$$x_k^n = \begin{cases} 1 & if\, u_k^n = +1 \\ \frac{P_k^m}{P_k^m+N_k^m} & if\, u_k^n = 0 \\ 0 & if\, u_k^n = -1 \end{cases} \qquad y_k^n = \begin{cases} 1 & if\, u_k^n = -1 \\ \frac{N_k^m}{P_k^m+N_k^m} & if\, u_k^n = 0 \\ 0 & if\, u_k^n = +1 \end{cases}$$

$$\forall_{m\in\{1,...,M\},k\in\{1,...,K\}}\ P_k^m = \sum_{u_k^n\in\left\{u^n\in U\vee C^m: u_k^n>0\wedge n\in\{1,...,Q\}\right\}} u_k^n$$

$$\forall_{m\in\{1,...,M\},k\in\{1,...,K\}}\ N_k^m = \sum_{u_k^n\in\left\{u^n\in U\vee C^m: u_k^n<0\wedge n\in\{1,...,Q\}\right\}} -u_k^n$$

The new discrimination coefficient (introduced in this paper) is insensitive for quantitative differences in representation of classes and concerns statistical differences of features quantity after their values in classes and TD. The bigger discrimination coefficient the more discriminative is the feature in view of classification and *vice versa*.

The same features for different TSs can be grouped together in order to save connections between neurons modelling the given TD set. There should be maximized the quantity of grouped features for single connections in order do minimize the size of the NN. The described optimization of NN topology is proceeded using a special kind of table (Tab. 1) that enables to compute the quantity of saved connections in dependence of the chosen divided feature (DF).

The DFs are used to divide the TD set (or some subsets of it in next periods) to two subsets depending on that feature: one subset contains TSs with the true (+1) DF value and the second one with the false (-1) DF value. If TD contain unknown features (0) there is created the third subset of such training samples that are separately divided using another DF selected for this subset. The SONN extension uses the specific tables (Tab. 1) to compute optimal DFs. The tables are filled with the $d_k^n \cdot u_k^n$ (instead of the u_k^n as in classical SONNs) for all TSs n and all not static features k is created. Next, the quantities of $u_k^n < 0$ and $u_k^n > 0$ are computed for all TSs (or for some subsets of TSs in the next periods) for each relevant feature that has not been already used in one of the previous periods. In the following step, the quantities of already not used features of same sign and same value for the same training samples as the considered feature are calculated. The groups of same sign and value features for the exemplar feature 8 are framed and marked bold in the table 1. Each group of bold framed features of the same value and sign can be transformed into single connection of the NN, e.i. $132 - 19 = 113$ for the example in the table 1. The quantity of saved connections for any considered feature is computed as:

$$Save_k = SameNeg_k \cdot (QuantNeg_k - 1) + SamePos_k \cdot (QuantPos_k - 1) \quad (2)$$

$$DF = \max_{k=1,...,K} \{Save_k\} \quad (3)$$

Next, there is chosen the divided features DF as a feature that maximally saves connections. If there are several features with the same maximal saving property

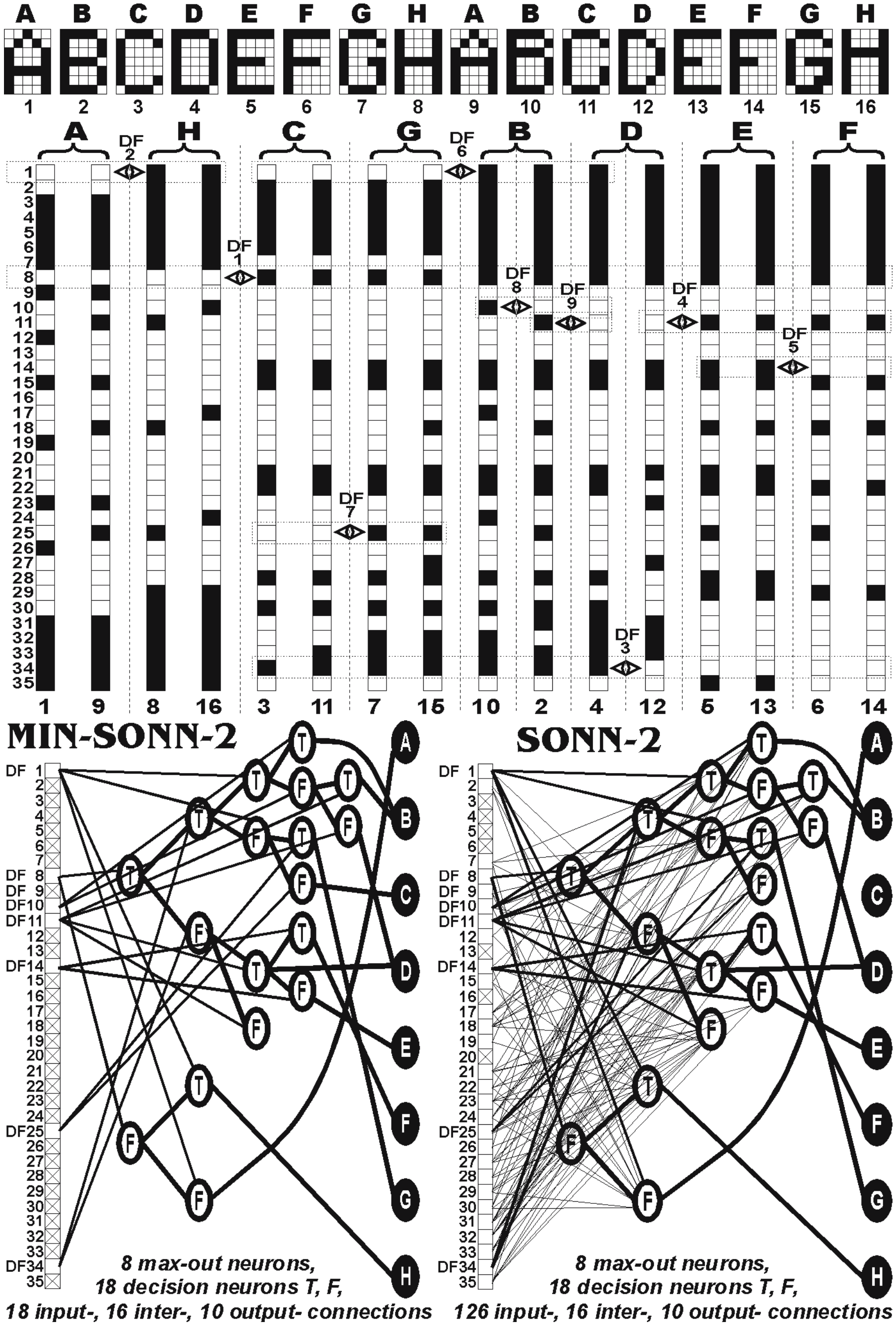
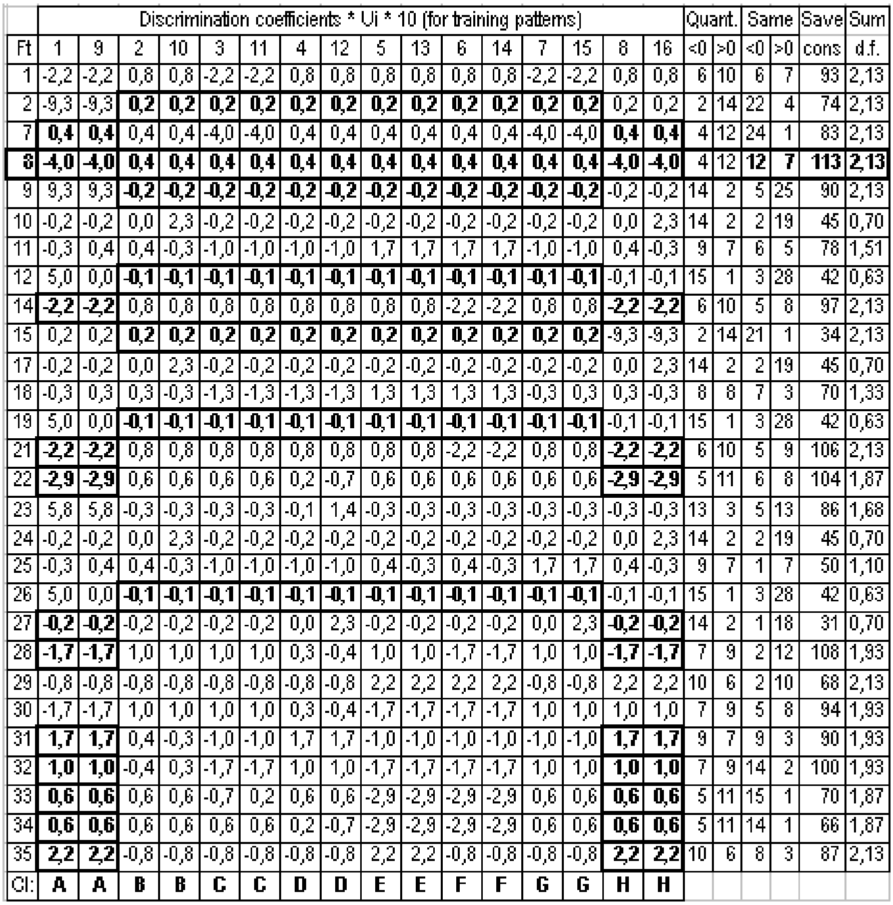


Fig. 1. The extended SONN development process shown on the exemplar data set and the selected divided features (DFs)

Table 1. Divided feature (DF) computation for the exemplar data set from Fig. 1



	Discrimination coefficients * Ui * 10 (for training patterns)																Quant.		Same		Save	Sum
Ft	1	9	2	10	3	11	4	12	5	13	6	14	7	15	8	16	<0	>0	<0	>0	cons	d.f.
1	-2,2	-2,2	0,8	0,8	-2,2	-2,2	0,8	0,8	0,8	0,8	0,8	0,8	-2,2	-2,2	0,8	0,8	6	10	6	7	93	2,13
2	-9,3	-9,3	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	0,2	0,2	2	14	22	4	74	2,13
7	**0,4**	**0,4**	0,4	0,4	-4,0	-4,0	0,4	0,4	0,4	0,4	0,4	0,4	-4,0	-4,0	**0,4**	**0,4**	4	12	24	1	83	2,13
8	**-4,0**	**-4,0**	**0,4**	**0,4**	**0,4**	**0,4**	**0,4**	**0,4**	**0,4**	**0,4**	**0,4**	**0,4**	**0,4**	**0,4**	**-4,0**	**-4,0**	4	12	**12**	**7**	**113**	**2,13**
9	9,3	9,3	**-0,2**	**-0,2**	**-0,2**	**-0,2**	**-0,2**	**-0,2**	**-0,2**	**-0,2**	**-0,2**	**-0,2**	**-0,2**	**-0,2**	-0,2	-0,2	14	2	5	25	90	2,13
10	-0,2	-0,2	0,0	2,3	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	0,0	2,3	14	2	2	19	45	0,70
11	-0,3	0,4	0,4	-0,3	-1,0	-1,0	-1,0	-1,0	1,7	1,7	1,7	1,7	-1,0	-1,0	0,4	-0,3	9	7	6	5	78	1,51
12	5,0	0,0	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	-0,1	-0,1	15	1	3	28	42	0,63
14	**-2,2**	**-2,2**	0,8	0,8	0,8	0,8	0,8	0,8	0,8	0,8	-2,2	-2,2	0,8	0,8	**-2,2**	**-2,2**	6	10	5	8	97	2,13
15	0,2	0,2	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	**0,2**	-9,3	-9,3	2	14	21	1	34	2,13
17	-0,2	-0,2	0,0	2,3	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	0,0	2,3	14	2	2	19	45	0,70
18	-0,3	0,3	0,3	-0,3	-1,3	-1,3	-1,3	-1,3	1,3	1,3	1,3	1,3	-0,3	0,3	0,3	-0,3	8	8	7	3	70	1,33
19	5,0	0,0	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	-0,1	-0,1	15	1	3	28	42	0,63
21	**-2,2**	**-2,2**	0,8	0,8	0,8	0,8	0,8	0,8	0,8	0,8	-2,2	-2,2	0,8	0,8	**-2,2**	**-2,2**	6	10	5	9	106	2,13
22	**-2,9**	**-2,9**	0,6	0,6	0,6	0,6	0,2	-0,7	0,6	0,6	0,6	0,6	0,6	0,6	**-2,9**	**-2,9**	5	11	6	8	104	1,87
23	5,8	5,8	-0,3	-0,3	-0,3	-0,3	-0,1	1,4	-0,3	-0,3	-0,3	-0,3	-0,3	-0,3	-0,3	-0,3	13	3	5	13	86	1,68
24	-0,2	-0,2	0,0	2,3	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	-0,2	0,0	2,3	14	2	2	19	45	0,70
25	-0,3	0,4	0,4	-0,3	-1,0	-1,0	-1,0	-1,0	0,4	-0,3	0,4	-0,3	1,7	1,7	0,4	-0,3	9	7	1	7	50	1,10
26	5,0	0,0	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	**-0,1**	-0,1	-0,1	15	1	3	28	42	0,63
27	**-0,2**	**-0,2**	-0,2	-0,2	-0,2	-0,2	0,0	2,3	-0,2	-0,2	-0,2	-0,2	0,0	2,3	**-0,2**	**-0,2**	14	2	1	18	31	0,70
28	**-1,7**	**-1,7**	1,0	1,0	1,0	1,0	0,3	-0,4	1,0	1,0	-1,7	-1,7	1,0	1,0	**-1,7**	**-1,7**	7	9	2	12	108	1,93
29	-0,8	-0,8	-0,8	-0,8	-0,8	-0,8	-0,8	-0,8	2,2	2,2	2,2	2,2	-0,8	-0,8	2,2	2,2	10	6	2	10	68	2,13
30	-1,7	-1,7	1,0	1,0	1,0	1,0	0,3	-0,4	-1,7	-1,7	-1,7	-1,7	1,0	1,0	1,0	1,0	7	9	5	8	94	1,93
31	**1,7**	**1,7**	0,4	-0,3	-1,0	-1,0	1,7	1,7	-1,0	-1,0	-1,0	-1,0	-1,0	-1,0	**1,7**	**1,7**	9	7	9	3	90	1,93
32	**1,0**	**1,0**	-0,4	0,3	-1,7	-1,7	1,0	1,0	-1,7	-1,7	-1,7	-1,7	1,0	1,0	**1,0**	**1,0**	7	9	14	2	100	1,93
33	**0,6**	**0,6**	0,6	0,6	-0,7	0,2	0,6	0,6	-2,9	-2,9	-2,9	-2,9	0,6	0,6	**0,6**	**0,6**	5	11	15	1	70	1,87
34	**0,6**	**0,6**	0,6	0,6	0,6	0,6	0,2	-0,7	-2,9	-2,9	-2,9	-2,9	0,6	0,6	**0,6**	**0,6**	5	11	14	1	66	1,87
35	**2,2**	**2,2**	-0,8	-0,8	-0,8	-0,8	-0,8	-0,8	2,2	2,2	-0,8	-0,8	-0,8	-0,8	**2,2**	**2,2**	10	6	8	3	87	2,13
Cl:	**A**	**A**	**B**	**B**	**C**	**C**	**D**	**D**	**E**	**E**	**F**	**F**	**G**	**G**	**H**	**H**						

there is selected this one which sum of discrimination features for the whole set (or subset in the next periods) of considered samples is the biggest. If these sums are equal for several features there can be chosen whichever of them. The selection of DF conclude in division of the given set (or subset) of TSs into two separate subsets as described above. There are created two new neurons for these two subsets. If minimal extended SONN topology is created, there are added only two input connections for the established divided feature DF. In order to build well-generalizing extended SONN topology, there should be created input connections to those two neurons for all (or for any most discriminative subset of) same sign and value framed bold features (shown in the Tab. 1 and the Fig. 1) producing a thrifty redundant NN topology. In the following periods, the subsets of samples are divided into next two subsets as long as the final subsets contain TSs of single classes. This is also the definition of the stopping condition of the algorithm guaranteeing the minimal quantity of periods necessary to develop the topology that correctly classifies all TSs. The described algorithm of TD division can be also used for fast unsupervised clustering of TSs. The described algorithm keeps the mostly similar TSs together as long as possible. Such division of samples enables also to watch the pathes of processing data in the NN and looking back for rules. The paper [4] describes the way of weights and the neuron outputs computation that do not change for SONN extension.

3 Conclusions

The described extended SONNs methodology for simultaneous gradual deterministic configuration and adaptation of NNs builds up the topology optimized for the given TD and computes optimal weights for them. It does not need to use any *a priori* configuration parameters and is free from different training problems. The SONNs always correctly classify all TD and offer very good generalization properties, because it is created after the most representative and discriminating TD features established after the new discrimination coefficients introduced in this paper. The extended SONN topology is even smaller than the classical one because the topology is optimized for all TD classes together instead of for each TD class separately. Moreover, the SONN extension is able to create even a minimal SONN structure though it can be sometimes insufficient for generalization. The described extended SONN algorithm normally develops a thrifty redundant topology in order to achieve better discrimination properties. The SONNs can also automatically recognize and classify inverse samples of the defined classes in the TD set. The extended SONNs can also automatically reduce the input data dimension even more than classical SONNs. Finally, the extended SONN topology size, the generalization property and fast computation make SONNs very attractive to use in comparison to other AI methods.

Acknowledgements. Support from research funds 11.11.120.113 and the Polish Committee for Scientific Research is gratefully acknowledged.

References

1. Duch, W., Korbicz, J., Rutkowski, L., Tadeusiewicz, R. (eds): Biocybernetics and Biomedical Engineering. Neural Networks, EXIT Warsaw **6** (2000) 257–566
2. Dudek-Dyduch, E., Horzyk, A.: Analytical Synthesis of Neural Networks for Selected Classes of Problems. Knowlege Engineering and Experts Systems, Bubnicki Z., Grzech A. (eds), OWPN Wroclaw (2003) 194–206
3. Fiesler, E., Beale, R. (eds): Handbook of Neural Computation. IOP Publishing Ltd and Oxford University Press, Bristol & New York (1997) B3–C1
4. Horzyk, A., Tadeusiewicz, R.: Self-Optimizing Neural Networks. Advances in Neural Networks - ISNN 2004, Yin F., Wang J., Guo C. (eds), Springer-Verlag Berlin Heidelberg (2004) 150–155
5. Tadeusiewicz, R., Ogiela, M.R.: Machine Perception and Automatic Understanding of Medical Visualization. Automatic Image Processing in Production Process, Damczyk M. (eds), Second Polish-German Seminar, CAMT, Wroclaw (2003) 39–48.

A Novel Technique for Data Visualization Based on SOM

Guanglan Liao, Tielin Shi, Shiyuan Liu, and Jianping Xuan

School of Mechanical Science and Engineering,
Huazhong University of Science and Technology, Wuhan, Hubei 430074, China
{g.l.liao, tlshi, shyliu, jpxuan}@mail.hust.edu.cn

Abstract. When used for visualization of high dimensional data, the self-organizing maps (SOM) requires a coloring scheme, or interpolation, or applying some projection techniques to analyze the intrinsic structure of the data. Even so, the structures of the data clusters may not be apparent and their shapes are often distorted. In order to overcome some shortcomings of the traditional SOM visualization method a novel technique is presented in this paper. Several experimental data sets including the chain-link problem and IRIS data are used to test the approach. The analysis results prove that the presented technique provides a better picture of the high dimensional data to understand their intrinsic structure.

1 Introduction

Data mining is an emerging area of new research efforts, responding to the presence of large databases in commerce, industry and research. It is a part of a larger framework, knowledge Discovery in Databases, whose purpose is to find new knowledge from databases where dimension, complexity or amount of data is prohibitively large for human observation alone. Data mining is an interactive process requiring that the intuition and background knowledge of humans be coupled with modern computer technology. Therefore, visualization by means of the projection of data from an invisible high-dimensional space to a low perceptible one is very important as it can reveal the data structures and cluster tendency.

Linear projection is one way to map a multidimensional space to a lower-dimensional one realizing dimension reduction, and neural networks, for example, Kohonen's SOM [1], present another approach to nonlinear data analysis. When the SOM is used for visualization, however, as the inter-neuron distances are not directly visible or measurable on the map, one has to use a coloring scheme [2, 3], or interpolation [4], to mark the relative distances between the weights of neighboring neurons referred in the input space, or apply some projection techniques, such as Sammon's projection [5] and the curvilinear component analysis [6]. Even so, the structures of data clusters may not be apparent and often appear distorted. Had the distributions of the data not been known, it would be impossible to depict them from the learnt map.

To overcome some shortcomings of the traditional SOM visualization method a novel technique is proposed. Several experimental data sets including a synthetic data randomly distributed in three-dimensional space, the so-called chain-link problem [2]

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 421 – 426, 2005.

and the well-known IRIS data [7], are used to test the approach. The analysis results demonstrate that the novel technique may offer attractive advantages over the commonly used SOM visualization techniques in giving a better picture of the high dimensional data to understand their intrinsic structure.

2 Description of the Novel SOM Visualization Technique

SOM does not directly show the inter-neuron distances on the map when used for visualization of potentially high dimensional data. In order to visualize the data more apparently, a novel SOM visualization technique for high dimensional projection and structure visualization is introduced in the following.

Suppose there are *m* neurons in total arranged in a 2-dimensional sheet with hexagonal (or rectangular) shape in the SOM network. The coordinates of neurons *j* in the 2-dimensional sheet are denoted by $\boldsymbol{x}_j$, j=1,2,…,m, and the corresponding *n*-dimensional synaptic weights $\boldsymbol{w}_j=(w_{j1}, w_{j2}, \ldots, w_{jn})^T$. To project the *n*-dimensional input vectors, say $\boldsymbol{p}=(p_1,p_2,\ldots,p_n)^T$, into a 2-dimensional space for visualization, instead of simply pointing out the Best Matching Units (BMUs), the responses of all neurons for the input data must be obtained. As the Euclidean distance metric is usually adopted to be the discriminant function during the competition in the SOM network, a simple approach is to use the quantization error, $q_j=\|\boldsymbol{p}-\boldsymbol{w}_j\|$, j=1,2,…,m, as the indicator of the responses of the output neurons *j* for $\boldsymbol{p}$ evaluated in the form

$$R_j(\boldsymbol{p}) = \frac{g(q_j)N_c(j)}{\sum_{i=1}^{m} g(q_i)N_c(i)},\ j=1,2,\ldots,m. \tag{1}$$

Here *c* is the BMU of $\boldsymbol{p}$, and $N_c(j)$ a neighboring function decreasing with the distance between the BMU and the output neurons *j*.

Since the responses of output neurons for $\boldsymbol{p}$ are related closely with the Euclidean distances between the neurons weights and $\boldsymbol{p}$, where the further the weights are away from $\boldsymbol{p}$, the less responses the neurons have, it is appropriate to choose $g(q_j)$ to be a bounded function monotonically decreasing with q_j. Usually, the decreasing exponential, sigmoid, or Lorentz functions are all suitable choices. Here the decreasing exponential function is used, $g(q_j)=\exp(-q_j)$. Notice that the function has several beneficial properties: g(0)=1, $g(q_j \to \infty)=0$.

Then the coefficients, given by the responses $R_j(\boldsymbol{p})$, can be used to provide a visualization of the trained SOM results in the 2-dimensional space by summarizing the responses by the mean. And the image point for the input vector $\boldsymbol{p}$ in the 2-dimensional space, $\boldsymbol{x}_{\boldsymbol{p}}^{mean}$, can be performed by

$$\boldsymbol{x}_{\boldsymbol{p}}^{mean} = \sum_{j=1}^{m} R_j(\boldsymbol{p})\boldsymbol{x}_j\,. \tag{2}$$

The mode of $\boldsymbol{p}$ is then to be evaluated by

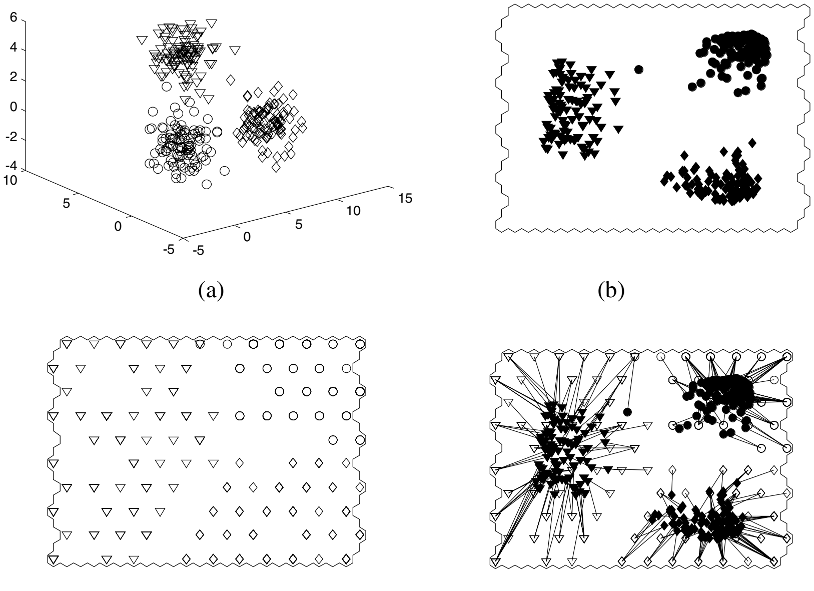
$$\boldsymbol{x}_{\boldsymbol{p}}^{mode} = \arg\max_{\boldsymbol{x}_j}\{R_j(\boldsymbol{p})\},\ j=1,2,\ldots,m. \tag{3}$$

Actually, the mode of the vector ***p*** in the 2-dimensional space gives the position of the BMU for the vector in the learnt map.

3 Experimental Results

To demonstrate the application of the proposed visualization technique and its distance-preserving and structure revealing properties, several experiments and results were presented. The examples chosen were for the purpose of illustration. The results of the traditional SOM mapping technique were also presented for comparison.

Results Using Synthetic Data Set. A synthetic data set was constructed to demonstrate the performance of the novel technique in visualizing the SOM learnt results. In order to observe the original distribution clearly in the input space only three-dimensional randomly distributed data represented with "*circle*", "*triangle*" and "*diamond*" respectively were selected as inputs for SOM training as shown in Fig.1(a) yet higher dimensions were not used, as they were hard to understand and represent. Obviously the original input data were classified into 3 classes. A 10×12 hexagonal



(a)

(b)

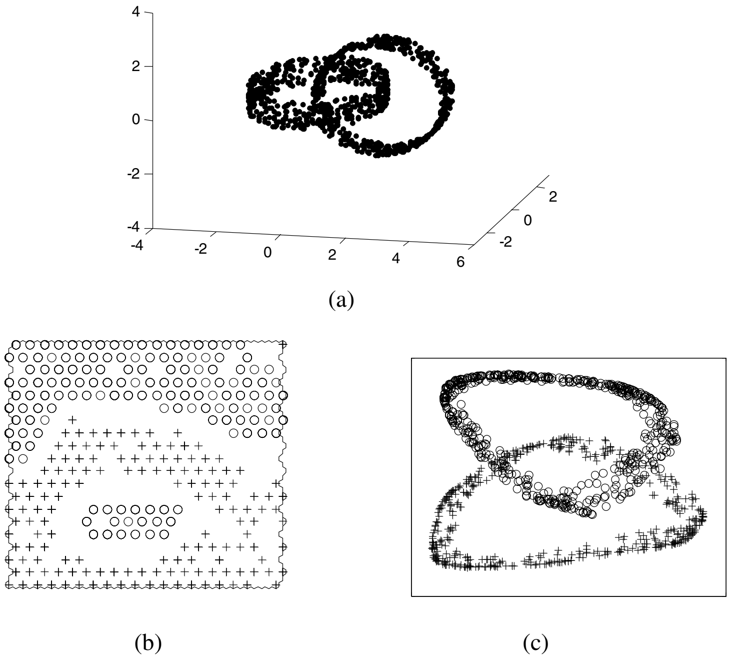
(c)

(d)

Fig. 1. Experimental results using 3-dimensional synthetic data set represented with "*circle*", "*triangle*" and "*diamond*" respectively (a), visualized with the novel technique (b), the traditional mapping technique (c), and combination of the novel technique and traditional mapping method (d). The "*circle*", "*triangle*" and "*diamond*" represented the modes of the input data in the output map, and solid "circle", "triangle" and "diamond" represented image points, each mode linking to image point correspondingly in (d).

SOM had been applied to the data set, and the novel technique was used to visualize the projected data on the 2-dimensional space. The result was shown in Fig. 1(b), where the solid "*circle*", "*triangle*" and "*diamond*" signs denoted the image points for the input data accordingly. For comparison, traditional SOM visualization result, i.e. the BMUs for the input data on the output map, was shown in Fig. 1(c), in which the BMUs were labeled with "*circle*", "*triangle*" and "*diamond*" correspondingly. Fig. 1(d) showed the SOM visualization result of the novel technique combining with the traditional mapping method, in which each BMU was linking to image point correspondingly. It can be found that the image points in the figures "*agglomerate*" the dispersive modes of the input data on the output map, and the novel technique has some advantages over the traditional SOM mapping technique on visualization, as it can provide a fairly good and clear separation of the three class on the map. Meanwhile, the technique presented here will also save much computing time compared with other projection techniques such as Sammon's projection or the curvilinear component analysis, as there is no need of iterative solutions.

The chain-link problem. Additionally, the so-called chain-link problem was used to test the approach, which consisted of two intertwined three-dimensional rings. One of the rings was extended into x-y direction and the other one into x-z direction. Each of the rings consisted of 500 data points. A scatter plot of the data was given in Fig. 2(a). A 20×20 hexagonal SOM network was applied to the data set, and the visualization



(a)

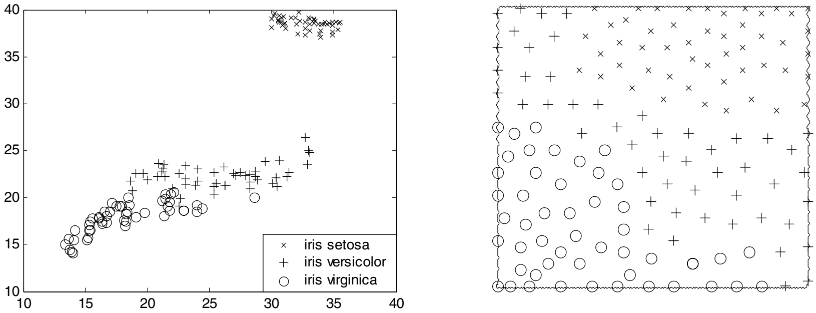
(b)

(c)

Fig. 2. Experimental results of the chain-link problem. Scatter plot of the chain-link problem (a), visualization result of the traditional mapping technique (b) and the novel technique (c).

result of the novel technique was shown in Fig. 2(b). For comparison the result of the traditional mapping method, i.e. BMUs on the map, was shown in Fig. 2(c). From the results, it can be found that whereas the two rings are still visible in the case of the novel technique, no such structure can be found in the output map of the traditional visualization result. Therefore, compared with the traditional SOM mapping method, the novel visualization technique reflects the intrinsic structure of the input data more clearly.

The IRIS dataset. The well-known Fisher's IRIS data set [7] consisted of 4 measurements taken from 150 iris plants. Each plant is from one of 3 species of iris, *setosa*, *versicolor*, and *virginica*. The 4 measurements from each plant were treated as points in 4-dimensional space. Many data visualization experiments have used this data set as a benchmark. In this paper a 50×50 hexagonal SOM had been applied to the data set, and the novel technique was used to visualize the trained results as shown in Fig. 3(a). For comparison, traditional SOM visualization result was shown in Fig.3(b). It can be seen from the figures that the novel technique is better than the traditional mapping method for providing a fairly good separation of the three species on the map, as it captures more details of intra-cluster and inter-point distribution.



(a)

(b)

Fig. 3. Visualization result of IRIS data using the novel technique (a), and the traditional SOM mapping method

4 Conclusions

A novel technique for data visualization based on SOM is proposed in this paper. The technique preserves the inter-point distances of high-dimensional data as well as the topology, in which the structures and cluster tendency of the data may not be apparent and their shapes are often distorted, thus projects the input data into a two-dimensional space. It has been demonstrated that the technique is easy to implement and may offer several attractive advantages over the commonly used SOM visualization techniques, in providing a better picture of the high dimensional data to understand their intrinsic structure.

Acknowledgements

The authors would like to acknowledge the financial support provided by the National Key Basic Research Special Foundation of China (Grant 2003CB716207) and the National Natural Science Foundation of China (Grant 50205009 and 50405033).

References

1. Kohonen, T. Self-Organized Formation of Topologically Correct Feature Maps. Biological Cybernetics.43 (1982) 59–69
2. Ultsch, A., Vetter, C. Self-Organizing-Feature-Maps versus Statistical Clustering Methods: A benchmark. University of Marburg, FG Neuroinformatik & Kuenstliche Intelligenz, Research Report 0994 (1994)
3. Johan, H. A SOM based cluster visualization and its application for false coloring. Proc. IJCNN2000, Como, Italy, Vol. 3. (2000)587–592
4. Yin, H., Allinson, N.M. Interpolating self-organizing map (iSOM). Electronics Letters. Vol. 35. (1999) 1649–1650
5. Sammon, J.W. 1969. A nonlinear mapping for data analysis. IEEE Transactions on Computers. C-18 (1969) 401–409
6. Demartines, P., Hérault, J. Curvilinear Component Analysis: A Self-Organizing Neural Network for Nonlinear Mapping of Data Sets. IEEE Transaction on Neural Networks Vol. 8. (1997) 148–154
7. Fisher, R.A. The use of multiple measurements in taxonomic problems, Annual Eugenics. Vol.7. (1936) 178–188

Statistical Properties of Lattices Affect Topographic Error in Self-organizing Maps

Antonio Neme[1,2] and Pedro Miramontes[2]

[1] Universidad Autónoma la Ciudad de México, San Lorenzo 290, México, D.F
neme@nolineal.org.mx
http://www.nolineal.org.mx/∼neme/som.html
[2] Universidad Nacional Autónoma de México, Cd Universitaria, México, D.F

Abstract. Self-organizing maps (SOM) have been obtained mainly on regular lattices, embedded in euclidean or non-euclidean spaces [1]. We present preliminar results that show SOM can be formed on non-regular lattices by giving evidence that topographic error (TE) is influenced by a few statistical parameters of the neuron lattice, such as the characteristic path length (L), the cluster coefficient (C) and the characteristic connectivity length (Lg). TE is lower not in regular lattices, but in lattices that present a particular set of statistical parameters. In an attempt to identify that set of statistical parameters, we applied mutual information function between the parameters and the TE as well as C4.5 algorithm to obtain rules that identify lattices in which SOMs show low TE.

1 Introduction

Self-organizing map (SOM) is presented as a model of the self-organization of neural connections, what is translated in the ability of the algorithm to produce organization from disorder [2]. One of the main properties of SOM is its ability to preserve topographical relations present in input data in the output map [3].

The principal goal of the SOM is to transform an incoming signal pattern (input data) of arbitrary dimension into a low-dimensional discrete map (usually of dimension one or two) and to adaptively transform data in a topologically ordered fashion [4,5]. Each input data is mapped to a single neuron in the lattice, that with the closest weight vector to the input data. The SOM preserve relationships during training through the neighbourhood function, which stablishes the effect of the winner neuron to any other neuron. Weight neurons are updated accordingly to:

$$w_n(t+1) = w_n(t) + \alpha_n h_t(g,n)(x_i - w_n(t))$$

Where α is the learning rate and $h_t(g,n)$ is the neighbourhood function from winner neuron g to neuron n. Neighbourhood function decreases monotonically as a function of the distance from neuron g to neuron n. This has been reported to be a necessary condition for convergence [6,7].The SOM tries to preserve relationships of input data by starting with a large neighbourhood and reducing the neighbourhood size during the course of training [5].

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 427–432, 2005.

As pointed out by Ritter [1], SOM and related algorithms (ASSOM [8], GTM [9], and others) share the idea of using a deformable lattice to transfrom data similarities into spatial relationships and where topological structure of that lattice plays a main role in the generated mappings. As an alternative to obtain a map that reflects in a better way the structure of multidimensional data (i.e. hierarchical structures), non-euclidean SOMs have been propossed [1,10]. There is also a biological reason for studying SOM on nonregular lattices. Ayali et al [19] have used *in vitro* neural networks as a model for studying self-organization proceses in nervous system and they have found the networks obtained fall into the category of non-regular networks known as small-world.

To measure topological preservation, several metrics have been proposed. Bauer et al [11] and Kiviluoto [12] give a detailed exploration of several topological preservation measures. The metric applied in this work is the topographic error (TE), defined as the proportion of data points which the closest and second-closest weight neurons are not adjacent on the neuron lattice [13]

$$\epsilon_t = 1/N \sum_{i=1}^{N} \eta(x_i, w_i, ..., w_p), \text{ where } \eta(x_i, w_i, ..., w_p) =$$

$$= \begin{cases} 1, & \text{if } \forall l \exists j,k : l \in \{1, ..., j-1, j+1, ..., k-1, k+1, ...p\} \\ & \|m_j - x_i\| \leq \|m_k - x_i\| < \|m_l - x_i\|, |j-k| > 1 \\ 0, & \text{otherwise} \end{cases}$$

Lattices in which SOMs are formed are almost always regular. By regular, we mean that neurons are connected only to neighbours and that every neuron has approximately the same number of connecting edges (same degree). In the present work, we form SOMs on non-regular lattices. In other words, the regular topology of lattices is changed by deleting edges, adding edges between non-neighbor neurons and replacing edges. Fig. 1a shows a non-regular lattice. We observed that TE is closely related to some statistical properties of lattices.

2 Statistical Properties of Lattices

Given a neuron lattice (in general, a network or a graph), a set of statistical properties can be obtained from it. In this work, we obtained five parameters. The first one is the characteristic path length, L. It is the average distance from each neuron to any other, in terms of the number of edges from one neuron to other. It is defined as follows:

$$L(G) = \sum_{\{u,v\} \subset N, u \neq v} d(u,v) / \binom{n}{2}$$

Where N is the set of neurons in the lattice and n is the number of neurons. $d(u,v)$ is the number of edges that separate neuron u from neuron v [14]. In general, $L(G) \leq diam(G)$, where $diam(G)$ is the usual diameter of a network.

The second parameter is the clusterimg coefficient, C, that represents the proportion of the acquaintances of a neuron that know each other (the proportion of each neuron neighbour that are in fact neighbours among them). The local clustering coefficient at neuron v is:

$$C_v(G) = \frac{N_e}{k_n(k_n-1)/2}$$

where N_e is the number of edges between neighbours of neuron N and $k_n(k_n-1)/2$ is the maximun number of edges in a network of k_n nodes [14,15]. $C(G)$ is definded by:

$$C(G) = \sum_{v=1}^{n} C_v(G)/n$$

The third parameter is the normalized neuron degree, O, obtained as the neuron degree divided by the number of nodes in the network. The fourth parameter is the neuron characteristic edge length and the fifth parameter is a clustering coefficient, defined as [15]:

$$C_2(G) = \left(\sum_{v=1}^{n} \frac{N_e}{k_n(k_n-1)/2} C_v(G)\right) / \left(\sum_{v=1}^{n} \frac{N_e}{k_n(k_n-1)/2}\right)$$

When a regular lattice is modified by removing some edges or replacing edges between neighbours by edges between non-neighbour neurons (in the sense of euclidean or normal metric), statistical properties are changed. Fig. 1b shows a regular lattice and its statistical parameters and fig. 1a shows a modified lattice. In fig. 1a, the distance from the neuron in the upper-left corner to the bottom-right corner is 2, because there is a connecting edge between the former and a neighbour (two neurons are neighbours if there is and edge connection them) of the latter. When a few *shortcuts* are added, the parameter L is reduced.

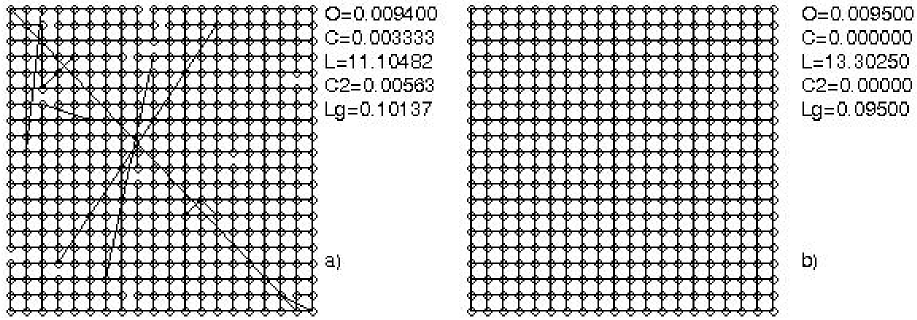


Fig. 1. Regular and a non-regular lattice and their statistical properties as defined in text

3 Results

Three data sets were used: spiral data (a bidimensional spiral), the 64-dimensional codon usage dataset (www.kazusa.or.jp/codon), and a subset from atmospheric conditions in the ionosphere (www.his.hut.fi/research/lvq_pak/), 34−dimensional. 500 lattices of size 20x20 neurons were generated. Some of them were generated following Watts and Strogatz model of *small-world networks* [16], others were generated following the Barabási-Albert model [17], some by a genetic algorithm to achieve a lattice with the desired parameters and some *by hand*. There is a TE error for each lattice and data set, obtained by generating 30 SOMs and calculating the average. Table 1 shows a partial list of those lattices, including the lattice parameters as well as the TE for each data set. As a method to seek patterns from the statistical properties of lattices and the TE, we study the *mutual information function*, Υ, which quantifies the amount of information than may be obtained from a set of parameters about the TE. We decided to apply Υ instead of, for example, correlation indexes, as we believe there is a non linear relation between the parameters and the TE that correlation is unable to seek. The average correlation index from L, C, O, $C2$ and Lg to TE are -0.450, 0.747, 0.395, 0.682 and 0.291, in that order.

Table 1. Statistical parameters of lattices and the TE of the SOM generated on it for each data set. For a complete list of results and software, visit the page indicated at the beginning of this contribution.

O	C	L	$C2$	Lg	TE codon	TE espiral	TE ion.
0.009275	0.000000	9.003115	0.000000	0.111000	0.122857	0.111950	0.121000
0.009400	0.003333	11.104822	0.005634	0.101375	0.091250	0.119497	0.122312
0.009500	0.001083	12.790542	0.001383	0.096000	0.076667	0.122641	0.110923
0.009300	0.005000	10.000551	0.000958	0.101875	0.103750	0.123091	0.109821
0.009500	0.000000	9.806051	0.000000	0.112000	0.054286	0.123270	0.128784
0.009500	0.000000	13.302508	0.000000	0.009500	0.090000	0.140162	0.123346
0.009212	0.001000	9.003400	0.001100	0.111056	0.158792	0.148934	0.142277
0.018525	0.465144	9.319141	0.441860	0.275501	0.340000	0.347709	0.349721

In the following, we applied the ideas from Grosse et al [18]. In the context of statistical mechanics, Υ can be interpreted as follows. Consider a compound system (P, T) consisting of the two subsystems P and T. Let p_i be the propability of finding system P in state i, q_j the probability of finding system T in state j and let P_{ij} denote the joint probability of finding the compound system (P, T) in state (i, j). Then, the entropies of the systems P, T and (P, T) are defined by:

$$H[P]=-K_B \sum_i p_i ln p_i; \;\; H[T]=-K_B \sum_j q_i ln q_i; \;\; H[P,T] = -K_B \sum_{i,j} P_{ij} ln P_{ij}$$

Where $-K_B$ is the Boltzmann constant. If T and P are statistically independent, $H[P]+H[T] = H[P, T]$. If P and T are statistically dependent, the sum of the entropies is strictly greater than the entropy of the compound system. The mutual information $\Upsilon[P, T]$ is defined as:

$$\Upsilon[P, T] = H[P] + H[T] - H[P, T]$$

$\Upsilon[P, T] = 0$ if the subsystems P and T are statistically independent. Table 2 shows Υ between the statistical properties studied and the TE. As in principle the parameters are continouos, a discretization was applied. Several ranges were used. Table 2 shows results for a uniform discretization in 10 intervals for each parameter and for the TE.

Table 2. Mutual information (Υ) between statistical properties and the TE for the SOM for the studied data sets. It is shown as well the Υ between the parameters and a random number and the Υ between the parameters and its sum. Υ for the sum is always higher than for any set of parameters, but the Υ for the parameters and TE is always higher that that for random cases. $K_B = 1$.

stat. params.	TE Codon	TE Spiral	TE Ion.	random	sum
L, C, O	0.585941	0.488386	0.576361	0.281995	0.772346
L, C, Lg	0.400717	0.338130	0.365279	0.223872	0.504323
O, C, Lg	0.444214	0.410664	0.431783	0.122588	0.448434
L, C	0.548978	0.475263	0.556487	0.262669	1.020474
C, Lg	0.437715	0.413056	0.437303	0.127958	0.504225
O,C,L,C2	0.381733	0.979393	0.346154	-0.792224	0.579540

From table 2, we observe it is possible to obtain information about the TE shown by the SOM from the statistical properties of the lattice on which it is generated. In table 1, the first and seventh lattices show this fact. Both of them are very similar, except for the parameters C and $C2$ and the TE is higher in the seventh lattice, that with a higher C. The sixth and eighth lattices are regular. The former is a von Neumann neighborhood lattice and the later is defined over a Moore neighborhood. It is observed that their TE is higher than that of other lattices. Using $C4.5$ [20], we have found most SOMs generated on lattices that satisfies $C \leq 0.005$, $Lg \leq 0.119$, $0.009 \leq O \leq 0.0095$ and $7.7 \leq L \leq 11.105$ present the lowest TE (0.13 or less), whereas those lattices with $L > 0.3$ and $C > 0.06$ generate SOMs with the highest TE (> 0.3).

4 Conclusions

SOM has been generated mostly on regular lattices. However, TE on non-regular lattices are in many cases lower than that obtained in regular lattices. It is possible to obtain information about the SOM's TE on a given lattice from its statistical properties, as shown by the use of mutual information function. C4.5

algorithm extracts important rules in order to identify those parameters that are related to lattices in which SOMs present low TE. The studied statistical parameters set, altough incomplete, is a good characterization of lattices.

References

1. Ritter, H. Self-Organizing Maps on non-euclidean Spaces Kohonen Maps, 97-108, Eds.: E. Oja and S. Kaski, 1999
2. Cottrell, M. Fort, J.C., Pagés, G. Theoretical aspects of the SOM algorithm. Neurocomputing **21** (1998) 119-138
3. Kirk, J. Zurada, J. A two-stage algorithm for improved topography preservation in self-organizing maps. Int. Con. on Sys., Man and Cyb. **4** (2000) 2527-2532
4. Haykin, S. Neural Networks, a comprehensive foundation. 2nd. ed. Prentice Hall. 1999
5. Kohonen, T. Self-Organizing maps. 3rd. ed. Springer-Verlag. 2000
6. Flanagan, J. Sufficiente conditions for self-organization in the SOM with a decreasing neighborhood function of any width. Conf. of Art. Neural Networks. Conf. pub. No. 470 (1999)
7. Erwin, Obermayer, K. Schulten, K. Self-organizing maps: Ordering, convergence properties and energy functions. Biol. Cyb. **67** (1992) 47-55
8. Kohonen, T. Kaski, S. Lappalainen, H. Self-organized formation of various invariant-feature filters in the adaptive-subspace SOM. Neural Computation **9** (1997) 1321-1344
9. Bishop, C. Svenson, M. Williams, C. GTM: The Generative Topographic Mapping. Neural Computation **10** (1999) 215-234
10. Ontrup, J. Ritter, H. Hyperbolic Self-Organizing Maps for Semantic Navigation. Advances in Neural Information Processing Systems 14, 2001
11. Bauer, H. Hermann, M. Villman, T. Neural maps and topographic vector quantization. Neural Networks **12** (1999) 659-676
12. Kiviluoto, K. Topology preservation in Self-organizing Maps. Proc. of Int. Conf. on Neural Networks, ICNN'96 **1** (294-299)
13. Villmann, T., Der, R., Herrmann, M., Martinetz, T. Topology preservation in self-organizing maps: Exact definition and measurement. IEEE Trans. on Neural Networks **8** (1997) 256-266
14. Watts, D. Strogatz, S. Collective dynamics of 'small-world' networks. Nature **393** (1998) 440- 442.
15. Bollobás, B. and Riordan, O. Mathematical results on scale-free random graphs, on Handbook of graphs and networks. Bornholdt, E. Schuster, G. (2002) 1-34. Wiley.
16. Strogatz,. S. Exploring complex networks Nature **410** (2001) 268-276
17. Barabási, A., Albert, R. Emergence of scaling in random networks. Science **286** (1999) 509-512
18. Grosse, I. Herzel, H., Buldyrev, S. Stanley, E. Species dependence of mutual information in coding and noncoding DNA Physical Review E **61** (2000) 5624-5630
19. Shefi, O., Golding, I., Segev, R., Ben-Jacob, E., Ayali, A. Morphological characterization of in vitro neuronal networks. Phys. Rev. E **66** (2002) 21905-1, 21905-5
20. Quinlan, R. C4.5: programs for machine learning. Morgan Kaufmann. 1993.

Increasing Reliability of SOMs' Neighbourhood Structure with a Bootstrap Process

Patrick Rousset[1] and Bertrand Maillet[2]

[1] CEREQ, 10 place de la Joliette F-13567 Marseille
rousset@cereq.fr
[2] A.A.Advisors-QCG (ABN Amro Group), Variances and Paris-1 (TEAM/CNRS),
106 bv de l'hôpital F-75647 Paris cedex 13
bmaillet@univ-paris1.fr

Abstract. One of the most interesting features of self-organizing maps is the neighbourhood structure between classes highlighted by this technique. The aim of this paper is to present a stochastic method based on bootstrap process for increasing the reliability of the induced neighbourhood structure. The robustness under interest here concerns the sensitivities of the output to the sampling method and to some of the learning options (the initialisation and the order of data presentation). The presented method consists in selecting one map between a group of several solutions resulting from the same self-organizing map algorithm but with various inputs. The selected (robust) map, called R-map, can be perceived as the map, among the group, that corresponds to the most common interpretation of the data set structure.

1 Introduction

In the context of classification, self-organizing maps focus on the neighbourhood structure between classes. The aim of this paper is to present a stochastic method based on bootstrap process for increasing the reliability of the underlying neighbourhood structure. The robustness under interest here concerns the sensitivities of the output to the sampling method and to some of the learning options (the initialisation and the order of data presentation). Several articles have already been dedicated to the Kohonen algorithm theory, specifically focusing on aspects concerning its convergence (see [1] and [4]) and its sensitivity to parameters options (the initialisation, the order of data presentation, the rate of decrease of neighbourhood function, the adaptation parameter...). As a result, some tools for controlling the coherence of the neighbourhood structure have already been proposed (see for instance [6]). In the same vein, we propose here a two-step procedure aiming to increase the reliability of SOM neighbourhood structure. At the first step, a bootstrap process is used to build a table of probability for any pair of individuals to be neighbours. At the second step, we choose between several maps the most in conformity with this table, called R-map. In final, the R- map gives a summary of the data and a neighbourhood structure between classes that are less sensitive to the

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 433 – 438, 2005.

sampling (owing to the first step), the initialisation and the order of the data presentation (owing to the second step). The R-map can also be considered as the most common interpretation of the structure among several SOM solutions. We do not consider that the R-map is the "best" one concerning the interpretation. On the contrary, the variability of interpretations is probably rich in information especially when one can compare various interpretations with the “common” one. From an other side, increasing the stability of the neighbourhood structure can make SOM more attractive for some users that are perplex in front of the variability of the interpretations. To measure reliability, the use of a bootstrap scheme [3] is common and has already been proposed to assess the statistical significance of the neighbourhood structure of SOM [2]. When the algorithm is run several times, one expects, for any pair of individuals, the “probability to be neighbours” to be 0 or 1. The difference for any pair between the “empirical probability to be neighbours” and both values 0 and 1 gives a measure of the sensibility of SOM to the sampling and option parameters (such as initialisation, order of data presentation, decrease of neighbourhood size, adaptation parameter, etc.). To test the proposed method, we use another time bootstrap process on R-maps and SOM and compare the reliability of their respective neighbourhood structures. To present this technique and test it, we use it in a practical example. We have already use SOM to give a summary of the financial funds market [5].

2 A Bootstrap Scheme for Building the Table of Individual’s Probability to Be Neighbours One-to-One

When SOM are used in classification, the algorithm is applied to the complete database that is generally a sample of some unknown stationary distribution. A first concern refers to the question of the stability of the SOM solution (specifically the neighbourhood organisation) when the sample changes. A second concern regards the reliability to the sample, the data presentation order and the initialisation. For limiting the dependence of the outputs to the original data sample and to the arbitrary choice within an algorithm, it is common to use a bootstrap process with a re-sampling technique. Here, this idea is applied for affecting an empirical probability to any pair of individuals (to be neighbours in the SOM algorithm). For any pair, this probability is estimated by the number of times the individuals have been neighbours at ray 1 (highest level of proximity) when running several times the same SOM algorithm using re-sampled data series. In the following, we call $NEIGHT_{boot}$ the table containing empirical probabilities to have two individuals considered as neighbours at the end of the classification. As a remark, note that the algorithm uses only individuals in the given re-sampled set of individuals (representing around 60% of the original population). At the end, the left individuals are classified using computed distances to centroïds. Thus, at each step, table of empirical probabilities concern all individuals in the original dataset even if only a part of them have been used within the algorithm.

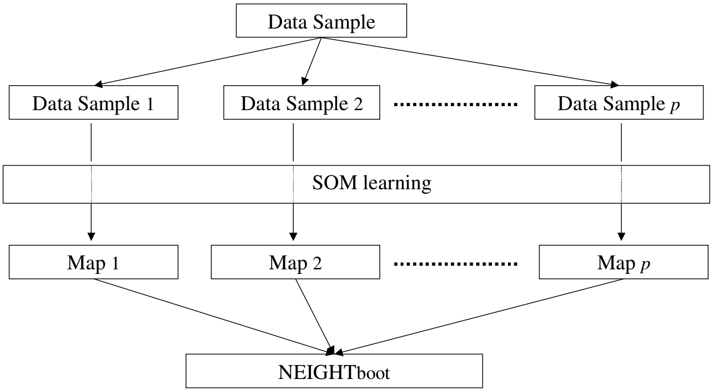


Fig. 1. Step1, bootstrap process in order to build the table *NEIGHT*$_{boot}$ of individual's empirical probability to be neighbours one to one

3 Choosing the R-Map from the Table of Individual's Probability to Be Neighbours One-to-One

When the table *NEIGHT*$_{boot}$ is built, the first step is over. In the second step, the SOM algorithm is also executed several times, but without re-sampling. For any map, we can build the *table NEIGHT*$_{map}$, similar to previous one, in which values are 1 for a pair of neighbour individuals and 0 for others. Then, using the Froebenius norm, we can compute the distance between both neighbourhood structures, defined respectively at the end of step1 (re-sampling the data) and step 2 (computing several maps with the original data), as follows:

$$D = \frac{1}{N^2}\sqrt{\sum_{(i,j)\in P}\left(NEIGHT_{boot}(i,j) - NEIGHT_{map}(i,j)\right)^2} \qquad (1)$$

where *P* is the set of *N*² individuals pairs (*i,j*). The selected R-map is the one between all SOM solutions that minimises the distance *D*. The R-map gives indeed a

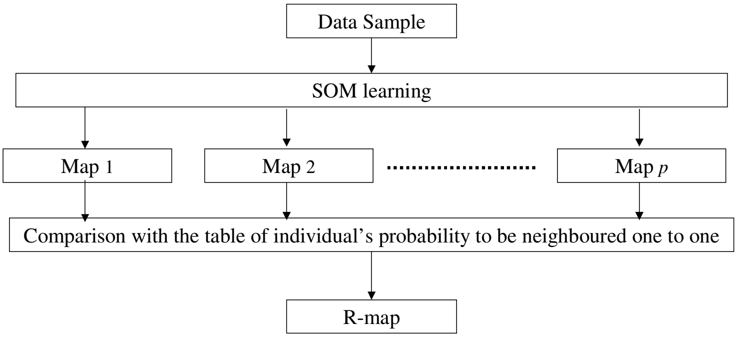

Data Sample
SOM learning
Map 1
Map 2
Map p
Comparison with the table of individual's probability to be neighboured one to one
R-map

Fig. 2. Step2, selection of the R-map between *p* solutions of the SOM algorithm

summary of the data and a neighbourhood structure between classes that are less sensitive to the sampling (after to the first step), the initialisation and the order of the data presentation (after the second step). The R-map can be then considered as the most common interpretation of the structure.

4 Test of the Method with a Financial Application to Hedge Fund Classification

We compare hereafter the classical 'simple' SOM method with the two-step algorithm presented in previous sections. More precisely, we first build several maps with the SOM algorithm as in [5] and, second, several maps leading to the definition of many R-maps. We then compute both related tables of individual's probability to be neighbours one-to-one: the $NEIGHT_{map}$ (set of 'simple' SOM with no two-step bootstrap process) and the $NEIGHT_{R\text{-}map}$ (corresponding to the two-step algorithm presented above). The theoretical probabilities 1, 0 and U_N/U correspond respectively to individuals that are sure to be respectively neighbours, un-neighbours and neighbours by chance. U_N/U is the uniform probability for a pair of individuals to belong to U_N neighbour classes out of U (for a grid structure, U_N equals 9 at ray 1).

As an illustration, we use a study already published, in which was proposed a robust typology of hedge funds based on the NAV time-evolutions, leading to a sound financial characterisation of the proposed typology with external risk measures (see [5] for more details). A typology of reference (given by *StandandPoors™*) based on the portfolio manager declared strategy is known to be unsatisfying for many reasons, from the lack of transparency of proprietary well-protected strategies, data error, changes in situation or complex mixture of 'pure' strategies. In this context, SOM is used in order to clean the classification from some arbitrary choices. The confidence in the neighbourhood is obviously crucial when real financial applications (fund of funds asset allocation and risk management) are at stake. In this practical example, the data set is composed with 294 funds and 67 observations of monthly NAV from January 1995 to September 2000. The chosen structure of the map is a six-by-six grid. Thirty R-maps are build each time in step one (using each time thirty samples of randomly chosen 194 individuals amongst the possible 294 hedge funds). From these 30 R-maps, we can build a table of individuals' probability to be neighbours one-to-one called as before $NEIGHT_{R\text{-}map}$. From 30 new self-organizing maps, we can build the equivalent $NEIGHT_{map}$. As an example, we present below Table 1, which is an abstract of three tables glued together: two tables called $NEIGHT_{map}$ (resulting from two repeated SOM on the same original data) and one table called $NEIGHT_{R\text{-}map}$ (resulting from the two-step procedure). Figures in Table 1 concerns empirical probabilities of pairs composed with individual number 1 and individuals number 25 to 33. We can see that, for any pair, the empirical probabilities in the column R-map are closer to 1 respectively to 0 when they are higher respectively lower than 25.00% (*i.e.* U_N/U =9/36 here). This property indicates that the neighbourhood structure with R-maps is more reliable than with classical maps. In table 2, columns indicate, for any empirical probability to be neighbours, the number of pairs concerned. When we take into consideration the R-map, 42.14%, 16.22 %, 12.41% and 8.58% of the set of pairs have an empirical probability to be neighbours, respectively, lower than 10.00%, greater than 80%, greater than 90%, equal to 100.00%. In comparison, in the case of classical maps, the respective values are 27.85%, 12.03%, 6.07%, 1.4% in the first case (Map 1) and 19.98%, 11.8%, 5.5% et 1.35% in the second case (Map 2). Thus, table 2 show as well the greater reliability of the neighbourhood structure in the case of R-maps when using the two-step algorithm procedure.

Table 1. Frequency to be neighbours for any pair of individuals

individual 1	individual 2	map 1	map 2	R-map
1	25	1	0,97	1
1	26	0,93	0,8	0,93
1	27	0,97	0,87	1
1	28	0,97	0,87	1
1	29	0,93	0,8	0,97
1	30	0,17	0,03	0
1	31	0,13	0,03	0
1	32	0,47	0,5	0,5
1	33	0,23	0,2	0

Table 2. Frequency of pairs of individuals for any empirical probability to be neighbours

value	frequency			cumulativ percent		
	map1	map2	R-map	map1	map2	R-map
0<<= 0.1	24076	17271	36420	27.85	19.99	42.14
0,1<<=0.2	10270	14568	8196	39.74	36.84	51.62
0,2<<=0.3	9664	10184	6698	50.92	48.63	59.37
0,3<<=0.6	23640	8648	4792	78.27	76.34	75.59
0,6<<=0.7	5175	6106	3374	84.25	83.40	79.49
0,7<<=0.8	3412	3964	3708	88.20	87.99	83.78
0,8<<=0.9	5448	5150	3288	94.50	93.95	87.59
0,9<<=1	4751	5229	10728	100.00	100.00	100.00

5 Remarks

As partially shown in the previous illustration, R-map method reduces SOM sensitivity to three parameters (the sample, the data presentation order and the initialisation. A similar technique can include others (the adaptation parameter) but not parameters linked to the neighbourhood structure (the size of the map). As a second remark, we can indicate that the distance D is not symmetrical between neighbour and un-neighbour, as a random distribution into the U units would create the probability $9/U$ to be neighbour. For example, in table 1, frequency 0,23 is interpreted as "un-neighbour" instead of "undecided" (pair (1, 33)) and frequency 0,47 is associated to "undecided" instead of "neighbour" (pair (1, 32)). Such is SOM itself, as individuals defined as neighbours on the map are close in the input space but close individuals can be put far on the map (for example when the map makes a fold). the distance D reduces more the possibility for individuals to be "neighbours by chance" than "un-neighbours by chance". One can use an other distance such as D_1. As third remark, using R-map bootstrap model avoid to keep in memory any SOM outputs. then, the memory capacity to compute R-map is focused on the size of the NEIGHTboot table. This table can be very large (N^2 pairs of N individuals) but can be reduced to a list of pairs such as the distance D would still be significant.

$$D_1 = \frac{1}{N}\sqrt{\sum_{(i,j)\in P} \frac{\left(NEIGHTB_{boot}(i,j) - NEIGHTB_{map}(i,j)\right)^2}{\left(NEIGHTB_{boot}(i,j) - 9/U\right)^2}} \qquad (2)$$

6 Conclusion

The presented method consists in selecting one map between a group of several solutions of the same self-organizing map algorithm. The selected map, called R-map, can be perceived as the map, among the group, that corresponds to the most common interpretation of the data set structure (interpretation means, here, the classification and the neighbourhood structure between classes). The neighbourhood structure is generally more robust with R-maps than one of a randomly selected map among the group. This reliability concerns both sensitivity to the sampling and to some algorithm parameters, in particular the initialisation and the data presentation order. This technique can also be completed by the tool presented in [6] that allows to control the coherence of the neighbourhood structure with the intra-classes distance.

Finally, above aiming to offer a robust classification, R-map selection provides self-organizing maps users with a practical method that gives the same results - without any further interpretation - when executed several times.

References

1. Cottrell, M., Fort, J.C., Pages, G. : Theoretical Aspects of the SOM Algorithm, Neurocomputing 21 (1998) 119-138
2. De Bodt E., Cottrell M., Letremy P., Verleysen M. : On the Use of Self-Organizing Maps to Accelerate Vector Quantization, Neurocomputing, Elsevier, Vol. 56 (January 2004), pp. 187-203
3. Efron, B.,Tibshirani, R. : An Introduction to the Bootstrap, Chapman and Hall (1993)
4. Kohonen, T.: Self-Organising Maps, Springer, Berlin (1995)
5. Maillet, B., Rousset, P. : Classifying Hedge Funds with Kohonen Maps: A first Attempt, in Connectionist Approaches in Economics and Management Sciences, Lesage C., Cottrell M. (Eds) (2003)
6. Rousset, P., Guinot, C.: Distance between Kohonen Classes: Visualization Tool to Use SOM in Data Set Analysis and representation, International Work-Conference on Artificial Neural Networks II (2001) 119-126

Artificial Neural Receptive Field for Stereovision

Bogusław Cyganek

AGH - University of Science and Technology,
Al. Mickiewicza 30, 30-059 Kraków, Poland
cyganek@uci.agh.edu.pl

Abstract. A capability of depth perception in biological visual systems evolved throughout thousands of years to help animals and us, humans, to survive in a real life. This ability has helped us to navigate and avoid threatening obstacles. However, we still know very little about the biological processes that lead to such a perfection which is by far not achievable for artificial vision systems. Thus, proper models of these mechanisms would help in their better comprehension, as well as they could guide construction of better computer stereovision systems. In this paper we try to propose a new topology of an artificial neural network for the stereovision system. We try to construct a very simple model of a binocular system that is biologically inspired in a behavioral aspect and which, at the same time, is computationally efficient. It is a hybrid that consists of the convolutional, binocular receptive, and the Hamming neural networks. The input signal is non-parametrically transformed for better statistical preconditioning. The paper ends with the experimental results.

1 Introduction

Abilities of motion and depth perception have played a very important role in evolution of biological organisms. They have helped in daily navigation, avoidance of dangerous objects, capturing food, and so on. Despite much effort, we are still at the preliminary stage of cognition of the biological mechanism of vision, however. At the other hand, stereovision by computer cannot be compared to the speed, precision, and inferring perfection of biological counterparts [5][3][6].

It has been found that at the input layers of mammals' visual cortex most of the signals from the two eyes are separated, however neurons of its external layers respond to light present to either eye. Thus, the latter neurons have binocular receptive fields. Further on, the cortical area V1 (the striate cortex) is the first place in the visual pathways where individual neurons receive binocular input [7][11][12].

There have been some attempts to model binocular stereopsis with neural networks. For example Iwahori et.al. propose a neural network to the photometric stereo with a nearby rotational moving light source [9]. Another neural approach to the stereovision proposed [1]. Similarly Wei et.al. proposed a neural system with radial base functions to solve stereo problem [13]. Area based matching with the Hamming neural network is presented in [4]. This paper greatly extends this concept.

In this paper we present a simple model of an artificial binocular receptive field for computer stereopsis. We do not attempt to precisely mimic the biological system,

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 439 – 445, 2005.

however, but rather to model biologically inspired functional behavior on a computer system. The novel neural network structure is proposed that, under some simplifying assumptions, captures spatial relations among corresponding neuro-pixels. Their novelty comes from the hybrid structure of the proposed system which consists of the convolutional, binocular receptive network field, and the Hamming neural networks. The input signal is in a form of non-parametrically transformed intensity signals rather than pure intensity values. This proved to have much better statistical properties [3]. The Hamming neural network is applied with a winner-take-all layer that resolves winning disparity for a local receptive field.

The paper starts with a short introduction on binocular vision, then the input and processing layers of the proposed network are explained. Finally, we present experimental results and conclusions.

2 Binocular Depth Perception

Depth perception is possible with two or more visual sensors (i.e. eyes or cameras) observing the same scene but from a slightly different position. The human eyes constitute a very complex visual system. Fig. 1 depicts a basic model of the artificial binocular vision system obtained from two cameras with *parallel* optical axes.

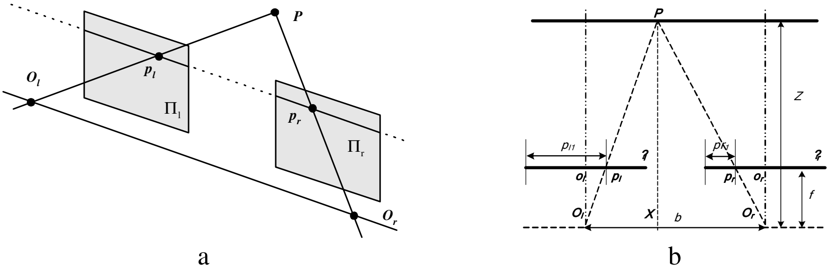


Fig. 1. Epipolar geometry of a canonical stereo setup. All images of 3D points (such as *P*) of opaque objects can lie only on corresponding epipolar lines. Π_l and Π_r are left and right camera planes, O_l, O_r are left and right centers of projection, respectively (a). Depth of a point *P* can be computed exclusively from difference of its images p_L and p_R created on the camera planes (b).

It can be easily shown [5][3] that depth of a 3D point *P* (i.e. its third coordinate) is directly dependant on the disparity value $D_x(p_l, p_r)$ between image points p_l and p_r:

$$D_x(p_l, p_r) = p_{r1} - p_{l1} = bf / Z \tag{1}$$

where the points $p_l=(p_{l1},p_{l2})$ and $p_r=(p_{r1},p_{r2})$ are image points (i.e. in the camera coordinate system) of a point *P*, *b* is a distance between cameras (a base line), *f* stands for cameras focus (the same value for the two cameras), *Z* is the sought third coordinate of a point *P* (Fig. 1).

The canonical stereo setup with pinhole cameras is assumed hereafter. However, in a general case, knowledge of the fundamental matrix *F* allows us to find epipolar lines in the forms: $l_r=Fp_r$, $l_l=F^Tp_l$, for the left and right cameras, respectively. The matrix *F* can also be computed with neural networks [2].

3 Neural Model of a Binocular System

In the human visual system (HVS) the signals from the two retinas are transferred to the primary visual cortex (V1, the layers 4C) through the lateral geniculate nucleus (LGN). It is characteristic that a way of signals from the right visual field leads to the left LGN and then to the left hemisphere of a brain. It is also interesting that within V1, signals from each eye are segregated into different layers of V1, whereas they are moved together into individual neurons of the superficial layers of the cortex.

Fig. 2 depicts our very simple computational model of a receptive field for stereovision which is biologically inspired in its behavioral aspect. In this paper we focus on its three core modules (grayed in Fig. 2). The optional convolutional NN allows for scale-space processing [6], while the radial-based NN (RBF) can be used for disparity resolving, once the disparity space is built [13].

It is already known [8] that some parts of V1 respond to the binocular stimuli – sometimes this part is called a binocular receptive field (BRF), although sometimes the concept of BRF is extended onto the whole visual pathway from eyes to the brain.

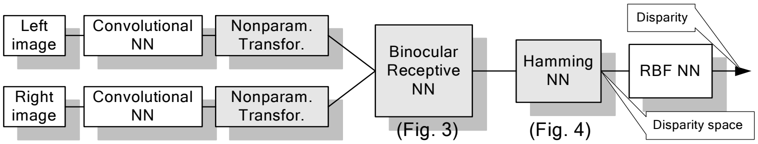


Fig. 2. Neural model of a receptive field for stereovision

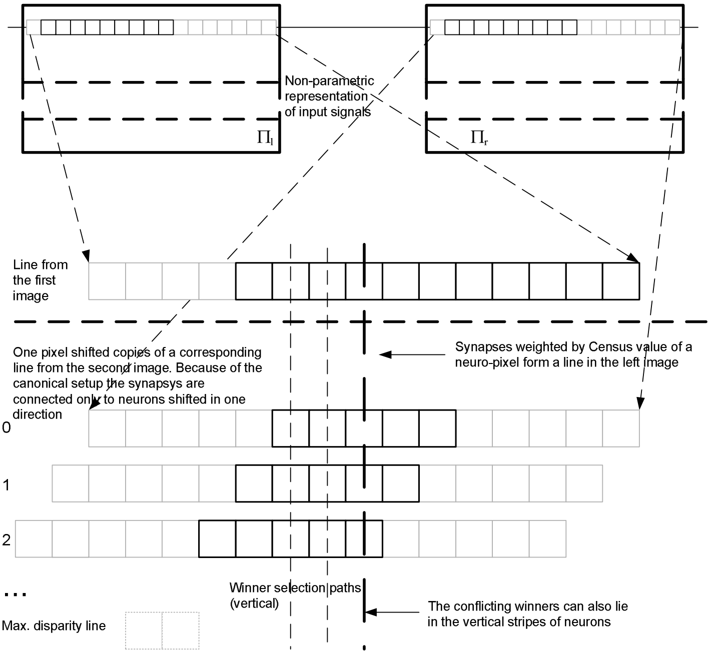

Non-parametric representation of input signals
Π_l
Π_r
Line from the first image
One pixel shifted copies of a corresponding line from the second image. Because of the canonical setup the synapsys are connected only to neurons shifted in one direction
Synapses weighted by Census value of a neuro-pixel form a line in the left image
0
1
2
...
Max. disparity line
Winner selection paths (vertical)
The conflicting winners can also lie in the vertical stripes of neurons

Fig. 3. The binocular receptive neural network

In these terms, the presented model of a receptive field for stereovision can be conceived as BRF in its latter sense. At the other hand, the binocular receptive NN (middle grayed block in Fig. 2, and Fig. 3) relates to the BRF in its former definition.

Elements of the binocular receptive NN are neuro-pixels from corresponding lines. However, their input constitute signal of the *Census* transformed intensities which have very desirable properties. The most important is that *Census* representation conveys information on *nearest neighborhood* around a pixel [14][3]. This crucial property allows us to process only one-pixel-wide stripes (Fig. 3). There is one reference line from the first image and number of one-pixel-shifted lines from the second one. Due to assumed canonical stereo setup we need only to shift lines in one direction. Their number is the same as expected maximum disparity between images (1). A disparity is determined by a winning neuron along the vertical stripe – this is achieved by the Hamming NN. (Fig. 2 and Fig. 4). The number of a line with a winning neuron gives the sought disparity value. We can also analyze mutual disparity relations to refine the match, e.g. to impose the ordering constraint [3].

Fig. 4 shows structure of the Hamming NN for disparity selection in a receptive field, i.e. on the winner selection paths in Fig. 3. The purpose of the first layer of neurons presented in Fig. 4, is to compute the Hamming distance between an input vector and corresponding weights. Output of this layer is given as follows [10]:

$$\mathbf{H} = 0.5\left(n^{-1}\mathbf{W}^{(1)}\mathbf{x} + \mathbf{1} \right) \tag{2}$$

where **1** is a column vector and $W^{(1)}$ is a matrix of weights which are *Census* values of the pretending neuro-pixels from the shifted lines; **x** is a neuro-pixel from the first image. More details can be found in [4] where the area based version was discussed.

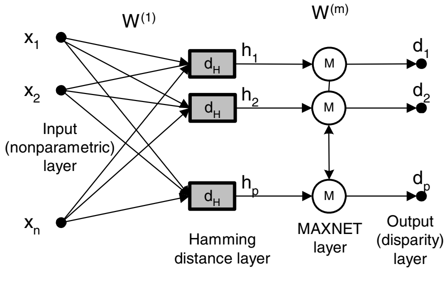


Fig. 4. Structure of the Hamming neural network for disparity selection in the receptive field

The purpose of the final MAXNET layer in Fig. 4 is to choose the winning neuron, i.e. the best matching one in a sense of the Hamming distance. The distinctive feature of this recursive layer is an incentive connection from the neuron to itself. Values of neurons of this layer are determined by the following recursive equation [10][4]:

$$d_i(k) = \mathrm{f}\left(y_i(k-1) - \varepsilon(k-1)\sum_{j\neq i} y_j(k-1) \right) \quad , \quad \varepsilon(k) = (p-k)^{-1} \tag{3}$$

where k denotes iteration step and the initial value for $y_j(0) = h_j$, p is a number of initial patterns (equal to the expected maximum disparity), y is an output layer.

The iteration proceeds up to the point where values of neurons are fixed and only one neuron has value other than zero. It is possible, however, that under sufficiently balanced conditions no winner can be reliably determined. To cope with such cases we can increase the window of the *Census* transformation, which in our experiments was from 3×3 up to 7×7. However, the better solution is to include an additional block of disparity selection that operates in the entire disparity space. In Fig. 2 it is the RBF-NN module, e.g. the one presented in [13]. However, in this realization and experiments the simple averaging in the disparity space was applied.

4 Experimental Results

The computational platform consists of the IBM PC with Pentium 4, 3.4 GHz, implementation in C++. Results of the dashed blocks in Fig. 2 are presented here.

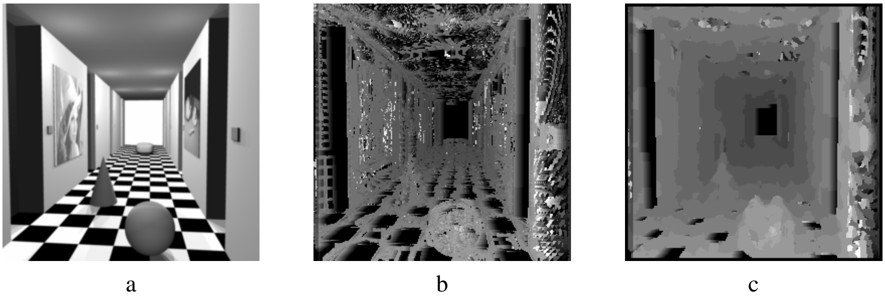


Fig. 5. The neural technique applied to the "*Corridor*" stereo image (256×256): left image of a pair (a), disparity map with 5×5 averaging in the disparity space (b), disparity map with 7×7 averaging in the disparity space (c). The *Census* transformation was done in a 7×7 window.

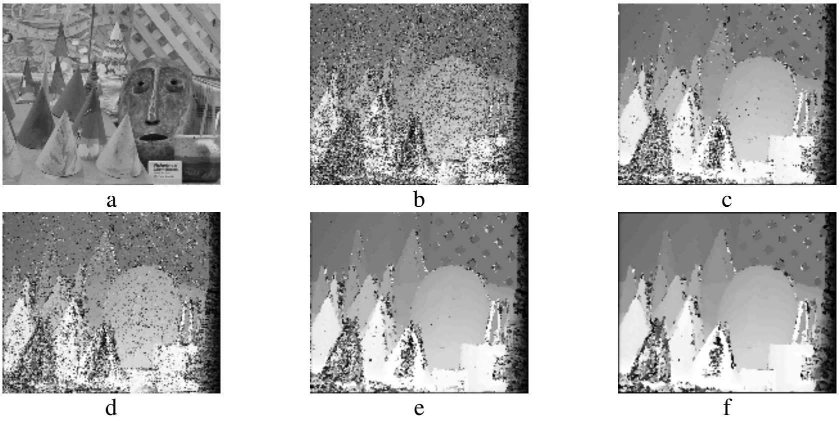


Fig. 6. The "*Mask*" 450×375 (a). Disparity map in 5×5 *Census* representation, with averaging in the disparity space: 1×1 (b), 5×5 (c); For *Census* 7×7, averaging: 1×1 (d), 5×5 (e), 7×7 (f).

Fig. 5 presents results of the neural matching of the "*Corridor*" stereo pair of size 256×256 (a). The disparity map obtained with 1×1 averaging in the disparity space is depicted in (b) while disparity with 7×7 in (c). The *Census* window is 7×7 in all cases. Execution times are 99.35 and 107.55 s, respectively. Number of iterations k=80.

Fig. 6 contains results of the "Mask" 450×375 stereo pair. Disparity map in 5×5 *Census* representation, with 1×1 averaging in the disparity space (b), averaging 5×5 (c); For *Census* 7×7 representation: averaging 1×1 (d), 5×5 (e), 7×7 (f). Number of iterations k=200. Execution times for this case are as follows: 92.12 for (b), 94.0 for (c), 98.22 for (d), 120.83 for (e), and 124.22 for (f).

5 Conclusions

This paper presents a hybrid neural model of a receptive field for stereovision; Its behavior is motivated biologically by the BRF. In this paper we especially focus on a computational aspect of this model. The novel neural network structure is proposed that accepts *Census* transformed nonparametric signals on its input. Its main virtue is encoded information on neighboring pixels and resistance to image noise and lighting inequalities. Then, thanks to the layers of neurons that correspond to the shifted image lines, the disparity space is found with the Hamming NN ended with the MAXNET winner selection layer. Based on this structure it is also possible to impose additional conditions, such as ordering or disparity gradient constraint. Additional neural processing modules allow for biologically motivated scale-space computations. A global search in the disparity space is also possible for better resolving of matches.

The experimental results show that the system exhibits many interesting properties and can be joined with other neural modules, e.g. to the neural module for computation of the fundamental matrix or match disambiguation network.

References

1. Cruz J.M., Pajarez G., Aranda J.: A Neural Model in Stereovision Matching, Neural Networks, Vol.8, No.5 (1995) 805-813
2. Cyganek B.: Neural Computation of the Fundamental Matrix, Springer LNAI 3070, 7th Int. Conference on Artificial Intelligence and Soft Computing – ICAISC 2004 (2004) 718-723
3. Cyganek, B.: Three Dimensional Image Processing, (in Polish) EXIT Warsaw (2002)
4. Cyganek, B.: Neural Networks Application to the Correlation-Based Stereo-Images Matching, Proceedings of the EANN '99, Poland (1999) 17-22
5. Faugeras, O.D., Luong, Q.-T.: The Geometry of Multiple Images. MIT Press (2001)
6. Haykin, S.: Neural Networks. A Comprehensive Foundation. Prentice Hall (1999)
7. Howard, I.P., Rogers B.J.R.: Binocular Vision and Stereopsis. Oxford Univ. Press (1995)
8. Hubel, D.H.: Eye, Brain and Vision. W.H. Freeman & Company (1995)
9. Iwahori Y., Woodham R.J.: Neural Network Based Photometric Stereo with a Nearby Rotational Moving Light Source. IEICE Trans. Inf. & Syst., Vol. E80-D, No. 9 (1997)
10. Osowski, S.: Neural Networks – Algorithmic Approach, (in Polish), WNT (1996)
11. Poggio, G.F, Talbot, W.H.: Mechanisms of static and dynamic stereopsis in foveal cortex of the rhesus monkey. J. Physiol., 315 (1981) 469-492
12. Wandell B.A.: Foundations of Vision. Sinauer Associates Publishers Inc. (1995)

13. Wei G-Q., Brauer W., Hirzinger G.: Intensity- and Gradient-Based Stereo Matching Using Hierarchical Gaussian Basis Functions. IEEE PAMI, Vol. 20, No. 11 (1998)
14. Zabih, R., Woodfill, J.: Non-Parametric Local Transforms for Computing Visual Correspondence. Proc. Third European Conf. Computer Vision (1994) 150-158

Pattern Detection Using Fast Normalized Neural Networks

Hazem M. El-Bakry

University of Aizu, Aizu Wakamatsu, Japan 965-8580
d8071106@u-aizu.ac.jp

Abstract. Neural networks have shown good results for detecting of a certain pattern in a given image. In our previous papers [1-6], a fast algorithm for object/face detection was presented. Such algorithm was designed based on cross correlation in the frequency domain between the input image and the weights of neural networks. Our previous work also solved the problem of local subimage normalization in the frequency domain. In this paper, the effect of image normalization on the speed up ratio of pattern detection is presented. Simulation results show that local subimage normalization through weight normalization is faster than subimage normalization in the spatial domain. Moreover, the overall speed up ratio of the detection process is increased as the normalization of weights is done off line.

1 Introduction

Pattern detection is a fundamental step before pattern recognition. Its reliability and performance have a major influence in a whole pattern recognition system. Nowadays, neural networks have shown very good results for detecting a certain pattern in a given image [5,8,10,12]. But the problem with neural networks is that the computational complexity is very high because the networks have to process many small local windows in the images [7,9]. In our pervious papers, we presented fast neural networks based on applying cross correlation in the frequency domain between the input image and the input weights of neural networks. It was proved that the speed of these networks is much faster than conventional neural networks [1-3]. It was also proved that fast neural networks introduced by previous authors [11,13,14] are not correct. The reasons for this were given in [1-3]. The problem of subimage (local) normalization in the Fourier space was presented in [10]. This problem was solved in [6]. We proved that the number of computation steps required for weight normalization is less than that needed for image normalization. But, we did not discuss the effect of normalization on the speed up ratio. Here, the effect of weight normalization on the speed up ratio is theoretically and practically discussed. Mathematical calculations prove that the new idea of weight normalization, instead of image normalization, provides good results and increases the speed up ratio. This is because weight normalization requires fewer computation steps than subimage normalization. Moreover, for neural networks, normalization of weights can be easily done off line before starting the search process. In section 2, fast neural networks for pattern detection are described. Subimage normalization in the frequency domain

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 447 – 454, 2005.

through normalization of weights is presented in section 3. The effect of weight normalization on the speed up ratio is presented in section 4.

2 Fast Neural Networks Based on Cross Correlation in the Frequency Domain for Pattern Detection

Finding a certain pattern in the input image is a search problem. Each subimage in the input image is tested for the presence or absence of the required pattern. At each pixel position in the input image each subimage is multiplied by a window of weights, which has the same size as the subimage. The outputs of neurons in the hidden layer are multiplied by the weights of the output layer. A high output implies that the tested subimage contains the required pattern and vice versa. Thus, we may conclude that this searching problem is cross correlation between the image under test and the weights of the hidden neurons. The convolution theorem in mathematical analysis says that a convolution of f with h is identical to the result of the following steps: let F and H be the results of the Fourier Transformation of f and h in the frequency domain. Multiply F and H in the frequency domain point by point and then transform this product into the spatial domain via the inverse Fourier Transform. As a result, these cross correlations can be represented by a product in the frequency domain. Thus, by using cross correlation in the frequency domain a speed up in an order of magnitude can be achieved during the detection process [1,2,3,4,5,6,8]. In the detection phase, a sub image I of size mxn (sliding window) is extracted from the tested image, which has a size PxT, and fed to the neural network. Let X_i be the vector of weights between the input sub image and the hidden layer. This vector has a size of mxn and can be represented as mxn matrix. The output of hidden neurons h_i can be calculated as follows:

$$h_i = g\left(\sum_{j=1}^{m} \sum_{k=1}^{n} X_i(j,k) I(j,k) + b_i \right) \tag{1}$$

where g is the activation function and b_i is the bias of each hidden neuron (i). Eq. (1) represents the output of each hidden neuron for a particular subimage I. It can be obtained from image Z as follows:

$$h_i(u,v) = g\left(\sum_{j=-m/2}^{m/2} \sum_{k=-n/2}^{n/2} X_i(j,k)\, Z(u+j, v+k) + b_i \right) \tag{2}$$

Eq. (2) represents a cross correlation operation. Given any two functions f and d, their cross correlation can be obtained by [1,3]:

$$f(x,y) \otimes d(x,y) = \left(\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(x+m, y+n) d(m,n) \right) \tag{3}$$

Therefore, Eq. (2) may be written as follows [1,3]:

$$h_i = g\left(Z \otimes X_i + b_i\right) \tag{4}$$

where h_i is the output of the hidden neuron (i) and h_i (u,v) is the activity of the hidden unit (i) when the sliding window is located at position (u,v) and (u,v) $\in$[P-m+1, T-n+1]. Now, the above cross correlation can be expressed in terms of the Fourier Transform:

$$Z \otimes X_i = F^{-1}\left(F(Z) \bullet F^*\left(X_i\right)\right) \tag{5}$$

Hence, by evaluating this cross correlation, a speed up ratio can be obtained comparable to conventional neural networks. Also, the final output of the neural network can be evaluated as follows:

$$O(u,v) = g\left(\sum_{i=1}^{q} w_o(i)\, h_i(u,v) + b_o\right) \tag{6}$$

where q is the number of neurons in the hidden layer. O(u,v) is the output of the neural network when the sliding window is located at the position (u,v) in the input image Z.

The complexity of cross correlation in the frequency domain can be analyzed as follows [1-4]:

1- For a tested image of NxN pixels, the 2D-FFT requires a number equal to $O(N^2 \log_2 N^2)$ of complex computation steps. Also, the same number of complex computation steps is required for computing the 2D FFT of the weight matrix for each neuron in the hidden layer.

2- At each neuron in the hidden layer, the inverse 2D FFT is computed. So, q backward and (1+q) forward transforms have to be computed. Therefore, for an image under test, the total number of the 2DFFT to compute is $O((2q+1)N^2 \log_2 N^2)$.

3- The input image and the weights should be multiplied in the frequency domain. Therefore, a number of complex computation steps equal to $O(qN^2)$ should be added.

4- The number of computation steps required by fast neural networks is complex and must be converted into a real version. It is known that the two dimensions Fast Fourier Transform requires $O((N^2/2)\log_2 N^2)$ complex multiplications and $O(N^2 \log_2 N^2)$ complex additions. Every complex multiplication is realized by six real floating point operations and every complex addition is implemented by two real floating point operations. So, the total number of computation steps required to obtain the 2D-FFT of an NxN image is [1-4]:

$$\rho = O(6((N^2/2)\log_2 N^2) + 2(N^2 \log_2 N^2)) \tag{7}$$

which may be simplified to:

$$\rho = O(5N^2 \log_2 N^2) \tag{8}$$

Performing complex dot product in the frequency domain also requires $O(6qN^2)$ real operations.

5- In order to perform cross correlation in the frequency domain, the weight matrix must have the same size as the input image. So, a number of zeros = (N^2-n^2) must be added to the weight matrix. This requires a total real number of computation steps = $O(q(N^2-n^2))$ for all neurons. Moreover, after computing the FFT2 for the weight matrix, the conjugate of this matrix must be obtained. So, a real number of computation steps $=O(qN^2)$ should be added in order to obtain the conjugate of the weight matrix for all neurons. Also, a number of real computation steps equal to O(N) is required to create butterflies complex numbers $(e^{-jk(2\Pi n/N)})$, where 0<K<L. These (N/2) complex numbers are multiplied by the elements of the input image or by previous complex numbers during the computation of FFT2. To create a complex

number requires two real floating point operations. So, the total number of computation steps required for fast neural networks becomes [1-4]:

$$\sigma=O((2q+1)(5N^2\log_2N^2)+6qN^2+q(N^2-n^2)+qN^2+N) \quad (9)$$

which can be reformulated as:

$$\sigma=O((2q+1)(5N^2\log_2N^2)+q(8N^2-n^2)+N) \quad (10)$$

6- Using a sliding window of size nxn for the same image of NxN pixels, $O(q(2n^2-1)(N-n+1)^2)$ computation steps are required when using traditional neural networks for pattern detection process. The theoretical speed up factor η can be evaluated as follows [1-4]:

$$\eta=O\left(\frac{q(2n^2-1)(N-n+1)^2}{(2q+1)(5N^2\log_2N^2)+q(8N^2-n^2)+N}\right) \quad (11)$$

The correct theoretical speed up ratio with different sizes of the input image and different in size weight matrices is listed in Table 1. Practical speed up ratio for manipulating images of different sizes and different in size weight matrices is listed in Table 2 using 700 MHz processor and Matlab ver 5.3.

Table 1. The theoretical speed up ratio for images with different sizes

Image Size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	3.67	5.04	6.34
200x200	4.01	5.92	8.05
300x300	4.00	6.03	8.37
400x400	3.95	6.01	8.42
500x500	3.89	5.95	8.39
600x600	3.83	5.88	8.33
700x700	3.78	5.82	8.26
800x800	3.73	5.76	8.19
900x900	3.69	5.70	8.12
1000x1000	3.65	5.65	8.05

Table 2. Practical speed up ratio for images with different sizes using MATLAB ver 5.3

Image Size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	7.88	10.75	14.69
200x200	6.21	9.19	13.17
300x300	5.54	8.43	12.21
400x400	4.78	7.45	11.41
500x500	4.68	7.13	10.79
600x600	4.46	6.97	10.28
700x700	4.34	6.83	9.81
800x800	4.27	6.68	9.60
900x900	4.31	6.79	9.72
1000x1000	4.19	6.59	9.46

3 Subimage Centering and Normalization in the Frequency Domain

In [10], the authors stated that image normalization to avoid weak or strong illumination could not be done in the frequency space. This is because the image normalization is local and not easily computed in the Fourier space of the whole image. Here, a simple method for image normalization is presented. Centering and normalizing the image can be obtained by centering and normalizing the weights as follows [8]:

Let $\overline{X}_{rc}$ be the zero-mean centered subimage located at (r,c) in the input image ψ:

$$\overline{X}_{rc} = X_{rc} - \overline{x}_{rc} \tag{12}$$

where, $\overline{x}_{rc}$ is the mean value of the sub image located at (r,c). We are interested in computing the the dot product between the subimage $\overline{X}_{rc}$ and the weights W_i that is:

$$\overline{X}_{rc}.W_i = X_{rc}.W_i - \overline{x}_{rc}.W_i \tag{13}$$

where,

$$\overline{x}_{rc} = \frac{X_{rc}}{n^2} \tag{14}$$

Combining (13) and (14), we get the following expression:

$$\overline{X}_{rc}.W_i = X_{rc}.W_i - \frac{X_{rc}}{n^2}.W_i \tag{15}$$

which is the same as:

$$\overline{X}_{rc}.W_i = X_{rc}.W_i - X_{rc}.\frac{W_i}{n^2} \tag{16}$$

The centered zero mean weights are given by:

$$\overline{W}i = W_i - \frac{W_i}{n^2} \tag{17}$$

Also, Eq. (16) can be written as:

$$\overline{X}rc\,.W_i = X_{rc}.\left(W_i - \frac{W_i}{n^2}\right) \tag{18}$$

So, we may conclude that:

$$\overline{X}_{rc}.W_i = X_{rc}.\overline{W}i \tag{19}$$

which means that dot multiplying a centered image with the weight matrix is equal to the dot multiplication of the non – normalized image with the normalized weight matrix.

4 Effect of Weight Normalization on the Speed up Ratio

Normalization of subimages in the spatial domain (in case of using traditional neural networks) requires $2n^2(N\text{-}n+1)^2$ computation steps. On the other hand, normalization of subimages in the frequency domain through normalizing the weights of the neural networks requires $2qn^2$ operations. This proves that local image normalization in the frequency domain is faster than that in the spatial one. By using weight normalization, the speed up ratio for image normalization Γ can be calculated as:

$$\Gamma = O\left(\frac{(N-n+1)^2}{q}\right) \tag{20}$$

The speed up ratio of the normalization process for images of different sizes is listed in Table 3. As a result, we may conclude that:

1- Using this technique, normalization in the frequency domain can be done through normalizing the weights in spatial domain.
2- Normalization of an image through normalization of weights is faster than normalization of each subimage.
3- Normalization of weights can be done off line. So, the speed up ratio in the case of weight normalization can be calculated as follows:

Table 3. The speed up ratio of the normalization process for images of different sizes

Image Size	Speed up ratio(n=20,q=100
100x100	62
200x200	328
300x300	790
400x400	1452
500x500	2314
600x600	3376
700x700	4638
800x800	6100
900x900	7762
1000x1000	9624

a) For Conventional Neural Networks:

The speed up ratio equals the number of computation steps required by conventional neural networks with image normalization divided by the number of computation steps needed by conventional neural networks with weight normalization, which is done off line. The speed up ratio η_c in this case can be given by:

$$\eta_c = O\left(\frac{q(2n^2-1)(N-n+1)^2 + 2n^2(N-n+1)^2}{q(2n^2-1)(N-n+1)^2}\right) \tag{21}$$

which can be simplified to:

$$\eta_c = O\left(1 + \frac{2n^2}{q(2n^2-1)}\right) \tag{22}$$

b) For Fast Neural Networks:
The over all speed up ratio equals the number of computation steps required by conventional neural networks with image normalization divided by the number of computation steps needed by fast neural networks with weight normalization, which is done off line. The over all speed up ratio η_o can be given by:

$$\eta_o = O\left(\frac{q(2n^2-1)(N-n+1)^2 + 2n^2(N-n+1)^2}{(2q+1)(5N^2\log_2 N^2) + q(8N^2 - n^2) + N}\right) \quad (23)$$

which can be simplified to:

$$\eta_o = O\left(\frac{(N-n+1)^2(q(2n^2-1)+2n^2)}{(2q+1)(5N^2\log_2 N^2) + q(8N^2 - n^2) + N}\right) \quad (24)$$

The relation between the speed up ratio before (η) and after (η_o) the normalization process can be summed up as:

$$\eta_o = O\left(\eta + \frac{2n^2(N-n+1)^2}{(2q+1)(5N^2\log_2 N^2) + q(8N^2 - n^2) + N}\right) \quad (25)$$

The overall speed up ratio with images of different sizes and different sizes of windows is listed in Table 4. We can easily note that the speed up ratio in case of image normalization through weight normalization is larger than the speed up ratio (without normalization) listed in Table 2. This means that the search process with normalized fast neural networks is done faster than conventional neural networks with or without normalization of the input image.

Table 4. Simulation results for speed up ratio in case of normalizing the input weights

Image Size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	82.32	99.86	119.76
200x200	97.20	116.76	136.65
300x300	110.61	132.98	147.54
400x400	120.77	143.65	163.59
500x500	128.48	156.86	178.79
600x600	133.51	168.64	195.32
700x700	138.99	174.43	213.12
800x800	134.18	170.65	206.18
900x900	138.87	175.11	219.96
1000x1000	136.93	173.54	214.65

5 Conclusion

Normalized neural networks for fast pattern detection in a given image have been presented. It has been proved mathematically and practically that the speed of the detection process becomes faster than conventional neural networks. As a result, fast neural networks, based on cross correlation in the frequency domain, have become faster than conventional neural networks. Furthermore, we have generally proved that the speed up ratio in the case of image normalization through normalization of weights is faster than without normalization.

References

1. Hazem M. El-Bakry, and Qiangfu Zhao " Fast Object/Face Detection Using Neural Networks and Fast Fourier Transform," accepted for publication in the International Journal of Signal Processing.
2. Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Pattern Detection Using Normalized Neural Networks and Cross Correlation in the Frequency Domain," accepted and under publication in the EURASIP Journal on Applied Signal Processing.
3. Hazem M. El-Bakry, and Qiangfu Zhao, " Fast Sub-Matrix Detection Using Neural Networks and Cross Correlation in the Frequency Domain, " Second Workshop of Tohoku Branch, IPSJ, (Information Processing Society of Japan), University of Aizu, Aizuwakamatsu, Japan, Jan. 21, (2005)
4. Hazem M. El-Bakry, "Comments on Using MLP and FFT for Fast Object/Face Detection," Proc. of IEEE IJCNN'03, Portland, Oregon, July, 20-24, (2003) 1284-1288
5. Hazem M. El-Bakry, "Human Iris Detection Using Fast Cooperative Modular Neural Networks and Image Decomposition," Machine Graphics & Vision Journal (MG&V), Vol. 11, No. 4, (2002) 498-512)
6. Hazem M. El-Bakry, "Face detection using fast neural networks and image decomposition," Neurocomputing Journal, vol. 48, (2002) 1039-1046.
7. S. Srisuk and W. Kurutach, "A New Robust Face Detection in Color Images", Proc. of IEEE Computer Society International Conference on Automatic Face and Gesture Recognition (AFGR'02), Washington D.C., USA, May 20-21, (2002) 306-311
8. Hazem M. El-Bakry, "Automatic Human Face Recognition Using Modular Neural Networks," Machine Graphics & Vision Journal (MG&V), Vol. 10, No. 1, (2001) 47-73
9. Ying Zhu, Stuart Schwartz, and Michael Orchard, "Fast Face Detection Using Subspace Discriminate Wavelet Features," Proc. of IEEE Computer Society International Conference on Computer Vision and Pattern Recognition (CVPR'00), South Carolina, June 13 - 15, , Vol.1, (2000) 1636-164.
10. R. Feraud, O. Bernier, J. E. Viallet, and M. Collobert, "A Fast and Accurate Face Detector for Indexation of Face Images," Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition, Grenoble, France, 28-30 March, (2000)
11. S. Ben-Yacoub, B. Fasel, and J. Luettin, "Fast Face Detection using MLP and FFT," in Proc. of the Second International Conference on Audio and Video-based Biometric Person Authentication (AVBPA'99)", (1999)
12. S. Baluja, H. A. Rowley, and T. Kanade, "Neural Network - Based Face Detection," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 20, No. 1, pp. 23-38, 1998.
13. Beat Fasel, "Fast Multi-Scale Face Detection," IDIAP-Com 98-04, (1998)
14. S. Ben-Yacoub, "Fast Object Detection using MLP and FFT," IDIAP-RR 11, IDIAP, (1997)

Neural Network Model for Extracting Optic Flow

Kunihiko Fukushima and Kazuya Tohyama

Tokyo University of Technology, Hachioji, Tokyo 192-0982, Japan
fukushima@media.teu.ac.jp, tmkz@mf.teu.ac.jp
http://www.teu.ac.jp/media/~fukushima/

Abstract. In the MST area of the monkey, there are cells that respond selectively to specific motions of a large area of the visual field, such as rotation or expansion/contraction. They respond steadily even when the location of the center of optic flow shifts on the retina. They are thought to analyze optic flows of the retinal images. This paper proposes a neural network model for these cells. The model performs processing similar to mathematical operations called *rot* and *div* in the vector field analysis. It is a hierarchical multilayered network: retina, layer V1, layers MT and layer MST. Each MT cell extracts relative velocity between two adjoining small visual fields, and an MST cell adds the response of many MT cells to extract a specific optic flow. The difference in type of optic flows extracted by MST cells can be created simply by the difference in relative locations between inhibitory and excitatory areas in the receptive fields of the preceding MT cells.

1 Introduction

In the visual systems of mammals, information concerning visual motion is mainly analyzed through the occipito-parietal pathway: retina → area V1 (primary visual cortex) → area MT (middle temporal area) → area MST (medial superior temporal area).

In a dorsal part of MST of the monkey, there are cells that respond selectively to specific motions of a large area of the visual field, such as rotation, expansion/contraction and translation [1][2][3][4][5]. These cells are reported to respond steadily even when the location of the center of optic flow shifts on the retina. They are thought to analyze optic flows of the retinal images.

Historically, there have been two major streams of hypotheses proposed so far to explain neural networks extracting optic flow: direction mosaic hypothesis and vector field hypothesis [6].

The direction mosaic hypothesis supposes that a rotation- or expansion/contraction-selective receptive field is constructed receiving signals from a group of direction selective cells, whose preferred directions are arranged circularly or radially. This mechanism alone cannot explain, however, the location-invariant responses of MST cells. To acquire the location-invariance, the receptive field of an MST cell has to be made up with a number of direction-mosaics from different locations of the visual field [1]. This requires a very complicated operations performed by a single cell, which does not necessarily seem biologically plausible.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 455–460, 2005.

The vector field hypothesis assumes that the brain analyzes optic flows using operations called *rot* and *div* in the vector field analysis. It has long been pointed out mathematically that optic flows can be extracted with these operations [7]. Neural network models based on this hypothesis, however, have little been proposed so far.

This paper proposes a neural network model for MST cells, which follows the vector field hypothesis, and demonstrates the behavior of the model with computer simulation.

2 Principles of Extracting Optic Flow

Since the rotation and expansion/contraction of the visual field are extracted with a similar mechanism, we will first discuss the principle of extracting rotation.

We assume the moving velocity $\boldsymbol{v} = (v_x, v_y)$ of the scene can be measured at any location (x, y) in the visual field. Here, we define a value ρ with

$$\rho = \frac{\partial v_y}{\partial x} + \left(-\frac{\partial v_x}{\partial y}\right) \tag{1}$$

Incidentally, this value equals the signed absolute value (or length) of a vector obtained by $\boldsymbol{v} = \nabla \times \boldsymbol{v}$, which is an operation called *rotation* (or *curl*) in the vector-field analysis.

If an image covering the whole visual field is rotating around a point with a constant angular velocity, ρ becomes constant everywhere in the visual field and is not affected by the shift in location of the center of the rotation. The sign of ρ indicates the direction of the rotation (counter-clockwise or clockwise), and its absolute value is proportional to the angular velocity of the rotation.

We can then extract the rotation of the whole visual field, if we sum (or take an average of) the values of ρ for all points within a large area S.

$$\bar{\rho} = \sum_S \rho \tag{2}$$

Expansion/contraction of optic flow can also be extracted with a similar mechanism. We define δ, which corresponds to a scalar, div $\boldsymbol{v} = (\nabla \cdot \boldsymbol{v})$, obtained by an operation called *divergence* in the vector-field analysis.

$$\delta = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} \tag{3}$$

If the image is expanding radially from (or contracting to) a point, δ becomes always constant everywhere in the visual field and is not affected by the location of the center of the expansion/contraction. A positive value of δ indicates expansion, and a negative value indicates contraction. We then can extract the expansion/contraction of the optic flow, $\bar{\delta}$, by summing (or taking an average of) δ within a large area in the visual field.

3 The Model

Outline of the Model. The model is a hierarchical multilayered network. It consists of layers of cells connected in a cascade: retina → V1 → MT$_{\text{abs}}$ → MT$_{\text{rel}}$ → MST.

Layer V1 corresponds to the primary visual cortex, layers $\mathrm{MT}_{\mathrm{abs}}$ and $\mathrm{MT}_{\mathrm{rel}}$ to area MT, and layer MST to the dorsal part of area MST of the animal.

The retina of our model is a photoreceptor array that receives visual image. In the computer simulation discussed later, visual stimuli are moving random-dot patterns.

Each layer consists of a number of cell-planes. Incidentally, a cell-plane is a group of cells that are arranged retinotopically and share the same set of input connections. As a result, all cells in a cell-plane have receptive fields of an identical characteristic, but the locations of the receptive fields differ from cell to cell.

V1-Cells. Each cell of layer V1 has a small receptive field and has a direction selectivity. It responds strongly to an object moving in a specific direction. The strength of the response is assumed to be proportional to the velocity of the object.

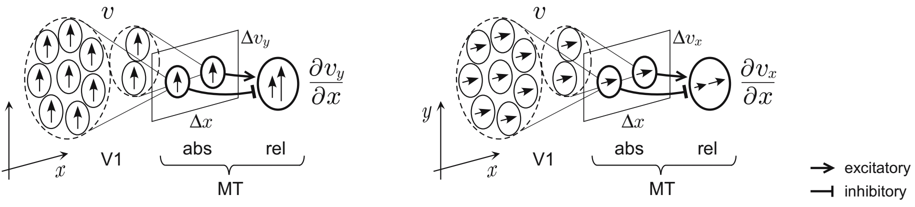
Since our model uses cells whose responses are proportional to the firing frequencies of biological neurons, their responses take only positive analog values or zero, and cannot be negative. We need a pair of cells to represent the velocities of opposite directions: that is, one for responding to the positive, and the other to the negative velocity.

$\mathbf{MT}_{\mathbf{abs}}$-Cells. Layer $\mathrm{MT}_{\mathrm{abs}}$ consists of cells that extract absolute velocity. In other words, the cells respond to the moving velocities of the stimuli in a similar way as V1-cells, but have somewhat larger receptive fields. Each $\mathrm{MT}_{\mathrm{abs}}$-cell has excitatory input connections from a group of V1-cells of the same preferred direction but of slightly scattered receptive-field locations. It detects and responds in proportion to the maximum output among these V1-cells. Namely, each $\mathrm{MT}_{\mathrm{abs}}$-cell extracts the maximum velocity within its receptive field.

$\mathbf{MT}_{\mathbf{rel}}$-Cells. Layer $\mathrm{MT}_{\mathrm{rel}}$ consists of cells that extract relative velocity. Namely, $\mathrm{MT}_{\mathrm{rel}}$-cells work to extract gradients of velocity, which correspond to $\partial v_y/\partial x$, $(-\partial v_x/\partial y)$, $\partial v_x/\partial x$ and $\partial v_y/\partial y$ in Eqs. (1) and (3).

Each $\mathrm{MT}_{\mathrm{rel}}$-cell receives antagonistic inputs from two $\mathrm{MT}_{\mathrm{abs}}$-cells of the same preferred direction, whose receptive fields adjoins each other. An $\mathrm{MT}_{\mathrm{rel}}$-cell thus extracts the gradient of local velocity at a certain location in the visual field.

Fig. 1(a) illustrate the network that extracts $\partial v_y/\partial x$, which is used for extracting rotation. It calculates the difference in vertical velocity Δv_y between two points separated horizontally by Δx, by receiving antagonistic inputs from two $\mathrm{MT}_{\mathrm{abs}}$-cells of vertical preferred direction, whose receptive field centers differ horizontally.



(a) Network for extracting rotation (b) Network for extracting expansion

Fig. 1. A neural network for extracting relative velocity

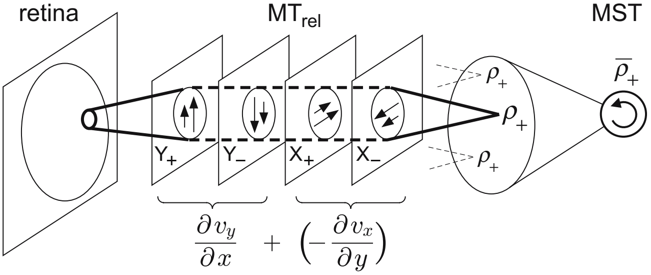

retina
MT_rel
MST
$\bar{\rho}_+$
ρ_+
ρ_+
ρ_+
Y+
Y−
X+
X−
$\frac{\partial v_y}{\partial x} + \left(-\frac{\partial v_x}{\partial y}\right)$

Fig. 2. Extracting rotation at a location from four components of relative velocities.

Each term of Eq. (3), which is used for extracting expansion/contraction, can be obtained by a network like Fig. 1(b), which resembles Fig. 1(a). The only difference between the two networks resides in the relative locations of the receptive fields of the excitatory and inhibitory MT$_{\rm abs}$-cells. For the extraction of rotation, they are arranged in the direction orthogonal to their preferred direction, while they are arranged in the direction of their preferred direction for the extraction of expansion/contraction.

Since MT$_{\rm abs}$-cells, like V1-cells, cannot generate negative outputs, a pair of cells is used to represent positive and negative components of the velocity in each direction. Gradients of velocity have to be extracted from these four components. Fig. 2, for example, illustrates MT$_{\rm rel}$-cells relevant to extracting counter-clockwise rotation ρ_+ at a location in the visual field. The outputs of cells Y_+ and Y_- in the figure correspond to values of $\partial v_y/\partial x$ extracted from the positive and negative components of the vertical velocity, respectively, and outputs of X_+ and X_- to the value of $(-\partial v_x/\partial y)$ extracted from those of the horizontal velocity.

MST-Cells. Each cells of layer MST gathers outputs of MT$_{\rm rel}$-cells from a very large area and detect the types of optic flow, such as rotation or expansion/contraction. The optic flow of a large visual field, $\bar{\rho}$ or $\bar{\delta}$, can thus be extracted independently of the location of its center. We write $\bar{\rho}_+$ and $\bar{\rho}_-$ to represent MST-cells extracting counter-clockwise ($\bar{\rho} > 0$) and clockwise ($\bar{\rho} < 0$) rotations, respectively. We also write $\bar{\delta}_+$ and $\bar{\delta}_-$ to represent MST-cells extracting expansion ($\bar{\delta} > 0$) and contraction ($\bar{\delta} < 0$).

Mathematically, the operation performed by a counterclockwise-rotation selective MST-cell, for example, can be expressed roughly by

$$\bar{\rho}_+ = \sum_S \left(\varphi\left[\frac{\partial v_y}{\partial x}\right] + \varphi\left[-\frac{\partial v_x}{\partial y}\right] \right) \tag{4}$$

because the outputs of MT$_{rel}$-cells do not take negative values. Here, $\varphi[\,]$ represents a function that takes the positive component of its argument, namely, $\varphi[x] = \max(x, 0)$.

It should be noted here that an MST-cell adds responses of MT$_{\rm rel}$-cells directly without introducing interneurons. Although we showed an illustration like Fig. 2 above, this does not mean that an independent cell like ρ_+ really exists in the network.

Sharper Direction Tuning Curves. In the explanations up to here, we have made discussions assuming that the direction selectivity (or the tuning curve) of V1-cells takes a cosine curve. If it follows a cosine curve, the velocity in any direction can be represented by two pairs of cells representing the velocity in two orthogonal directions, namely, by four cells of preferred directions of 0°, 90°, 180° and 270°. The direction selectivity of V1- and MT-cells of animals, however, is usually slightly sharper than a cosine curve. In the computer simulation, we prepare cells of eight different preferred directions. In other words, our model uses, not only horizontal and vertical components of velocities, but also velocity-components in another set of orthogonal axes rotated by 45°.

4 Computer Simulation

The neural network is simulated on a computer. The stimuli given to the retina are black-and-white random-dot patterns that are moving.

The mechanism of extracting velocity by V1-cells is abbreviated in the present simulation. We assume that the output of each V1-cell is proportional to the velocity of the dot that drops in its receptive field, and that velocity-tuning of V1-cells follows a cosine curve.

Fig. 3 shows a response of the network to a random-dot pattern rotating counter-clockwise. Each square in the figure represent a cell-plane. The output of each cell is shown by the darkness of a dot in the figure.

Each cell-plane of layer MST consists of nine cells, which have receptive fields of the same characteristics, but at different locations. It can be seen from the figure that all $\bar{\rho}_+$-cells, which extract counter-clockwise rotation, exhibit strong responses independently of the locations of their receptive fields, while all other MST-cells are silent.

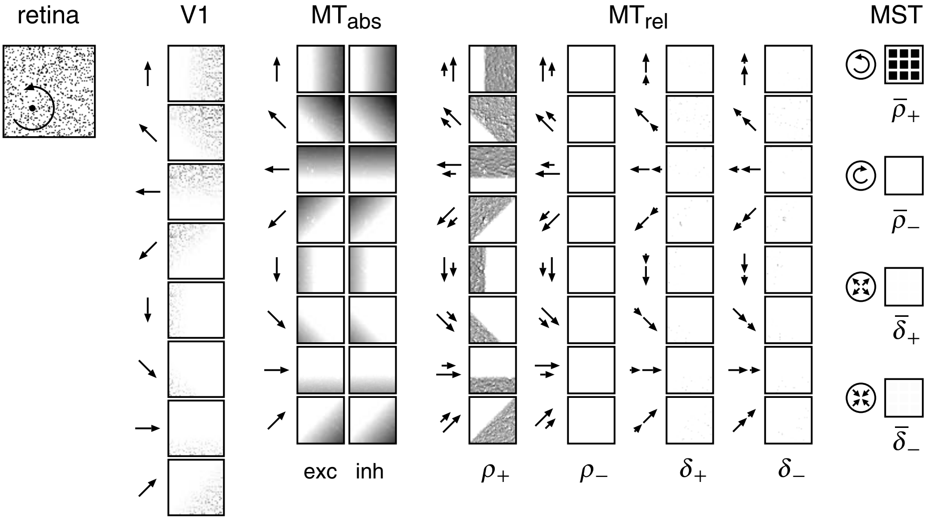


Fig. 3. A response of the model to a random-dot pattern rotating counter-clockwise

This means that MST-cells extract the counter-clockwise rotation correctly without being affected by the location of the center of the rotation.

5 Discussion

In this paper, we have proposed a neural network model that extracts optic flows and demonstrated with computer simulation that the MST-cells in our model respond to optic flow in a similar way as MST-cells in the biological brain.

We have shown that rotation and expansion/contraction can be extracted in our model, not by neural networks of different architectures, but by neural networks of the same architecture by simply changing the relative locations between inhibitory and excitatory areas in the receptive fields of $\mathrm{MT}_{\mathrm{rel}}$-cells. More specifically, we hypothesize that the difference in the relative location of the inhibitory to the excitatory areas create four different groups of $\mathrm{MT}_{\mathrm{rel}}$-cells, which take charge of extracting ρ_+, ρ_-, δ_+ and δ_-; and hence four different types of MST-cells, $\bar{\rho}_+$, $\bar{\rho}_-$, $\bar{\delta}_+$ and $\bar{\delta}_-$.

Existence of such $\mathrm{MT}_{\mathrm{rel}}$-cells in the biological brain are suggested by several reports on single cell recordings from the visual area MT of macaque monkeys. For example, Tanaka, et al. reported MT-cells that are thought to respond to relative motion [8]. Xiao, et al. reported that, in half of the MT-cells, antagonistic surround was asymmetric, and the inhibitory area was located in a single region on one side of the excitatory area of the receptive field [9].

Graziano, et al. recorded MST neurons that respond preferentially to spiral motions [5]. Our model can also emulate well these MST-cells, if we properly choose $\mathrm{MT}_{\mathrm{rel}}$-cells, from which the MST-cell receive signals.

Acknowledgement. This work was partially supported by Grants-in-Aid for Scientific Research #14380169 from Japan Society of Promotion of Science.

References

1. H. Saito, M. Yukie, K. Tanaka, K. Hikosaka, Y. Fukada, E. Iwai: *J. Neuroscience*, **6**, 145–157 (1986).
2. K. Tanaka, H. Saito: *J. Neurophysiology*, **62**, 626–641 (1989).
3. K. Tanaka, Y. Fukada, H. Saito: *J. Neurophysiology*, **62**, 642–656 (1989).
4. C. J. Duffy, R. H. Wurtz: *J. Neurophysiology*, **65**, 1329–1345 (1991).
5. M. S. A. Graziano, R. A. Andersen, R. J. Snowden: *J. Neuroscience*, **14**, 54–67 (1994).
6. C. J. Duffy, R. H. Wurtz: *J. Neurophysiology*, **65**, 1346–1359 (1991).
7. J. J. Koenderink: *Vision Research*, **26**, 161–179 (1986).
8. K. Tanaka, K. Hikosaka, H. Saito, M. Yukie, Y. Fukada, E. Iwai: *J. Neuroscience*, **6**, 134–144 (1986).
9. D.-K. Xiao, S. Raiguel, V. Marcar, G. A. Orban: *Cerebral Cortex*, **7**, 1047–3211 (1997).

A Modular Single-Hidden-Layer Perceptron for Letter Recognition

Gao Daqi, Shangming Zhu, and Wenbing Gu

Department of Computer Science,
State Key Laboratory of Bioreactor Engineering,
East China University of Science and Technology,
Shanghai 200237, China
{gaodaqi, smzhu, guwb}@ecust.edu.cn

Abstract. An n-class problem is decomposed into n two-class problems. Naturally, modular multilayer perceptrons (MLPs) come into being. A single-output MLP is behalf of a class and trained by a two-class learning subset. A training subset only consists of all samples from a special class and a part samples from the nearest classes. If the decision boundary of a single-output MLP is open, its outputs are amended by a correction coefficient. This paper clarifies such a fact that the generalization of a single-output MLP is seriously affected by the sample disequilibrium situation. Therefore, the samples from the little class have to be multiplied an enlarging factor. The result of letter recognition shows that the above methods are effective.

1 Introduction

Classification accuracy and learning speeds are two main items of evaluations for classifiers. Undoubtedly, multilayer perceptrons (MLPs) with sigmoid activation functions have got a great success for solving such small-sample, low-dimensional and limited-class problems as the exclusive-OR (XOR) problem [1-2], the IRIS recognition [2-3], etc. And sometimes so are they for the small-sample and high-dimensional problems, for instance, the Sonar recognition [2-3]. It is a pity that up to now there are few reports in the literature for MLPs to successfully solve the large-sample, high-dimensional and multi-class problems. Hand-written digit recognition [4], a relatively typical large-scale learning dataset, is not a multi-class problem, because it has only 10 classes. Generally speaking, the structures and learning algorithms of classifiers suiting for the small-scale problems cannot be simply generalized for the large-scale problems.

This paper focuses on modular single-hidden-layer perceptrons with sigmoid activation functions (SAFs) for solving the large-sample, high-dimensional and multi-class learning problems.

2 Decomposition of Large-Scale Learning Sets

After decomposing an *n*-class problem into *n* two-class problems, an obvious fact is that a great number of samples have not actually any use in deciding the decision boundary of a specific subclass. Based on that, we can first decompose a complicated

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 461 – 467, 2005.

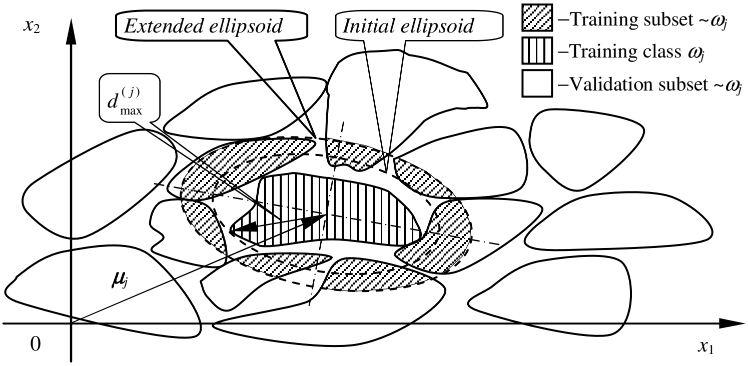

x_2
Extended ellipsoid
Initial ellipsoid
▨ –Training subset ~ω_j
▥ –Training class ω_j
□ –Validation subset ~ω_j
$d_{\max}^{(j)}$
$\boldsymbol{\mu}_j$
0
x_1

Fig. 1. Component parts of the training subset $\boldsymbol{\Xi}^{(j)}$

n-class problem into n simpler two-class problems, and then get rid of those futile patterns for determining the decision boundary of a certain class, say ω_j, as shown in Fig. 1. Obviously, only the samples in the shaded regions take part in. The detailed approach to determining a most economic training subset is as follows.

(A) Draw an initial hyper-dimensional oblique ellipsoid, which center and half axis directs coincide with those of ω_j, and which sizes of major and minor axes are determined by the max Mahalanobis distance $d_{\max}^{(j)}$ between samples from ω_j and the center $\boldsymbol{\mu}_j$. Let $\boldsymbol{X}\in R^{N\times m}$ is the original learning set, $\boldsymbol{X}^{(j)}=(\boldsymbol{x}_1^{(j)},\ldots,\boldsymbol{x}_p^{(j)},\ldots,\boldsymbol{x}_{Nj}^{(j)})\in R^{Nj\times m}$ the sample set from ω_j, and $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ the mean vector and covariance matrix of ω_j respectively, the max Mahalanobis distance of the initial ellipsoid is

$$d_{\max}^{(j)}=\max_{1\le p\le N_j}\left(\left(\boldsymbol{x}_p^{(j)}-\boldsymbol{\mu}_j\right)\boldsymbol{\Sigma}_j^{-1}\left(\boldsymbol{x}_p^{(j)}-\boldsymbol{\mu}_j\right)^T\right)^{\frac{1}{2}} \tag{1}$$

If $\boldsymbol{\Sigma}_j$ is singular or almost singular, $\boldsymbol{X}^{(j)}$ is added a normal perturbation matrix $\boldsymbol{\varepsilon}\sim N(\boldsymbol{0}, 0.01\boldsymbol{1})$, namely $\boldsymbol{X}_{new}^{(j)}=\boldsymbol{X}^{(j)}+\boldsymbol{\varepsilon}$. Here $\boldsymbol{1}\in R^{Nj\times m}$ is a matrix consisting solely of 1's.

(B) Calculate the number $N_{\sim j0}$ of samples that are included within the initial ellipsoid and from the other classes. Supposed $\boldsymbol{x}_p\in\boldsymbol{X}$ and $\boldsymbol{x}_p\notin\boldsymbol{X}^{(j)}$, $N_{\sim j0}$ is calculated by the following iterative loop:

```
$N_{\sim j0}=0$.
**For** $p=1:N-N_j$
    *if* $\left(\boldsymbol{x}_p-\boldsymbol{\mu}_j\right)\boldsymbol{\Sigma}_j^{-1}\left(\boldsymbol{x}_p-\boldsymbol{\mu}_j\right)^T\le\left(d_{\max}^{(j)}\right)^2$
        $N_{\sim j0}\leftarrow N_{\sim j0}+1$
**End**
```

(C) Determine the max radius of the extended ellipsoid. Our starting point is that the more the samples from the other classes are included within the initial ellipsoid, the larger the max radius of the extended one is. The max radius of the extended ellipsoid can be determined by

$$D_{\max}^{(j)}=\left(1+N_j/N_{\sim j0}\right)d_{\max}^{(j)} \tag{2}$$

If $N_{\sim j0}=0$, $D_{\max}^{(j)}=100d_{\max}^{(j)}$.

*(D)*Form the finally learning subset $\Xi^{(j)}$, which consists of all the samples within the extended ellipsoid. $\Xi^{(j)}$ can be determined by the following loop:

```
$\Xi^{(j)}(0)= \boldsymbol{X}^{(j)}$.
$k=0$.
For $p=1:N-N_j$
  if $\left(\boldsymbol{x}_p-\boldsymbol{\mu}_j\right)\boldsymbol{\Sigma}_j^{-1}\left(\boldsymbol{x}_p-\boldsymbol{\mu}_j\right)^T \le \left(D_{\max}^{(j)}\right)^2$
    $\Xi^{(j)}(k+1) \leftarrow \Xi^{(j)}(k)+\boldsymbol{x}_p$
    $k \leftarrow k+1$
End
$N_{\sim j}=k$.
$N^{(j)}=N_j+N_{\sim j}$.
```

In brief, the final learning subset $\Xi^{(j)}$ is only made up of the samples both $\boldsymbol{X}^{(j)}$ from ω_j and $\boldsymbol{X}^{(\sim j)}$ from the other categories that are most neighboring to ω_j.

3 Structure and Generalization of Modular MLPs

3.1 Modular Single-Hidden-Layer Perceptrons

In correspondence with the decomposition of a complex n-class problem into n simple 2-class problems, the paper proposes a type of modular MLP ensemble. For an n-class problem, there exist n MLP modules, and each module is with a single-output structure. After the structures of all modules are determined by learning, the module with the maximum output $max(y_{pj})$ gives the label of sample $\boldsymbol{x}_p$.

3.2 Amendment of Outputs of Many-to-One MLPs

Fig. 2 describes the generalization regions of three MLPs with the same structure, and all have the classification accuracy of 100% for the training samples from class ω_j, but quite different recognition correct rate for the test set. The recognizing accuracy for the test set depends upon whether or not the generalization regions coincide with the distribution regions at the best degree.

Generally speaking, if the margins between different classes are small, the magnitudes of weights and thresholds in an MLP ought to be relatively large in order to widen the margins. It is not suitable to take the real output of an MLP as the grade of membership of sample $\boldsymbol{x}$. In other words, the real output y_j of MLP j should be added a correction coefficient. The real output of a one-output MLP is related to both the input-to-hidden and the hidden-to-output weights. Here, we add an amended term to the real output, namely

$$\rho_j(\boldsymbol{x})=\exp\left(-\left(y_j-\overline{y}_j\right)^2\right)\exp\left(-\frac{1}{2}\sum_{i=1}^{m}\left(\frac{x_i-\mu_{ji}}{\sigma_{ji}}\right)^2\right) \qquad (3)$$

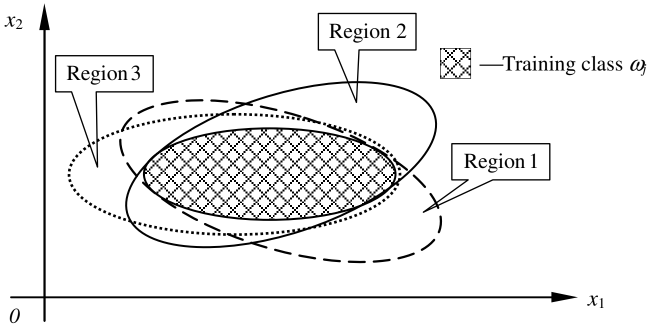


Fig. 2. Different generalization regions of a structure-fixed MLP

where $\overline{y}_j$ is the output of MLP j corresponding to the mean vector $\boldsymbol{\mu}_j$ of class ω_j, and σ_{ji} is the ith mean variance component of ω_j. When $\boldsymbol{x}$ is far from $\boldsymbol{\mu}_j$, the second term on the right side of (3) will quickly lessen. For example, if $|\boldsymbol{x}-\boldsymbol{\mu}_j|\geq 3|\boldsymbol{\sigma}_j|$, the second term is equal to $e^{-4.5}\leq 0.011$. The grade of membership of $\boldsymbol{x}$ will be $\rho_j(\boldsymbol{x})\leq 0.011$ on condition that $|\boldsymbol{x}-\boldsymbol{\mu}_j|\geq 3|\boldsymbol{\sigma}_j|$, because the first term on the right side of (3) is not larger than 1.0.

3.3 Sample Disequilibrium Problems in Training Subsets

Without a doubt, if the number of samples in a two-class problem, the decision boundary of the corresponding MLP will be close to the central sections of margins. In fact, the number of samples in training subsets is often unequal. Under the situation, the final decision boundaries will be closer to the classes with fewer samples.

Let us illustrate the problem by a simple example. Suppose two classes ω_1 and ω_2 are in the 1-dimensional space, ω_1 only consists of one sample situated at point 0, and ω_2 10 samples all located at point 1.0. We take a single one-to-one neuron with SAF to divide ω_1 and ω_2. When the decision equation is x-0.5=0, which is just the midpoint, the sum-of-squared error is $10\times(1-1/(1+exp(-0.5)))^2+(0-1/(1+exp(0.5)))^2=1.5679$. If the decision function is x=0, which coincides with point 0, the sum-of-squared error is $10\times(1-1/(1+exp(-1)))^2+(0-1/(1+exp(0)))^2=0.9733$. How incredible it is! And it is just the learning result of an MLP using the least square method! As a result, the generalization performances of MLPs trained with the disequilibrated sample constructions will be quite poor.

According to the above analysis, we can infer that the final decision boundary of ω_j using MLP module j trained with the subset $\boldsymbol{\Xi}^{(j)}=\boldsymbol{X}^{(j)}+\boldsymbol{X}^{(\sim j)}$ will be closer to $\boldsymbol{X}^{(j)}$ because the number N_j in $\boldsymbol{X}^{(j)}$ is often several times smaller than $N_{\sim j}$ in $\boldsymbol{X}^{(\sim j)}$. This situation is quite disadvantageous for the test points that are a little farther from the decision boundary of ω_j but belong to ω_j. In order to solve the disequilibrated problems in the training subsets, the following method can be used.

Add some virtual samples to the smaller part $\boldsymbol{X}^{(j)}$. If $\boldsymbol{x}_p^{(j)}\in\boldsymbol{X}^{(j)}$, the corresponding pth weight increment $\Delta w\left(\boldsymbol{x}_p^{(j)}\right)$ is added an enlarging factor $N_{\sim j}/N_j$, otherwise the increment keeps unchanged. All one need to do is only to add judgment conditions in the programming. Let the pth first-order partial derivatives of the sum-of-squared E_j with respect to the weight components v_{jh} and w_{hi} for $\boldsymbol{x}_p^{(j)}$ are respectively

$$\Delta v_{jh}(p) = \partial E_{pj} / \partial v_{jh} = -\left(t_{pj} - y_{pj}\right) y_{pj} \left(1 - y_{pj}\right) z_{ph}^{(j)} \tag{4}$$

$$\Delta w_{hi}^{(j)}(p) = \partial E_{pj} / \partial w_{hi}^{(j)} = -\left(t_{pj} - y_{pj}\right) y_{pj} \left(1 - y_{pj}\right) v_{jh} z_{ph}^{(j)} \left(1 - z_{ph}^{(j)}\right) x_{pi}^{(j)} \tag{5}$$

All one need to do is just to add some judgments in the following loop:

```
$\partial E_j / \partial v_{jh} = 0$, $\partial E_j / \partial w_{hi}^{(j)} = 0$.
**For** $p$=1: $N^{(j)}$,
If $\boldsymbol{x}_p^{(j)} \in \boldsymbol{X}^{(j)}$
    $\partial E_j / \partial v_{jh} \leftarrow \partial E_j / \partial v_{jh} + \left(N_{\sim j} / N_j\right) \Delta v_{jh}(p)$    (6)
    $\partial E_j / \partial w_{hi}^{(j)} \leftarrow \partial E_j / \partial w_{hi}^{(j)} + \left(N_{\sim j} / N_j\right) w_{hi}^{(j)}(p)$    (7)
**End**
```

In that way, there is hardly any additionally computational burden!

4 An Application Example-*Letter Recognition*

Our objective is to identify 26 capital letters (from A to Z) [2]. There are 20,000 samples in total. Each input component is scaled to fit into a range of integer values from 0 through 15. We typically take the first 16,000 samples for the training set and the remaining 4,000 ones for the test set. Any 16-s-26 MLP falls to solve the problem when taking all the 26 letters and 16,000 samples as a whole, no matter how careful to select the number s of hidden nodes and the learning parameters.

Here we take class "*C*" as an example to go in details on the generation of the training subset $\Xi^{(C)}$ as well as the training process of MLP module "*C*".

There are N_c=594 samples in class "*C*". According to (1), the max Mahalanobis distance of the initial ellipsoid is 8.2915, and there are 4,764 samples from other classes included in the ellipsoid. Therefore, the max radius of the extended ellipsoid is 9.3266,

Table 1. Recognition rates (%) of different classifiers for the test set

Classifier	G-ARTM AP	Genetic	IB1	BICA	CNNDA	C4.5
Recognition rate	95.95	82.7	95.7	90.3	81.7	77.7
Ref	[6]	[7]	[8]	[9]	[10]	[9]
Classifier	*K-NN*	NB	GBML	BML	M-MLP	M-MLP
Recognition rate	89.9	74.9	82.7	87.5	79.3	95.75
Ref	[9]	[9]	[9]	[9]	[11]	Our method

BICA: Bayesian Independent Component Analysis. C4.5: A Decision Tree. GBML: Gaussian Bayesian Maximum Likelihood. NB: Naive Bayesian. CNNDA: Cascade neural network design algorithm.

the number of samples from the other classes is $N_{\sim C}$=6,685. We find that the 6,685 samples are from all the other 25 classes. As a result, the final training subset consists of 594+6,685=7,279 samples, namely $\Xi^{(C)}\in R^{7,279\times 16}$, $X^{(C)}\in R^{594\times 16}$, and $X^{(\sim C)}\in R^{6,685\times 16}$. The enlarging factor of weight increment is $N_{\sim C}/N_C$= 6,685/594=11.2542.

According to the equation $s_0=int\lfloor 2\log_2 m\rfloor$ [5], the initial number of hidden nodes is s_0=11, or the initial structure of MLP module "*C*" is 16-11-1. Let the learning parameters η=0.00035 and α=0.0075. After finishing the first learning round with 10,000 iteration epochs and 2,775 *sec* (The time for PIV 2.6G CPU, 256M RAM, the same below), the root- mean-squared (RMS) error is 0.1084. The result of SVD for the hidden output matrix $Z^{(C)}$ is (417.47, 64.47, 50.11, 30.97, 24.31, 17.26, 16.64 9.08, 5.67, 4.67, 2.13)^T. According to [5], it is enough to select s=7 hidden nodes. Through the second learning round using a 16-7-1 MLP, iterating 10,000 epochs and taking 1,271.3 *sec*, the final RMS error is 0.0803. The 16-7-1 MLP "*C*" reaches the classification correct rates of 586/594=98.65% for the training subclass "*C*", and 135/147=95.07% for "*C*" in the test set. What a surprised situation is that no one sample from the other classes is misclassified for the whole test set at the moment.

After the real outputs of all MLP modules are amended, the final classification correct rates are 15,803/16,000= 98.88% for the training set, and 3,830/4,000=95.75% for the test set. What is the most important is that through learning the smaller subsets, the MLP modules have the same or even better generalization performances compared with learning the original training set.

Table 1 is the comparisons of recognition accuracy of different methods for the test set. It is obvious that the presented modular MLPs are among the best classifiers.

5 Conclusions

When deciding the separate surface of a certain category, say ω_j, the effects of those farther samples may be negligible. Therefore, a learning subset can be many times smaller than the original training set. Furthermore, this paper proposes the solution of sample disequilibrium problem in training subsets and the correct method of the final outputs of MLP modules. The results of letter recognition show that the presented task decomposition method and modular MLP classifiers can effectively solve the large-scale learning problems.

Acknowledgment

This work is supported by the National Science Foundation of China (NSFC) under Grant No. 60275017, 60373073, and the open project program of the State Key Laboratory of Bioreactor Engineering.

References

1. Setiono R.: Feedforward neural network construction using cross-validation, *Neural Computation*, 13 (2001) 2865-2877
2. Blake C.L., Merz C.J.: UCI Repository of machine learning databases: http://www.ics.uci.edu/~mlearn/ MLRepository.html. Irvine, CA (1998)

3. Gormann R.P., et al: Analysis of hidden units in a layered network trained to classify sonar targets, *Neural Networks*, 1: (1988) 75-89
4. Cho S.B.: Neural-network classifiers for recognizing totally unconstrained handwritten numerals, *IEEE Transactions on Neural Networks*, 8(1) (1997) 43-52
5. Daqi G., Genxing Y.: Influences of variable scales and activation functions on the performances of multilayer feedforward neural networks, *Pattern Recognition,* 36 (4) (2003) 869-878
6. Williamson J.R.: Gaussian ARTMAP: a neural network for fast incremental learning of noisy multidimensional maps, *Neural Networks*, 9(5) (1996) 881-897
7. Frey P.W., Slate D.J.: Letter recognition using Holland-style adaptive classifiers, *Machine Learning*, 6 (1991) 161-182
8. Fogarty T.: First nearest neighbor classification on Frey and Slate's letter recognition problem. *Machine Learning*, 9 (1992) 387-388
9. Bressan M., Guillamet D., Vitria J., Using an ICA representation of high dimensional data for object recognition and classification, *IEEE Conference on CVPR*, HI, USA 1 (2001) 1004-1009
10. Islam M.M., Murase K.: A new algorithm to design compact two-hidden-layer artificial neural networks, *Neural Networks*, 14 (2001) 1265-1274
11. Anand R., Mehrotra K.G., Mohan C.K., et al: Efficient classification for multiclass problems using modular neural networks, *IEEE Transactions on Neural Networks*, 6(1) (1995) 117-124

Fast Color-Based Object Recognition Independent of Position and Orientation

Martijn van de Giessen and Jürgen Schmidhuber

IDSIA, Galleria 2, 6928 Manno (Lugano), Switzerland &
TU Munich, Boltzmannstr. 3, 85748 Garching, München, Germany

Abstract. Small mobile robots typically have little on-board processing power for time-consuming vision algorithms. Here we show how they can quickly extract very dense yet highly useful information from color images. A single pass through all pixels of an image serves to segment it into color-dependent regions and to compactly represent it by a short list of the average hues, saturations and color intensities of its regions; all other information is discarded. Experiments with two image databases show that in 90 % of all cases the remaining information is sufficient for a simple weighted voting algorithm to recognize objects shown in query images, independently of position and orientation and partial occlusions.

1 Introduction

Small, fast, vision-based mobile robots must process many images per second to react in time. It does not matter so much if some object sometimes is not properly recognized the first instant it is seen, provided the robot's sequential vision system (SVS) can use subsequent camera shots to incrementally increase its confidence about whether or not the object is present in its visual field. In principle such an SVS for dealing with uncertainty and noise can be implemented by Bayesian sequential decision makers or learned by recurrent neural networks [1,2,3].

The image pre-processor should be able to quickly produce a compact yet informative description of the current image, to be fed into the SVS. So we are interested in fast algorithms that often (but not necessarily always) produce image descriptions containing all the information necessary for decent object recognition. Of course, the more reliable the pre-processor, the less burden on the SVS.

Many previous approaches to object recognition are computationally too demanding for the limited on-board computers of mobile robots, or permit only small changes in object position and orientation, and few if any occlusions. Here we propose a simple, fast, and rather reliable method based solely on the number of image regions with similar color.

In what follows we will describe two methods, a fast one for image processing and coding (Section 2), and another one for demonstrating that the image codes convey sufficient information for object recognition, given a database of images

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 469–474, 2005.

(Section 3). The latter searches for a match in the database by judging similarity between the query image and database images based on weighted votes. Performance in terms of recognition rate and speed is evaluated in Section 4.

2 Processing Images

2.1 HSV Images

We represent images in HSV color space, which provides a good distinction between three properties of a color: hue, saturation and illumination (value). These properties are related to each other in Figure 1. HSV codes provide simple ways of filtering out non-reliable color information conveyed by areas with low illumination or low saturation (e.g., almost grey areas). The shaded parts in figure 1 contain such non-reliable colors.

2.2 Region Extraction

Since every step in the recognition process should be fast, we use a very simple region extraction algorithm loosely inspired by the intensity-based method proposed by Tuytelaars and Van Gool [4]. From the top left to the bottom right of an image, every pixel j is compared to the regions of its upper and left neighbor, if these neighboring pixels exist (at borders the pixel j is only compared to the regions of its upper or left neighbor, depending on which pixel exists). We ask whether the differences between the hue (h_j), saturation (s_j) and value (v_j) of pixel j and the average hue (h_i), saturation (s_i) and value (v_i) of region i are smaller than the thresholds t_h, t_s and t_v, respectively:

$$|h_j - h_i| < t_h, |s_j - s_i| < t_s, |v_j - v_i| < t_v \tag{1}$$

Note that the hue describes a circle as in figure 1. The pixel j is added to the most similar region or, if inequality 1 does not hold for both regions, a new region is formed. When the pixel j is added to a region, the other region adjacent to j is merged with the region containing j if $|P_j - P_k| < t_p$ holds for hue, saturation and value. P_j and P_k again stand for the average hue, saturation and value of the region containing pixel j and the region adjacent to pixel j respectively. Comparing to the average characteristics of the neighboring region instead of to the neighboring pixel has the advantage that more coherent regions are generated. This is due to the fact that the average values change more slowly when a region grows, so pixels in fast changing gradients are not added to a region.

After all the regions have been extracted, regions with a very small area (e.g. less than 50 pixels) and regions with their average saturation and value in the shaded regions of figure 1 are discarded. One reason why small regions are abandoned is that there is a high chance that they will not appear on a picture of the same object on a different scale or viewed from a different angle. Another reason is that smaller regions are more sensitive to the distortions caused by pixels on the edge of that region. We save the average hue, saturation and value of every region in a database, since this information is completely position and orientation independent.

3 Querying Images

3.1 Initial Selection

The following object recognition procedure is not mandatory for mobile robots that just need to compactly encode images and feed them into an SVS-based controller, without having to match them against a database. But it serves to demonstrate that the image codes retain essential information about the depicted objects. It may be possible to further speed up the straight-forward procedure below, e.g., by rearranging the database in hierarchical form.

To search for images of objects similar to the one in a query image, we first extract the latter's regions as above. Every region in the query image is compared to all regions of all images in the database. To avoid wasting time on computing complex distance measures between very different regions, we first discard those that are clearly dissimilar to the queried region. This is done in a way similar to the one of Nene and Nayar [5]: All regions are considered as points in a 3D space with hue, saturation and intensity (value) on the axes. A box is computed with the query region in the center and the sides perpendicular to the three axes. Database regions outside this box are discarded. Thus groups of similar regions are formed for every region in the query image.

3.2 Weighted Voting

All database regions i within each group get a weighted vote w_{ij}. This vote is determined by the distance

$$d_{ij} = 5^{-\frac{1}{2}}((v_i - v_j)^2 + (s_i \cos h_i - s_j \cos h_j)^2 + (s_i \sin h_i - s_j \sin h_j)^2)^{\frac{1}{2}} \qquad (2)$$

in color between a region i in the database and a region j in the query image. We compensate for the distinctiveness of a query region and the complexity of a database image. The distance between the average hue, saturation and value of two regions is computed as in [6]. The distinctiveness of a region is determined on the basis of how many regions in the database survive after the initial selection in section 3.1. The larger the number of regions in the box, the less distinctive the region. To give more weight to more distinctive regions, we divide by the number of remaining regions, $n_{similar}$. It is also useful to compensate for 'complex' images with many regions, which have a greater chance of featuring a region close to a query region. That is why we divide by the number of regions in the image containing this region, n_{rpi}. The total vote per region becomes

$$w_{ij} = \frac{1 - d_{ij}}{n_{similar} \cdot n_{rpi}} \qquad (3)$$

The total vote for an object is given by the sum of the weighted votes for all regions of that object.

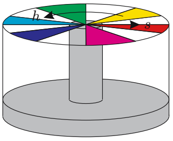


Fig. 1. The HSV color space in cylindrical form. Gray areas contain non-reliable color information.

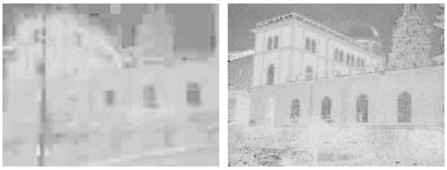


Fig. 2. The color information (hue) in low quality JPEGs is unreliable (left), compared to this information in a high quality image of the same object (right).

4 Experimental Results

4.1 ZuBuD Buildings Database

We tested both recognition rate and speed on a databases containing real-world images, resampled to 320 × 240 pixels, that were not made under special simplifying conditions. Our first choice was the ZuBuD database [7] containing 1005 pictures of 201 buildings in Zürich, taken outside and under varying weather conditions. The database comes with 115 query images (low-quality jpegs, resolution 320 × 240 pixels). For every queried image, we list the top five matches produced by our algorithm. Despite the low quality of color information in the queries, 82 images are recognized correctly, 29 images are within the top five, and 4 outside the top five. The color information in the query images is of a very low quality, as can be seen clearly in Figure 2, where the hue from a query image and a database image from the same building are shown next to each other. This lack of reliable color information prevented a correct recognition. The query image in the top row of Figure 3 exemplifies our algorithm's insensitivity to obstructions, such as trees.

To test performance on high-quality images (easily producible on small mobile robots), we built a new database containing 4 images of each of the 201 objects, using the 5th image of every object as a query image. The improved image quality led to substantially improved recognition rate: 183 objects correctly recognized, 13 in the top 5 matches, and only 5 out of 201 outside the top 5. The second row of figure 3 shows our algorithm's insensitivity to orientation and position.

The reasons for the five failures and some of the non-perfect results in the top five seem to be twofold. One reason is that these database and/or query images mainly have regions with very low saturation. The second is that some of these query images contain only a rather small part of the object. The last row of figure 3 shows one of the misses due to low saturation. For a learning robot with an SVS based on adaptive recurrent neural networks [3] these misses will not pose a big problem, since they will be identified as noise by the learning algorithm.

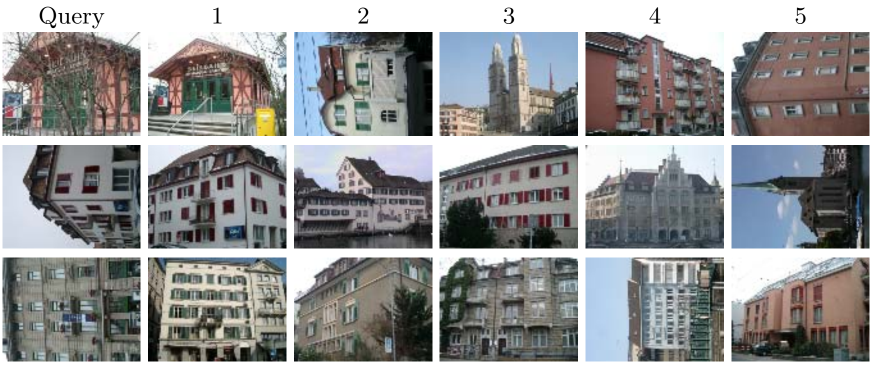


Fig. 3. Three examples of query images (left) and the top five similar objects according to our recognition algorithm. The top row shows the robustness against occlusions, the middle row shows the independence of position and orientation and the third row shows a miss, because of the low saturation of the object in the query image.

4.2 Coil-100 Object Database

Our method is *not* specialized on the ZuBuD database. It is designed to be widely applicable. We hardly tuned any parameters; one just has to select the tresholds of Section 3.1 as small as possible to speed up the recognition process. To illustrate the method's generality, we also applied it to the Coil-100 database [8], which contains 100 objects photographed from various angles. We placed images of objects taken under angles 0, 100, 215, 270 and 325 degrees in the database and used the 25 degree views as query images. Results: 84 recognized objects, 12 in the top 5, 4 outside the top 5. Again the non-recognized images mainly contain areas in various shades of gray, discarded for their poor color information.

4.3 Performance in Terms of Speed

We report results for a standard Pentium 4 2.8 GHz machine. The two time consuming parts of the recognition process are region extraction and the search for similar regions in the database. The latter is not mandatory for robots that just need to compress the relevant information for an SVS. The speed of the former depends on image complexity, but even complex images like those in the ZuBuD database were processed within 0.19s.

Database search speed linearly depends on the number of regions in the database and on the number of regions found in the query image. The database for the second experiment contains 93682 regions. On average 177 regions are extracted from an image in the ZuBuD database. An average search takes roughly 0.7s, including region extraction in the query image. Simple objects speed up the process: Recognizing one of the 100 objects in the Coil-100 database takes roughly 0.18s, including query image processing.

5 Conclusions

We propose a simple, fast, rather reliable algorithm for image coding and recognition. It uses a simple color-based region extractor and an object matcher based on weighted voting. The former works in HSV color space, discarding regions with unreliable color information, keeping only reliable, position and orientation independent color data to encode objects. Image similarity is measured by votes whose weights depend on the similarity between the regions of a query image and database images; we compensate for the number of similar regions and regions per database image. On the ZuBuD and Coil-100 databases we obtain satisfactory recognition rates and speeds.

It should be noted, however, that this work is quite preliminary; more in depth studies are necessary to compare our algorithm to previous proposals in the literature. This is the subject of ongoing work.

Acknowledgements

We would like to thank Hao Shao for supplying code used by his alternative color-based image recognition algorithm [9]. It gave us a fine starting point for the development of the present algorithm.

References

1. Williams, R.J., Zipser, D.: Gradient-based learning algorithms for recurrent networks and their computational complexity. In: Back-propagation: Theory, Architectures and Applications. Hillsdale, NJ: Erlbaum (1994)
2. Pearlmutter, B.A.: Gradient calculations for dynamic recurrent neural networks: A survey. IEEE Transactions on Neural Networks **6** (1995) 1212–1228
3. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9** (1997) 1735–1780
4. Tuytelaars, T., Van Gool, L.: Wide baseline stereo matching based on local, affinely invariant regions. In: British Machine Vision Conference. (2000)
5. Nene, S.A., Nayar, S.K.: A simple algorithm for nearest neighbor search in high dimensions. IEEE Transactions on Pattern Analysis and Machine Intelligence **19** (1997)
6. Smith, J.R., Chang, S.F.: Visualseek: A fully automated content-based image query system. In: ACM Multimedia. (1996) 87–98
7. Shao, H., Svoboda, T., Van Gool, L.: ZuBuD — Zürich buildings database for image based recognition. Technical Report 260, Computer Vision Laboratory, Swiss Federal Institute of Technology (2003) Database downloadable from http://www.vision.ee.ethz.ch/showroom/.
8. Nene, S., Nayar, S., Murase, H.: Columbia object image library: Coil-100. Technical Report CUCS-006-96, Department of Computer Science, Columbia University (1996)
9. Shao, H., Svoboda, T., Tuytelaars, T., Van Gool, L.: Hpat indexing for fast object/scene recognition based on local appearance. In Lew, M., Huang, T., Sebe, N., Zhou, X.S., eds.: Computer lecture notes on image and video retrieval. LNCS 2728, Springers (2003) 71–80

Class-Specific Sparse Coding for Learning of Object Representations

Stephan Hasler, Heiko Wersing, and Edgar Körner

Honda Research Institute Europe GmbH,
Carl-Legien-Str. 30, 63073 Offenbach am Main, Germany
stephan.hasler@honda-ri.de

Abstract. We present two new methods which extend the traditional sparse coding approach with supervised components. The goal of these extensions is to increase the suitability of the learned features for classification tasks while keeping most of their general representation performance. A special visualization is introduced which allows to show the principal effect of the new methods. Furthermore some first experimental results are obtained for the COIL-100 database.

1 Introduction

Sparse coding [4] searches for a linear code representing the data. Its target is to combine efficient reconstruction with a sparse usage of the representing basis, resulting in the following cost function:

$$P_{\mathrm{S}} = \frac{1}{2}\sum_i \left(\mathbf{x}_i - \sum_j c_{ij}\mathbf{w}_j\right)^2 + \gamma \sum_i \sum_j \Phi\left(c_{ij}\right) \ . \tag{1}$$

The left reconstruction term approximates each input $\mathbf{x}_i = \left(x_{i1}, x_{i2}, \dots, x_{iK}\right)^T$ by a linear combination $\mathbf{r}_i = \sum_j c_{ij}\mathbf{w}_j$ of the weights $\mathbf{w}_j = \left(w_{j1}, w_{j2}, \dots, w_{jK}\right)^T$, where $\mathbf{r}_i$ is called the reconstruction of the corresponding $\mathbf{x}_i$. The coefficients c_{ij} specify how much the j^{th} weight is involved in the reconstruction of the i^{th} data vector. The right sparsity term sums up the c_{ij}. The nonlinear function Φ (e.g. $\Phi(\cdot) = |\cdot|$) increases the costs, the more the activation is spread over different c_{ij}, and so many of them become zero while few are highly activated. The influence of the sparsity term is scaled with the positive constant γ.

An adaptation of the sparse coding is the nonnegative sparse coding [7]. In this approach the coefficients and the elements of the weights are kept positive. This forces the weights to become more distinct and produces a parts-based representation similar to that obtained by nonnegative matrix factorization [2] with sparseness constraints [1].

In Sect. 2 we introduce two new class-specific extensions of the nonnegative sparse coding. We visualize their principal effect with a simple 2D example to analyze the influence of certain parameters of the cost functions. Furthermore some first experimental results are obtained for the COIL-100 database [3] in Sect. 3. Section 4 gives a short conclusion of the results.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 475–480, 2005.

2 Class-Specific Sparse Coding

The sparse coding features are useful for general image representation but lack the property of being class-specific, and so their use in classification tasks is limited. Our two new approaches extend the nonnegative sparse coding with supervised components. In the first approach the class information has direct effect on the coefficients and it will therefore be referred to as coefficient coding:

$$P_{\mathrm{C}} = \frac{1}{2}\sum_{i}\left(\mathbf{x}_i - \sum_{j} c_{ij}\mathbf{w}_j\right)^2 + \gamma\sum_{i,j} c_{ij} + \frac{1}{2}\,\alpha\sum_{j}\sum_{\substack{i,\tilde{\imath}\\ Q_i \neq Q_{\tilde{\imath}}}} c_{ij}c_{\tilde{\imath}j}\,. \qquad (2)$$

The right coefficient term causes costs if coefficients belonging to the same weight $\mathbf{w}_j$ are active for representatives $\mathbf{x}_i$ and $\mathbf{x}_{\tilde{\imath}}$ of different classes Q_i and $Q_{\tilde{\imath}}$ respectively. Q_i stands for the class of a data vector $\mathbf{x}_i$. The influence of the coefficient term is scaled with the positive constant α. In the second approach the class information has a more direct effect on the weights and it will therefore be referred to as weight coding:

$$P_{\mathrm{W}} = \frac{1}{2}\sum_{i}\left(\mathbf{x}_i - \sum_{j} c_{ij}\mathbf{w}_j\right)^2 + \gamma\sum_{i,j} c_{ij} + \frac{1}{2}\,\beta\sum_{j}\sum_{\substack{i,\tilde{\imath}\\ Q_i \neq Q_{\tilde{\imath}}}} \left(\mathbf{x}_i^T\mathbf{w}_j\right)\left(\mathbf{x}_{\tilde{\imath}}^T\mathbf{w}_j\right)\,. \qquad (3)$$

The right weight term is similar to a linear discriminator and causes costs if a $\mathbf{w}_j$ has a high inner product with representatives $\mathbf{x}_i$ and $\mathbf{x}_{\tilde{\imath}}$ of different classes Q_i and $Q_{\tilde{\imath}}$ respectively. The weight term is scaled with the positive constant β.

The minimization of both cost functions is done by alternately applying coefficient and weight steps as described in [7]. In the coefficient step the cost function is minimized with respect to the c_{ij} using an asynchronous fixed-point search, while keeping the $\mathbf{w}_j$ constant. The weight step is a single gradient step with a fixed step size in the $\mathbf{w}_j$, keeping the c_{ij} constant.

In Fig. 1 we introduce our special visualization schematically and apply it to the nonnegative sparse coding, and in Fig. 2 it is used to compare coefficient and weight coding.

The coefficient coding restricts the use of features through different classes. This means that each feature concentrates on a single class and so the influence of other classes is weakened. There is no increase in discriminative quality, because this would require a strong interaction of different classes. The weight coding instead has a more direct effect on the features and removes activation from them, that is present in different classes. So the features represent more typical aspects of certain objects and so their suitability for object representation and recognition is increased. The cost function shows similarity to Fishers linear discriminant and the MRDF approach [5] but does not minimize the intra class variance. The advantage of the weight coding is that it can produce an overcomplete representation while the number of features in the Fisher linear discriminant is limited by the number of classes and in the MRDF by the num-

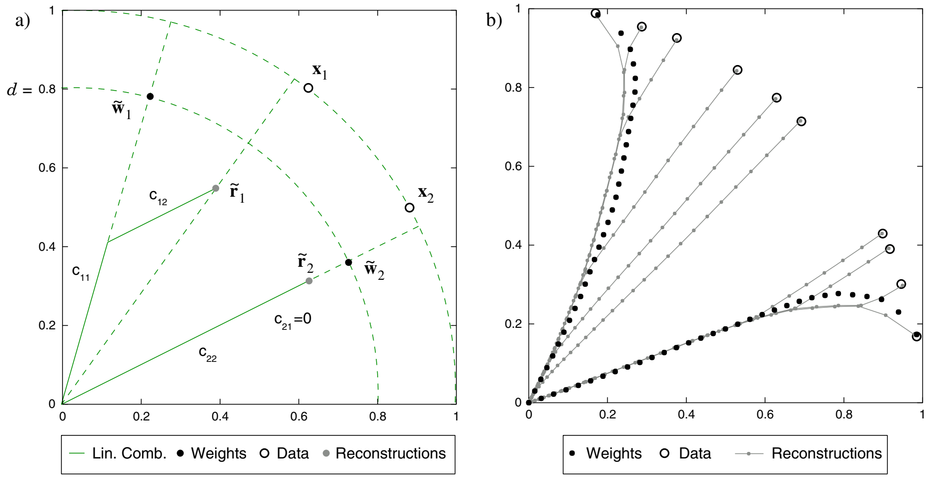


Fig. 1. a) Schematic description of visualization. In the visualization the positions of the data vectors, their reconstructions and the weights are plotted for different values of a control parameter of the cost function, e.g. $\gamma = \gamma_{min} \dots \gamma_{max}$. The data vectors $\mathbf{x}_i$ lie on the unit circle. The shown $\widetilde{\mathbf{w}}_j = d\mathbf{w}_j$ are the weights which have been scaled by a factor $d = (\gamma_{max} - \gamma)/(\gamma_{max} - \gamma_{min})$. Similarly the $\widetilde{\mathbf{r}}_i = d\mathbf{r}_i = d\sum_j c_{ij}\mathbf{w}_j$ are the reconstructions scaled by the same factor. The scaling causes the $\widetilde{\mathbf{r}}_i$ and the $\widetilde{\mathbf{w}}_j$ to move towards the origin with increasing γ. This simply should increase the ability to distinguish the points belonging to different values of γ (see b). Because the weights $\mathbf{w}_j$ are normalized, the distance of the $\widetilde{\mathbf{w}}_j$ from the origin is as big as the scaling factor d. The distance of the $\widetilde{\mathbf{r}}_i$ from the origin is also influenced by the cost function itself. In the nonnegative case it is always shortened, since low values of the basis coefficients c_{ij} are enforced. From the position of the $\widetilde{\mathbf{r}}_i$ and the $\widetilde{\mathbf{w}}_j$ it is possible to determine the coefficients c_{ij} and hence to judge the sparsity of the reconstructions of certain $\mathbf{x}_i$. For example $\mathbf{x}_2$ is reconstructed sparsely, because it only uses $\mathbf{w}_2$. In the visualization the linear combinations will not be plotted and there will be points for each $\widetilde{\mathbf{r}}_i$ and $\widetilde{\mathbf{w}}_1$, $\widetilde{\mathbf{w}}_2$ for each value of γ. The points belonging to the same value of γ could be determined by counting, preferably from the unit circle to the origin. b) Visualization of the influence of the parameter γ on the nonnegative sparse coding. Two scaled weights and the scaled reconstructions of 10 data vectors are plotted for 31 different influences γ of the sparsity term. The scaling factor is $d = (\gamma_{max} - \gamma)/(\gamma_{max} - \gamma_{min})$. The reconstructions belonging to successive values of γ are connected. For $\gamma = \gamma_{min} \to 0$ the algorithm searches for the sparsest perfect reconstruction. And since $d = 1$ in this case the $\widetilde{\mathbf{r}}_i$ lie directly on the $\mathbf{x}_i$. The corresponding $\widetilde{\mathbf{w}}_j$ are aligned with the outermost $\mathbf{x}_i$. With increasing γ the points move towards the origin. Each $\widetilde{\mathbf{r}}_i$ gives up the use of the less suitable weight and therefore the $\widetilde{\mathbf{r}}_i$ unite to two main paths. Each $\widetilde{\mathbf{w}}_j$ aligns to the center of the $\widetilde{\mathbf{r}}_i$ which are assigned to it.

ber of dimensions in the data. The two parameters have to be chosen carefully. When the influence of the sparsity term is too weak, many features tend to represent the same activation.

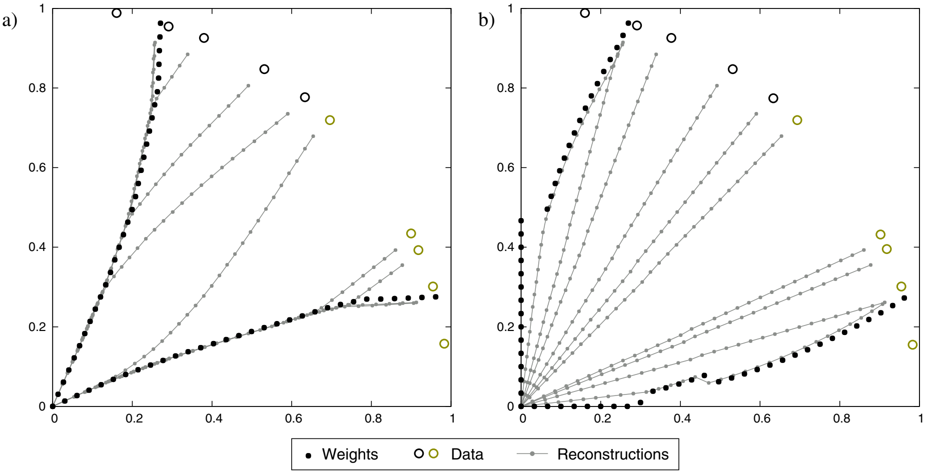


Fig. 2. Visualization of the influence of the parameters α and β on the coefficient and weight coding. The 10 data vectors are now assigned to 2 classes and again are reconstructed using two weights. The influence of the sparsity term is the same constant value $\gamma = 0.05$ in both cases. a) In the coefficient coding the influence α of the coefficient term varies in a defined range. The scaling is $d = (\alpha_{max} - \alpha)/(\alpha_{max} - \alpha_{min})$. With increasing α the points are pulled to the origin. Each weight is forced to specialize on a certain class and therefore moves to the center of this class. There is no gain in discriminative power. b) In the weight coding the influence β of the weight term varies in a defined range. The scaling is $d = (\beta_{max} - \beta)/(\beta_{max} - \beta_{min})$. With increasing β the points are pulled to the origin. The suitability of the weights for different classes is reduced and each aligns to activation which is most typical for the class it is representing. So one weight moves towards the x-axis and the other one towards the y-axis. In the nonnegative case this can be referred to as a gain in discriminative power.

3 Experimental Results

To underpin the qualitative difference between coefficient coding and weight coding both approaches have been applied to a more complex problem. Nine car objects and nine box objects from the COIL-100 database [3], each with 72 rotation views, were combined to two classes (see Fig. 3). Fifty features were trained using the same influence $\gamma = 0.1$ of the sparsity term and relative high

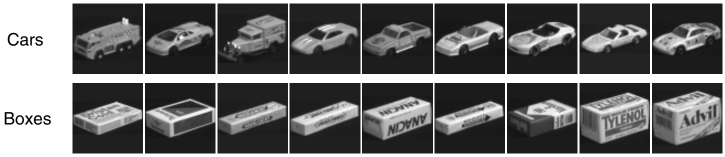


Fig. 3. Two class problem. Nine cars and nine boxes were combined to two classes.

Table 1. The table shows the values of the different terms, neglecting their actual influences (γ, α, and β)to the cost functions. Note that values in brackets were not used for optimization, but are shown to highlight qualitative differences between features.

	Reconstruction	Sparsity	Coefficient	Weight
Nonneg. Sparse Coding	$4.182 \cdot 10^4$	$4.293 \cdot 10^4$	$(1.726 \cdot 10^7)$	$(1.428 \cdot 10^{10})$
Coefficient Coding	$4.682 \cdot 10^4$	$3.872 \cdot 10^4$	$5.709 \cdot 10^5$	$(1.731 \cdot 10^{10})$
Weight Coding	$4.031 \cdot 10^4$	$4.976 \cdot 10^4$	$(2.393 \cdot 10^7)$	$1.035 \cdot 10^{10}$

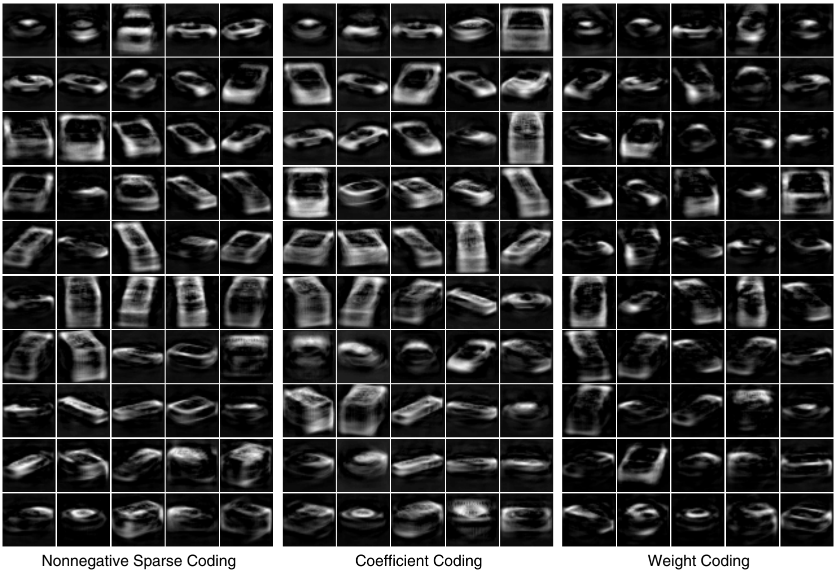


Fig. 4. Features for three approaches. For the visualization, we arranged the features in the following way: Each feature is assigned to the class in which it is most often detected. A feature is detected if its normalized cross-correlation with an image exceeds a feature-specific threshold. This threshold was determined as to maximize the mutual information conveyed by the detection of the feature about the classes (see [6]). The more car-like features start at the top-left and the more box-like features at the bottom-right. The features in one class are arranged by descending mutual information. Therefore the least informative features of the two classes meet somewhere in the middle. The features of the nonnegative sparse coding and the coefficient coding are very similar to each other. The features of the weight coding are sparser and concentrate on more typical class attributes, like certain parts of the cars or the vertices of the boxes.

values for the influence $\alpha = 1 \cdot 10^{-3}$ of the coefficient term and the influence $\beta = 5 \cdot 10^{-7}$ of the weight term.

The resulting features are shown in Fig. 4. Table 1 lists the values of the terms of the cost functions after optimization. These values are useful to interpret the effect of our two new approaches compared to the nonnegative sparse coding:

Since the coefficient term puts a penalty on the use of features across different classes, a splitting into partial class problems is to be observed, leading to a reduced feature basis for each class. As a result, there is an increase of the reconstruction costs and a decrease of the sparsity costs. The demand for sparsity of the coefficients in the nonnegative sparse coding has an opposite effect on the weights, forcing them to become very view-specific and leading to a higher reconstruction cost. In the weight coding the weight term removes activation from the features. They become less view-specific, which causes a decrease of the reconstruction costs, but an increase of the sparsity costs.

4 Conclusion

In this paper two new class-specific extensions of the nonnegative sparse coding were introduced. It was shown that the coefficient coding, by restricting the use of features through different classes, does not increase the discriminative quality of the features, but instead tends to cause a splitting into partial problems, using a distinct feature basis for representing each class. In contrast to that, the weight coding directly penalizes the suitability of features for different classes and so successfully combines representative and discriminative properties. This combination produces features which are more suitable for object representation than features with general representative quality only. The advantage of the weight coding is that it can produce an overcomplete representation, whereas most other approaches are using the covariance matrix directly, and so the number of features is limited by the number of dimensions in the data. The drawback is that two parameters have to be tuned suitably. Also the intra class variance is not reduced as e.g. in the MRDF approach. The evaluation of the usefulness of the weight coding features in object recognition will be subject to further investigations.

References

1. Hoyer, P.: Non-negative Matrix Factorization with Sparseness Constraints. Journal of Machine Learning Research **5** (2004) 1457-1469
2. Lee, D.L., Seung, S.: Learning the parts of objects by non-negative matrix factorization. Nature **401** (1999) 788–791
3. Nayar, S.K., Nene S.A., Murase H.: Real-time 100 object recognition system. In Proc. IEEE Conference on Robotics and Automation **3** (1996) 2321–2325
4. Olshausen, B., Field, D.J.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature **381** (1996) 607–609
5. Talukder, A., Casasent, D.: Classification and Pose Estimation of Objects using Nonlinear Features. In Proc. SPIE: Applications and Science of Computational Intelligence **3390** (1998) 12–23
6. Ullman, S., Bart, E.: Recognition invariance obtained by extended and invariant features. Neural Networks **17(1)** (2004) 833–848
7. Wersing, H., Körner, E.: Learning Optimized Features for Hierarchical Models of Invariant Object Recognition. Neural Computation **15(7)** (2003) 1559–1588

Neural Network Based Adult Image Classification*

Wonil Kim[1], Han-Ku Lee[2,**], Seong Joon Yoo[1], and Sung Wook Baik[1]

[1] College of Electronics and Information Engineering,
Sejong University, Seoul,
Republic of Korea
{wikim, sjyoo, sbaik}@sejong.ac.kr
[2] School of Internet and Multimedia Engineering,
Konkuk University, Seoul,
Republic of Korea
hlee@konkuk.ac.kr

Abstract. Digital multimedia data is dramatically being increased everyday since the Internet became popular. This increment in multimedia data increases adult image contents to the Internet as well. Consequently, a large number of children are exposed to these X-rated contents. In this paper, we propose an efficient classification system that can categorize input images into adult or non-adult images. The simulation shows that this system achieved 95% of the true rate whereas it reduces the false positive rate below 3%.

1 Introduction

The Internet has made people access more information than any time before and has become the major source of information. In the other hand, it also shows the dark side. Among the millions of Web sites, there are over 500,000 web sites related to pornography and other issues that are as harmful as poison to the children [1].

We propose a neural network based classification system that can classify input images into adult or non-adult images. The visual descriptors defined by MPEG-7 are used for extracting features for a given image. These features are used as inputs for the 2 class (adult, non-adult) neural network classification system. The simulation using the Color Structure descriptor shows that the system achieved 95% of the true rate, whereas it reduces the false positive rate below 3%.

We will review several feature extraction and the image classification methods in the next section. The proposed image classification system using MPEG7 descriptors are detailed in the section 3. The simulation results and analyses of the image classifier will be discussed in the section 4. Finally the conclusion is given in section 5.

* This paper was supported by Electronics and Telecommunications Research Institute (Project No. 0801-2004-0025).

** Author for correspondence : +82-2-2049-6089.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 481 – 486, 2005.

2 Related Works

There are two major approaches for adult image processing; one heavily depends upon a classic retrieval technique with image featuring, and another depends on an image mining technique, distinguishing adult images from normal images, using decision-boundary lines, rather than image featuring. The important point of the former is the feature extraction that is the very previous step of the image classification and that of the latter is the classifier itself, rather than the features being input to the classifier.

Most of the previous works for image rating systems does not specify these two different approaches and just presents the systems as one module. This paper discusses two different image processing techniques, and proposes a novel method that maximizes advantages of each technique and minimizes disadvantages.

2.1 Feature Extraction

In the middle of 90s, researchers and engineers became interested in detecting adult images on computers when the World Wide Web (WWW) became popular. The first algorithm to detect naked people in images was researched in [2, 8]. The main idea of the research is effectively masking skin regions using the skin filter. If skin regions that passed mask tests are matched to persons' figures, it is assumed that there are numerous naked parts in the image (e.g. the geometric filter). The algorithm detects whether or not the filtered skin color is matched to a specific body part in the whole image, rather than extracts primitive features helping effective classification.

Another approach had researched in [7]. The algorithm was based on image contents, and classified adult images by extracting skin regions and feature vectors that are useful for the image classification. The feature vector consists of color, texture, contour, placement, and relative size information for a given region. The advantages of this algorithm are that the importance of features is decided by the generic algorithm and the feature extraction is systematically (not experimentally) executed. Especially this algorithm is good to be applied to web sites rather than adult images. For instance, as a result of adopting the algorithm on 20 sites, the success rate is 89% and 11 images per second are executed.

The color histogram is one of the most common methods in the image classification. It is an effective method for the large size of data. A simple experiment based on the color histogram results in 80% of the detection rate and 8.5% of the false positive rate.

[6] proposed an efficient feature extraction system using MPEG-7 descriptor for the adult image classification. The system used three descriptors out of visual descriptors standardized by MPEG-7; edge histogram descriptor (EHD), color layout descriptor (CLD), homogeneous texture descriptor (HYD). They are effective to the adult image classification. The system also has the image database, which compares the descriptor value of a given image to that of images from the database, and retrieves 10 similar images with class information. The class of the given image is classified as the class that the majority of image is classified. Even though the results

showed that MPEG-7 descriptors can be used as very efficient features in the adult image classification, the classification method used in his paper is merely the *k*-nearest neighbor method.

The MPEG-7 descriptor is applied for extracting features in this paper. MPEG-7 has been developed by distinguished researchers for a long time, and has much effective definitions of descriptors. The visual area of MPEG-7 includes the color descriptor, the shape descriptor, the texture descriptor, and the movement descriptor. In the step of the feature extraction, the proposed system concentrates on finding descriptors to be used for the input of neural networks, rather than simply extracting values of features for descriptors.

2.2 Image Classification

As database techniques rapidly became advanced, image classification techniques that use statistical methods have improved a lot [3]. In particular, the field of data mining has been much improved, and a new field of study has appeared; image mining [4]. Thus, many research groups study and research the field of image classification via the image mining technique. The image classification can be categorized as the Neural Network, the Decision-Tree Model, and the Support Vector Machine.

Neural Network based method is the most common technique. This method concentrates on the study of decision-boundary surface telling adult images from non-adult images via the computer-based classification rule, called perceptron [10, 11]. An Artificial Neural Network (ANN) is an information processing paradigm inspired by biological nervous systems, such as the brain process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by examples. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons.

The decision tree model recursively partitions an image data space, using variables that can divide image data to most identical numbers among a number of given variables. This technique can give incredible results when characteristics and features of image data are known in advance [9].

The support vector machine technique is a brand-new image classification method. The purpose of the method is to find decision lines or surfaces distinguishing data from others like the technique using neural networks. The technique using the neural networks is just to find decision surfaces classifying the training data. But, SVM is to find decision surfaces maximizing the distance of two sets. Jiao et al. experimented on adult image classifiers using SVM [5].

We employ the Neural Network for the classification module in this paper. Inputs for the neural network are fed from the feature values extracted from MPGE-7 descriptors. Since the various descriptors can represent the specific features of a given image, the proper evaluation process should be required to choose the best one for the adult image classification.

3 Proposed System

3.1 Overall System Flow

The system uses the MPEG-7 XM program in the extraction phase to extract features from images in the database [12]. In the image database, there are about 9000 images (adult: 4500, non-adult: 4500) for training and testing. The system can use 6 descriptors; Dominant Color, Color Structure, Color Layout, Edge Histogram, Homogeneous Texture, and Region Shape. In the feature extraction module, the feature information is generated for each descriptor to extract the necessary features for the image classification. In the neural network classification step, the extracted descriptors are used for the input value and the neural networks are trained according to the 2 classes (adult and normal images).

3.2 Feature Extraction

By executing the MPEG-7 XM program, the features of training images are extracted in XML format. The feature information in XML is parsed and normalized into values between 0 and 1 with respect to values generated by each descriptor. These normalized values are used as inputs for the neural network classifier. After the phase of extracting input data used in the neural network by MPEG-7 XM, each image is attached with class information.

3.3 Classification

The neural network classifier is trained for the relation of the feature values and the corresponding class by modifying the weight values between nodes. We use the backpropagation algorithm to train the network. The classifier consists of input layer, output layer, and multiple hidden layers. The number of input nodes depends on the dimension of each descriptor, whereas the number of output nodes is two. The class information for the two output nodes is represented as (1,0) for adult images and (0,1) for normal images. In the testing process, as in the training process, the system extracts features from query images using MPEG-7 descriptors, and classifies query images using the neural network that generated by the training process.

4 Simulation and Result

In the simulation, we use MPEG-7 reference software: the e**X**perimentation **M**odel for feature extraction. The eXperimentation Model (XM) software is the simulation platform for the MPEG-7 Descriptors (Ds), Description Schemes (DSs), Coding Schemes (CSs), and Description Definition Language (DDL). Besides the normative components, the simulation platform needs some non-normative components, essentially to execute some procedural code to be executed on the data structures. The data structures and the procedural code together form the applications [12].

We simulated the 2-class image classification using 6 descriptors, such as Dominant Color, Color Structure, Color Layout, Edge Histogram, Homogeneous

Texture, and Region Shape. The classification module consists of 2 hidden layers, each with 10 nodes. The learning rate is 0.001 and the iteration number for training is 100,000.

Table 1. True / False rates of each Descriptor

Color layout

	Positive	Negative	Total
True (rate)	2107 (37.391)	2618 (46.46)	4725 (83.851)
False (rate)	207 (3.673)	703 (12.476)	910 (16.149)

Color structure

	Positive	Negative	Total
True (rate)	2551 (47.267)	2570 (47.619)	5121 (94.886)
False (rate)	143 (2.65)	133 (2.464)	276 (5.114)

Dominant color

	Positive	Negative	Total
True (rate)	1106 (29.082)	1524 (40.074)	2630 (69.156)
False (rate)	109 (2.866)	1064 (27.978)	1173 (30.844)

Edge histogram

	Positive	Negative	Total
True (rate)	2940 (57.908)	1545 (30.432)	4485 (88.34)
False (rate)	244 (4.806)	348 (6.854)	592 (11.66)

Homogeneous texture

	Positive	Negative	Total
True (rate)	493 (9.053)	2626 (48.219)	3119 (57.272)
False (rate)	108 (1.983)	2219 (40.745)	2327 (42.728)

Region shape

	Positive	Negative	Total
True (rate)	2033 (36.239)	1650 (29.411)	3683 (65.65)
False (rate)	1159 (20.66)	768 (13.69)	1927 (34.35)

Table 2. Test of Neural Network

Descriptor (dimension)	Color Layout (12)	Color Structure (256)	Dominant Color (24)	Edge Histogram (80)	Homogeneous Texture (30)	Region Shape (35)
Total	5635	5397	3803	5076	5446	5610
Correct	4725	5121	2630	4484	3119	3683
Incorrect	910	276	1173	592	2327	1927
Rate (%)	83.851	94.886	69.156	88.34	57.272	65.65

After training the network with 2600 images for each descriptor, we tested the performance of the classifier using about 5500 unused images in the testing process. Table 1 shows that the proposed image classification system performs about 95% of the true rate, whereas it reduces the false positive rate below 3 % for the Color Structure descriptor. The other descriptors, such as Edge Histogram descriptor and Color Layout descriptor, also show above par performance in both true positive and false positive rates. Table 2 shows that generally the classification using Color Layer, Color Structure, and Edge Histogram performs much better than using Homogeneous Texture, Region Shape, and Dominant Color.

5 Conclusion

This paper proposed a novel approach of applying MPEG-7 to adult image filtering systems as well as created a prototype system that can be used for the adult image classification by analyzing MPEG-7 descriptors. The visual descriptors defined by MPEG-7 are used for extracting features for a given image. These features are used as inputs for the 2 class (adult, non-adult) neural network classification system. The simulation using the Color Structure descriptor shows that the system achieved 95% of the true rate, whereas it reduces the false positive rate below 3%.

References

1. Will Archer Arentz and Bjorn Olstad, "Classifying offensive sites based on image contents", Computer Vision and Image Understanding, Vol 94, pp293-310, 2004
2. Margaret Fleck, David Forsyth, and Chris Bregler. "Finding Naked People", 1996 European Conference on Computer Vision, Vol. II, pp592-602, 1996
3. Michael J. Jones and James M. Rehg, "Statistical color models with application to skin detection", Technical Report Series, Cambridge Research Laboratory, December 1998
4. Yuna Jung, E. Hwang, Wonil Kim, Sports Image Classifier based on Bayesian Classification, Lecture Note in Artificial Intelligence 3040, 546-555, Springer, June 2004
5. Feng Jiao, Wen Giao, Lijuan Duan, and Guoqin Cui, “Detecting adult image using multiple features”, IEEE conference, Nov. 2001, pp.378 - 383 vol.3.
6. Sung-joon Yoo, “Intelligent multimedia information retrieval for identifying and rating adult images”, Lecture Note in Computer Science 3213, 165-170, Springer, June 2004
7. Mohamed Hammami, Youssef Chahir, and Liming Chen, "WebGuard: Web based adult content detection and filtering system", Proc. IEEE/WIC International Conference on Web Intelligence, pp 574-578, 2003
8. David Forsyth and Margaret Fleck, "Identifying nude pictures", IEEE Workshop on the Applications of Computer Vision 1996, pp103-108, 1996
9. David Hand, Heikki Mannila, and Padhraic Smyth, "Principles of Data Mining", MIT Press, pp343-347, 2001
10. F. Rosenblatt, "The Perceptron : A probabilistic model for information storage and organization in brain", Psychology Review 65, pp386-408
11. D.E. Rumelhart, G.E. Hinton, and R.J. Williams. "Learning representations by back-propagating errors", Nature(London), Vol. 323, pp533-536
12. http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm

Online Learning for Object Recognition with a Hierarchical Visual Cortex Model

Stephan Kirstein, Heiko Wersing, and Edgar Körner

Honda Research Institute Europe GmbH,
Carl Legien Str. 30,
63073 Offenbach am Main, Germany
{stephan.kirstein, heiko.wersing, edgar.koerner}@honda-ri.de

Abstract. We present an architecture for the online learning of object representations based on a visual cortex hierarchy developed earlier. We use the output of a topographical feature hierarchy to provide a view-based representation of three-dimensional objects as a form of visual short term memory. Objects are represented in an incremental vector quantization model, that selects and stores representative feature maps of object views together with the object label. New views are added to the representation based on their similarity to already stored views. The realized recognition system is a major step towards shape-based immediate high-performance online recognition capability for arbitrary complex-shaped objects.

1 Introduction

Although object recognition is a long-studied subject in computer vision, the main focus in research has been so far on achieving optimal recognition performance on selected data sets of object images. Since the training of a recognition system is normally done offline, the efficiency of learning with regard to the learning speed has been considered less relevant in most approaches, leading to typical training times from several minutes to hours. Another problem is that most powerful classifier architectures like the multi layer perceptrons or support vector machines do not allow online training with the same performance as for offline batch training. Due to the lack of rapid learning methods for complex shapes, research in man-machine interaction for robotics dealing with online learning of objects has mainly used histogram-based feature representations that offer fast processing [5,1], but only limited representational and discriminatory capacity. An interesting approach to online learning for object recognition was proposed by Bekel et al. [2]. Their VPL classifier consists of feature extraction based on vector quantisation and PCA and supervised classification using a local linear map architecture. Image acquisition is triggered by pointing gestures on a table, and is followed by a training phase taking some minutes.

We suggest to use a strategy similar to the hierarchical processing in the ventral pathway of the human visual system to speed up the object learning process considerably. The main idea is to use a sufficiently general feature representation that remains unchanged, while object-specific learning is accomplished only in

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 487–492, 2005.

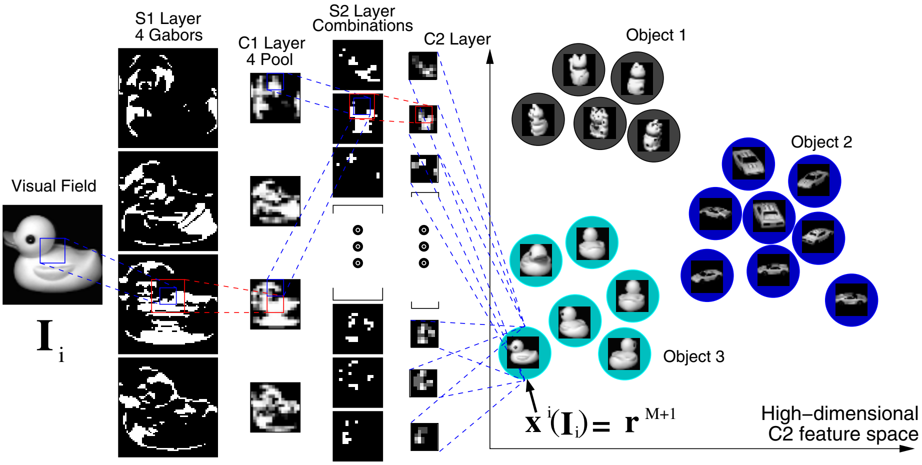


Fig. 1. The visual hierarchical network structure. Based on an image $\mathbf{I}_i$, the first feature-matching stage S1 computes an linear sign-insensitive receptive field summation, a Winner-Take-Most mechanism between features at the same position and a final threshold function. We use Gabor filter receptive fields, to perform a local orientation estimation in this layer. The C1 layer subsamples the S1 features by pooling down to a quarter of the original resolution in both directions using a Gaussian receptive field and a sigmoidal nonlinearity. The features in the intermediate layer S2 are sensitive to local combinations of the features in the planes of the C1 layer, and are thus capable of detecting more complex feature combinations in the input image. We use sparse coding for unsupervised training of these so-called combination feature neurons. A second pooling stage in the layer C2 again performs spatial integration and reduces the resolution by one half in both directions. Object representatives are learnt using an incremental vector quantization approach with attached class labels. Representatives $\mathbf{r}^k$ are computed as the output $\mathbf{x}^i(\mathbf{I}_i)$ of the hierarchy and added based on sufficient Euclidean distance in the C2 feature space to previously stored $\mathbf{r}^k$ of the same object.

the highest levels of the hierarchy. We perform online learning of objects using a short-term memory and similarity-based incremental collection of templates using the intermediate level feature representation of the proposed visual hierarchy from [6]. After a short introduction to our processing and memory model in Sect. 2, we demonstrate its effectiveness for an implementation of real-time online object learning in Sect. 3, and give our conclusions in Sect.4.

2 Hierarchical Visual Processing Model

Model Architecture. The visual hierarchical model proposed in [6] is based on a feed-forward architecture with weight-sharing [3] and a succession of feature-sensitive and pooling stages (see Fig.1). For a comparision to other recent feed-forward models of recognition see [6]. The output of the sparse feature representation of the complex feature layer (C2) is used to incrementally build up the appearance-based object representation with an incremental vector quantisation model. These extracted C2 features are sensitive to coarse local edge combinations like e.g. t-junctions and corners. Given a set of N input images $\mathbf{I}_i$, $i = 1, \ldots, N$, the feature map outputs of the C2 layer of the hierarchy are computed as $\mathbf{x}^i(\mathbf{I}_i)$. The labeled object information is stored in a set of M representatives $\mathbf{r}^k$, $k = 1, \ldots, M$, that are incrementally collected. We define R_l as a set of representatives $\mathbf{r}^k$ that belong to object l. The acquisition of templates is based on a similarity threshold S_T. New views of an object are only collected into the object representation if their similarity to the previously stored templates in R_l is less than S_T. The parameter S_T is critical, characterizing a compromise between the object representation accuracy and the computation time. We denote the similarity between view $\mathbf{x}^i$ and representative $\mathbf{r}^k$ by A_{ik} and compute it based on the quadratic Euclidean distance in feature space by $A_{ik} = \exp\left(-(\mathbf{x}^i - \mathbf{r}^k)^2/\sigma\right)$. Here, σ is chosen for convenience such that the average similarity in a generic recognition setup is approximately 0.5.

Online Training. For one learning step the similarity A_{ik} between the current training vector $\mathbf{x}^i$, labeled as object l and all representatives $\mathbf{r}^k \in R_l$ of the same object l must be calculated and the maximum value is computed as $A_i^{\max} = \max_{k \in R_l} A_{ik}$. The training vector $\mathbf{x}^i$ and the corresponding class label will be added to the object representation if $A_i^{max} < S_T$. Assuming that M representatives were present before, we then choose $\mathbf{r}^{M+1} = \mathbf{x}^i$. Otherwise we assume that the vector $\mathbf{x}^i$ is already sufficiently represented by one $\mathbf{r}^k$, and do not add it to the representation.

Online Recognition. Recognition of a test view $\mathbf{I}_j$ is done with a nearest neighbour search of the hierarchy output $\mathbf{x}^j(\mathbf{I}_j)$ to the set of representatives. In contrast to the training, the similarity A_{jk} must be calculated between the current C2 feature vector $\mathbf{x}^j$ and all representatives $\mathbf{r}^k$. The class label of the winning representative $\mathbf{r}^{k^j_{\max}}$ with $k^j_{\max} = \arg\max_k(A_{jk})$ is then assigned to the current validation vector $\mathbf{x}^j$. Due to the non-destructive incremental learning process, online learning and recognition can be done at the same time, without a separation into training and testing phases.

Rejection. For a real application of the online learning system e.g. in a robot interaction scenario it is crucial to reach good classification results, but also unknown objects and clutter should be rejected. This can be done based on the similarity of a test view to the winning representative. Due to the different structural complexity of the appearance variation of different objects, the rejection can be largely improved by choosing the detection threshold similarity dependent on an estimate of the object complexity. This can be estimated by the average number of non-zero elements of the C2 feature vectors.

3 Experimental Results

For our experiments we use a setup, where we show objects, held in hand with a black glove, in front of a black background. Images are taken with a camera, segmented using local entropy-thresholding, normalized in size (each view is 64x64 pixel large) and converted to grey scale. We show each object by rotating it freely by hand for a few ten seconds, which results in 500 input images $\mathbf{I}_i$ per object. Another set of 500 images for each object is recorded for validation. Some rotation examples are shown in Fig.2a. The difficulty of this training ensemble is the high variation of objects during rotation around three axes and the sometimes only partially segmented object views (e.g. mug and cup).

Figure 2b shows how the similarity threshold S_T influences the number of selected representatives. It can be seen that this number strongly depends on the complexity of the object, i.e. shape-dependent appearance variation.

The first investigation of training time should demonstrate how long it takes to incrementally train one object using a real camera. The training speed is limited by the frame rate of the used camera (12,5 Hz) and the computation time needed for the entropy segmentation, the extraction of the corresponding sparse C2 feature vector $\mathbf{x}^i$ with 3200 dimensions and the calculation of similarities A_{ik} (see Sect.2). For the shown curves of the teapot and the cup we trained all other seven objects and incrementally trained the teapot or cup as the eighth object. Figure 3a shows how long it takes until the newly added object can be robustly separated from all other objects.

We also investigated how fast our model performs on a saved image ensemble, without the limitation of the camera's frame rate and the segmentation. For this exploration all eight objects are trained in parallel. We started with a training ensemble of 25 training views for each object and determined the needed training time and the classification rate. Afterwards we increased the number of training views for each object in 25 view steps until all 500 training views for each object

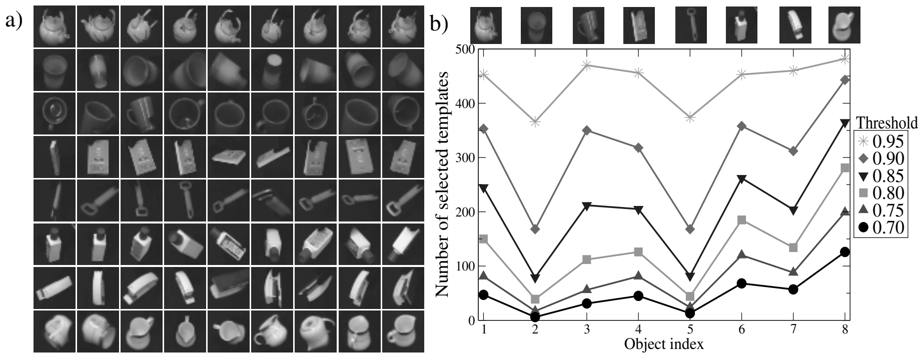


Fig. 2. Test images and the number of selected representatives for different similarity thresholds. (a) Some example images of the eight freely rotated objects, taken in front of a dark background and using a black glove for holding, causing also some minor occlusion effects. The difficulty of this database is the rotation of objects around three axes. Additionally some object views are only partially segmented. (b) Number of selected representative vectors for changing similarity thresholds. The selected number strongly depends on the shape-dependent appearance variation of the objects.

are reached. Figure 3a (labeled with "database") shows that the training phase takes less than 1 minute for a training ensemble of $8 \cdot 500 = 4000$ object views.

We compared the classification results (see Fig.3b) of our model with a normal nearest neighbour classifier (NNC), where *every* training vector $\mathbf{x}^i$ is directly used as a representative and a one-layered sigmoidal network trained by a gradient-based supervised learning on the C2 feature vectors $\mathbf{x}^i$. The sigmoidal network consists of an input and output layer, without hidden layers. For every object we used one output node, whereas each node has a linear scalar product activation and a sigmoidal transfer function. We trained these networks with all training vectors or with the selected representatives of our model. Figure 3b shows that the exhaustive NNC and our model using C2 activations perform quite equal, which means that our model can reduce the number of relevant representatives (2906 representatives are selected from 4000 training views (72,65%), for 64x64 pixel images and $S_T = 0.92$) without losing classification performance. The sigmoidal networks perform in comparison to our model slightly worse, but the representational resources are strongly reduced. We used only one linear discriminating weight vector per object, i.e. 8 weight vectors. We also trained these networks with the selected representatives of our model, which speeds up the training process but also slightly reduces the classification rates. Further we

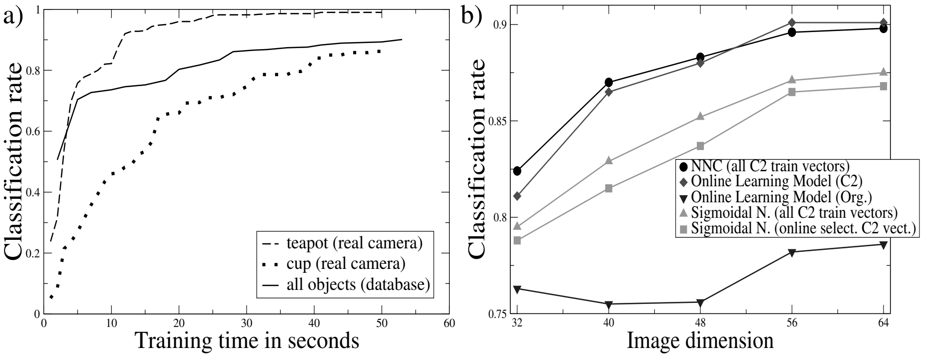


Fig. 3. Classification rate over training time and a comparison between nearest neighbour classifier (NNC) (all training vectors are used), our online learning model and sigmoidal networks for different image dimensions. (a) Classification rate dependent on needed computation time for the whole training set (labeled with "database") and the training of one object with a real camera. The training speed using a camera is also limited by the frame rate of the camera and the segmentation. Good recognition performance can be achieved within 20-30 seconds of online training. (b) Comparison of classification rates between a simple NNC (all training vectors are used), our model (C2 activations, original images) and sigmoidal networks for different image sizes. It can be seen that the classification rates of the NNC and our online learning model using the C2 activation are more or less equal and that the one-layered sigmoidal networks perform slightly worse. The classification results of our model using the C2 features are distinctly better than the results using the original grey value images.

show that the usage of C2 features considerably increases the classification performance of our model compared to the original grey value images (see Fig.3b).

The rejection capability of our model was tested with all 8 trained objects and 16000 different grey value clutter images. These images were randomly cut out of large scenes and contain parts of different objects, portions of buildings, faces and so on. The reached false negative rate for the 8 trained objects was 7.9%, whereas 8.3% of all clutter images are classified as an object.

4 Conclusion

We have shown that the hierarchical feature representation is well suited for online learning using an incremental vector quantization model approach. Of particular relevance is the technical realization of the appearance-based online learning of complex shapes for the context of man-machine interaction and humanoid robotics. This capability introduces many new possibilities for interaction scenarios and can incrementally increase the visual knowledge of a robot.

The simple template-based representation of objects in our approach allows a simple incremental buildup of the representation during online-learning. Nevertheless, due to the exhaustive storage of high-dimensional feature map information a similar approach seems prohibitive given the, arguably large, but finite neural resources in the brain. We have already investigated that a later offline refinement of the representation like supervised gradient-based training can be used to reduce the representational effort considerably. Such a process could be used to guide the transfer from a simple photographic short-term memory presented here to a more optimized long-term memory representation. The investigation of appropriate models that are related to the representation of visual representation in the inferotemporal cortex [4] will be the subject of future study.

Acknowledgments. We thank C. Goerick, M. Dunn, J. Eggert and A. Ceravola for providing the image acquisition and processing system infrastructure.

References

1. Arsenio, A.: Developmental learning on a humanoid robot. Proc. IJCNN 2004, Budapest.
2. Bekel, H., Bax I., Heidemann G., Ritter H.: Adaptive Computer Vision: Online Learning for Object Recognition. Proc. DAGM 2004 Springer C. E. Rasmussen et al. (2004) 447–454
3. Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics **36 (4)** (1980) 193–202
4. Tanaka, K.: Inferotemporal cortex and object vision: stimulus selectivity and columnar organization. Annual Review of Neuroscience, **vol. 19** (1996) 109–139
5. Steels, L., Kaplan, F.: AIBO's first words. the social learning of language and meaning. Evolution of Communication, **vol. 4, no. 1** (2001) 3–32
6. Wersing, H., Körner, E.: Learning Optimized Features for Hierarchical Models of Invariant Object Recognition. Neural Computation **15 (7)** (2003) 1559–1588

Extended Hopfield Network for Sequence Learning: Application to Gesture Recognition

André Maurer, Micha Hersch, and Aude G. Billard

Ecole Polytechnique Fédérale de Lausanne (EPFL),
Swiss Federal Institute of Technology Lausanne,
Autonomous Systems Laboratory,
CH-1015 Lausanne, Switzerland
aude.billard@epfl.ch

Abstract. In this paper, we extend the Hopfield Associative Memory for storing multiple sequences of varying duration. We apply the model for learning, recognizing and encoding a set of human gestures. We measure systematically the performance of the model against noise.

1 Introduction

The work we present here is part of a research agenda, that aims at modeling the neural correlates of human ability to learn new motions through the observation and replication of other's motions [1,2]. In this paper, we investigate the use of a biologically plausible mechanism for recognizing, classifying and reproducing gestures.

Associative memories based on Hebbian learning, such as the *Hopfield network*, are interesting candidates to model the propensy of biological systems to encode and learn complex sequences of motion [3]. The Hopfield network is known predominantly for its ability to code static patterns. However, recent work extended the Hopfield model to encode a time series of patterns [4]. In the present work, we extend this model to encode *several* sequences of patterns in the same model. While the capacity of such RNN models have been studied at length in simulation [5,6], there has been yet little work demonstrating their application to the storage of real data sequences. Here, we validate the model for encoding human gestures and measure the performance of the model in the face of a large amount of noise.

Fundamental features of human ability to imitate new motions are a) the ability to robustly recognize gestures from partially occluded demonstrations (this is tightly linked to our ability to predict the dynamics of the motion from observing only the onset of the motion); and b) to store and reproduce a generalized version of the motion, that encapsulates only the key features of the motion. We show that the model can successfully reproduce these two key features.

2 Experimental Set-Up and Model

Figure 1 shows a schematic of the data flow across the complete architecture. Input to the system consists of the kinematic data of human motion. The data is

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 493–498, 2005.

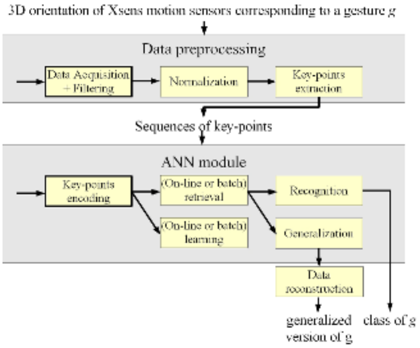


Fig. 1. Schematic of the data flow across the complete architecture

first preprocessed to smooth and normalize the trajectories, as well as to reduce the dimensionality of the dataset to a subset of keypoints. The time series of key-points is encoded and classified in a set of Artificial Neural Networks (ANNs). Learning results in the storage of a generalized form of the demonstrated gestures. The system outputs either the class of the gesture g or the generalized form of the gesture corresponding to the class g.

Data acquisition and preprocessing: Data consist of 45 gestures, composed of the 4 angular trajectories of the arm (shoulder abduction-adduction, flexion-extension and humeral rotation, and elbow flexion-extension) of 8 demonstrators during 5 repetitions of drawing the stylized letters A, B, C, D, E, (see figure 2).

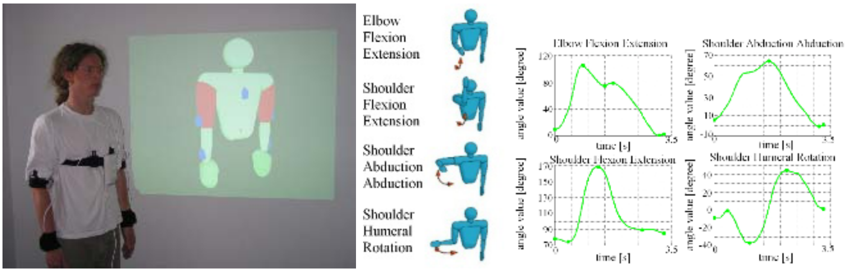


Fig. 2. The demonstrator's motions are recorded by a set of Xsens motion sensors, attached to the torso, upper and lower arms (left). The information is then used to reconstruct the trajectories of 4 joint angles of the arm (middle). (Right:) Raw data trajectories. Each subplot correspond to the trajectory of one of the 4 joint angle. Circles represent the extracted key-points.

Each trajectory is smoothed using a 1D local Gaussian filter of size 7. From those trajectories, we extract a set of P key-points $\{\theta_p^a, t_p^a\}$ (a=1..4, p=1..P). A key-point is either the first or last element of the trajectory or an inflexion point (zero velocity). Such a segmentation aims at extracting the correlations between the different joint trajectories. The duration of the whole trajectory is normalized so that two gestures belonging to the same class but performed at different speed are encoded into similar sequences.

Pattern encoding: Each element of the input sequence $\{t_p^a, \theta_p^a\}$ $(a = 1..4$, $p = 1..P$; P being the length of the sequence) is encoded in a 2D matrix $\tilde{x}$ of real values, as follows:

$$(t_{\tilde{i}}^{\tilde{a}}, \theta_{\tilde{i}}^{\tilde{a}}) \rightarrow (\tilde{x}_{u,v})$$

where u=1..M and $v = 1..N$. In order to preserve the notion of neigbourhood across inputs we encode a pair $(t_{\tilde{p}}^{\tilde{a}}, \theta_{\tilde{p}}^{\tilde{a}})$ using a 2D gaussian distribution function f, centered on $\boldsymbol{\mu} = (\mu_t, \mu_\theta)^T$ with standard deviation $\boldsymbol{\sigma} = (\sigma_t, \sigma_\theta)^T$:

$$f(x_u, x_v) = e^{\frac{-\frac{1}{2}(x_u - \mu_t)^2}{\sigma_t^2}} e^{\frac{-\frac{1}{2}(x_v - \mu_\theta)^2}{\sigma_\theta^2}} .$$

2.1 ANN Module

The general topology of the network is presented in Figure 3. Inputs to the network are sequences of key-points. The sequences are stored in a series of Hopfield networks linked to one another through the matrix of weights W. Each sequence is then classified according to a set of classes $c = 1, .., C$ and C, represented by a set of neurons y_c.

The activity of each neuron, for each angle a, and for each time step t=1..P, $x_{u,v}^a(t)$, as well as the weights $w_{u,v,u',v'}^a(t)$ storing the correlation between the neuronal activities $x_{u,v}^a(t)$ and $x_{u,v}^a(t+1)$ are normalized and bounded in [0..1].

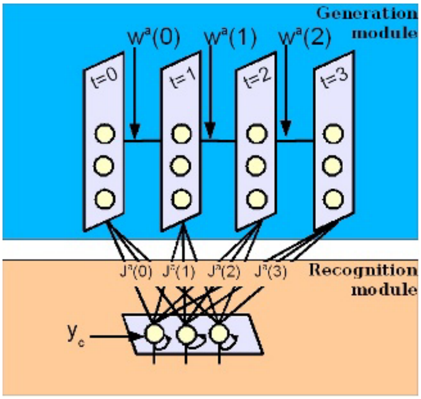


Fig. 3. Network topology. The weight matrix W (generation module) connects all neurons from one layer to all neurons of the next layer. Each output neuron y_c corresponds to a class c of gestures. The weights of the matrix J (recognition module) are set so that the output neuron $y_{\tilde{c}}$ is maximal when a sequence of class $\tilde{c}$ is generated. Each angular trajectory a (a=1..4) is encoded in a separate networks.

Learning process: The learning rule for updating elements of W is a modification of the one presented in [4], to allow storage of many sequences rather than just one, as well as to allow a non-overlapping encoding with $x^a_{u,v} = [0..1]$, as opposed to $x^a_{u,v} = \pm 1$.

$$w^a_{u,v,u',v'}(t) = \sum_s x^{s,a}_{u,v}(t) x^{s,a}_{u',v'}(t+1), \quad t = 1..P-1 \tag{1}$$

where s indicates the training sequence.

When learning a gesture s belonging to a class $\tilde{c}$, we set the output neurons $y_c = 0 \ \ \forall c \neq \tilde{c}$ and $y_{\tilde{c}} = 1$. Updating the elements of the recognition matrix J is done according to:

$$\Delta J^a_{u,v,c}(t) = x^{s,a}_{u,v}(t) y^{s,a}_c \tag{2}$$

Retrieval process: In order to retrieve the generalized form of the sequence associated with a given class, we activate one of the y_c neurons and then reactivate the neurons in each layer of the extended Hopfield in sequence for P-1 time steps. That is, we update each neuron $x^a_{u,v}(t+1)$ according to:

$$x^a_{u,v}(t+1) = \sum_{u'} \sum_{v'} w^a_{u,v,u',v'}(t) \cdot x^a_{u',v'}(t), \quad t = 1..P-1 \tag{3}$$

Figure 4 shows the history of the network state after the retrieval of a sequence of four elements.

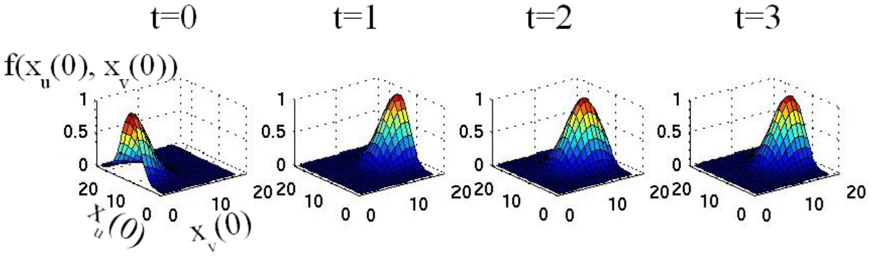


Fig. 4. State of a 4-layer extended Hopfield network while retrieving a sequence of four elements

During recognition of a gesture, we proceed conversely by activating a subset of the first layers of the extended Hopfield network. Recognition of the class to which the gesture belongs is done by reactivating the output neurons y_c according to:

$$y_c(t+1) = y^a_c(t) + \sum_a \sum_u \sum_v J^a_{u,v,c}(t). \tag{4}$$

3 Results

We evaluated the performance of the network to classify and regenerate our set of 45 gestures (stylized drawings of the letters A to E). Further, in order to evaluate the network's capacity against a large amount of noise, we generated a *synthetic dataset* of 250 gestures, by adding gaussian noise on one of the gestures belonging to the *real dataset*. Each dataset was divided equally into a *training set* and a *testing set*. Synthetic data were generated by displacing each key-point according to a gaussian distribution function, centered on the original key-point and with a given standard deviation $\boldsymbol{\sigma}^D = (\sigma_t^D, \sigma_\theta^D)^T$, see Figure 5. For each value of σ^D, we generated 10 different gestures. We measured a standard deviation (noise) on the *real dataset* of $\boldsymbol{\sigma}^D = (0.88, 22.23)^T$.

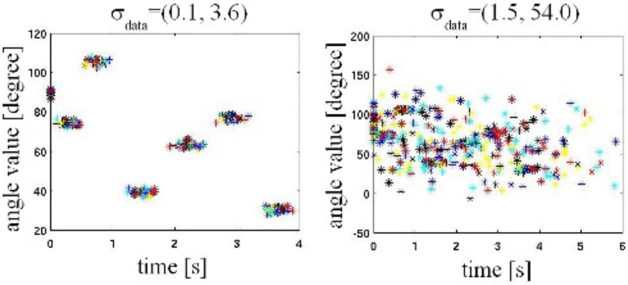


Fig. 5. Sequence of key-points (t, θ) when the noise is generated with $\sigma_t^D = 0.1$ and $\sigma_\theta^D = 3.6$. Clusters do not overlap. Middle: key-points (t, θ) with $\sigma_t^D = 1.5$ and $\sigma_\theta^D = 54.0$. The overlap between clusters is large.

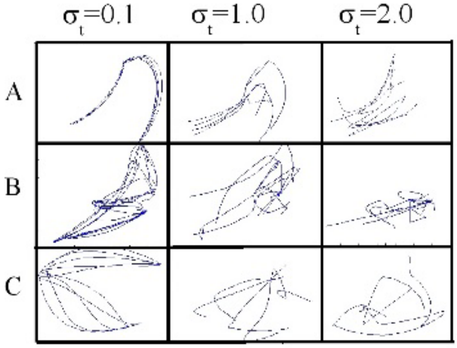


Fig. 6. Distortion of the original gestures with a noise level of $\sigma_t^D = 0.1, 1.0$ and 2 respectively. Sole the gestures on the left are easily recognizable by the human eye.

During the learning phase, the system is trained on a set of gestures. During the testing phase, the system is evaluated on its ability to both recognize and regenerate the data.

Recognition Performance: Figure 7, right, shows the recognition rate on the synthetic testing set as an effect of the temporal noise (average over 10 different gestures for each value of σ) with $\sigma_t^D = 3.6\sigma_D^\theta$. The recognition rate τ is given by *the proportion of correctly recognized patterns relative to the total number of patterns.*

We observe that the recognition is perfect for all gestures when the noise is inferior to ($\sigma_t^D \leq 0.25$). However, for high noise ($\sigma_t^D >= 1.0$), the recognition rate decreases importantly.

Data Regeneration: Figure 7, left, shows 3 examples of regenerated gestures, superimposed to a set of 4 training gestures generated with a noise value $\boldsymbol{\sigma}^D = (0.1, 3.6)$. The network generates a generalized form of the gestures that encapsulate the major qualitative features (point of curvature) of the demonstrations.

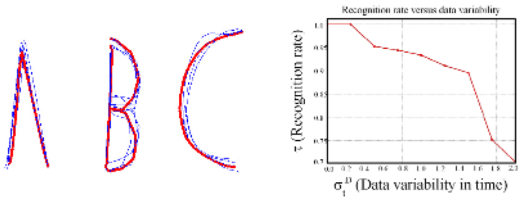


Fig. 7. (Left:) Regenerated gestures (Bold line) against a set of 4 examples of demonstrated gestures (thin line) with a noise of $\boldsymbol{\sigma}^D = (0.1, 3.6)^T$. (Right:) Recognition rate as an effect of the noise.

References

1. Arbib, M., Billard, A., Iacoboni, M., Oztop, E.: Mirror neurons, imitation and (synthetic) brain imaging. In: Neural Networks. Volume 13 (8/9). (2000) 975–997
2. Billard, A.: Imitation. In: Handbook of Brain Theory and Neural Networks. Volume 2. MIT Press (2002) 566–569
3. Wang, D.: Temporal pattern processing. The Handbook of Brain Theory and Neural Network **2** (2003) 1163–1167
4. Miyoshi, S., Yanai, H., Okada, M.: Associative memory by recurrent neural networks with delay elements. Neural Networks **17** (2004) 55–63
5. Miyoshi, S., Nakayama, K.: A recurrent neural network with serial delay elements for memorizing limit cycles. In: Proc. of ICANN'95. (1995) 1955–1960
6. Mueller, K.R., Ibens, O.: Sequence storage of asymmetric hopfield networks with delay. In: ICANN'91. (1991) 163–168

Accurate and Robust Image Superresolution by Neural Processing of Local Image Representations

Carlos Miravet[1,2] and Francisco B. Rodríguez[1]

[1] Grupo de Neurocomputación Biológica (GNB), Escuela Politécnica Superior,
Universidad Autónoma de Madrid, 28049 Madrid, Spain
[2] SENER Ingeniería y Sistemas, S.A., Severo Ochoa 4 (P.T.M.), 28760 Madrid, Spain
{carlos.miravet, f.rodriguez}@uam.es

Abstract. Image superresolution involves the processing of an image sequence to generate a still image with higher resolution. Classical approaches, such as bayesian MAP methods, require iterative minimization procedures, with high computational costs. Recently, the authors proposed a method to tackle this problem, based on the use of a hybrid MLP-PNN architecture. In this paper, we present a novel superresolution method, based on an evolution of this concept, to incorporate the use of local image models. A neural processing stage receives as input the value of model coefficients on local windows. The data dimensionality is firstly reduced by application of PCA. An MLP, trained on synthetic sequences with various amounts of noise, estimates the high-resolution image data. The effect of varying the dimension of the network input space is examined, showing a complex, structured behavior. Quantitative results are presented showing the accuracy and robustness of the proposed method.

1 Introduction

Image superresolution [1, 2] involves the processing of an image sequence to generate a high-resolution description of the underlying scene. From the earliest algorithm proposed by Tsai and Huang [3], a number of approaches have been proposed. Of these, bayesian MAP (Maximum A Posteriori) methods [4,5] have gained particular acceptance due to their robustness and their capability to incorporate *a priori* constraints. The main drawback of these methods comes from their associated high computational loads, as they use iterative techniques in spaces of high dimensionality.

Recently [6, 7], the authors have proposed a neural network based technique that provides results comparable to classical methods with a substantial decrease in computational complexity. This technique estimates image values in a dense grid using an irregular interpolation scheme, with distance dependent interpolation weights. Optimal distance-to-weight mappings are learned from synthetic sequences and corresponding high-resolution images, using a hybrid MLP-PNN (Multi Layer Perceptron – Probabilistic Neural Network) architecture. In a second step, high-resolution image values are restored from estimated grid values using optimal filters.

The use of interpolation schemes based on distance dependent weights, no matter how optimally these weights can be tuned, poses a fundamental limit on the attainable

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 499 – 505, 2005.

performance as, being independent of local image structure, the method is forced to operate in the same way near an image edge or inside a uniform image patch. In this paper, an evolution of our previous distance-based algorithm is presented, which makes explicit use of local image representations. As in our previous approach, the proposed system learns from examples how to perform superresolution. In this way, the computational load is mostly displaced to the off-line learning process, enabling a fast, non-iterative response in the network deployment phase.

The proposed method is based on the sequential application of two processing steps. In the first step, sequence pixels are projected onto the high-resolution frame, and local image representations are built for each site of an embedded high-resolution grid. In the second step, the image representation coefficients in the neighborhood of each grid site are processed by a neural network to estimate the high-resolution image values. The dimensionality of the network input data is previously reduced by application of a PCA (Principal Component Analysis) technique.

Out method has shown to provide excellent results over a wide range of input noise levels. In the following sections, we detail the processing steps involved, and present experimental results that include a quantitative comparison of several methods. In the last section, a brief discussion of the main results obtained is presented.

2 Local Image Representation

The first step of our superresolution method computes a local image representation for each site of the high-resolution (HR) grid to be estimated. These local representations are built using the sequence pixels values projected onto the HR grid. The projection operation requires the previous knowledge of the geometrical transformations that relate input sequence frames. To estimate this data, an adaptation of a classical sub-pixel registration procedure [8] has been used.

Table 1. RMS errors for different interpolation schemes

Method	**σ=0**	**σ=5**	**σ=10**	**σ =20**
NN_SEQ	9.28	10.40	13.48	21.97
Distance-based interpolation				
Inverse distance weight	6.82	7.48	9.23	14.16
MLP-PNN	5.61	6.00	6.85	8.43
Polynomial Models				
Order 1	7.24	7.37	7.51	8.48
Order 2	2.90	3.56	5.04	8.75
Order 3	2.34	3.29	5.25	9.72

Polynomial models have been used to describe the local image structure at sub-pixel level. Polynomial coefficients are determined by a general linear least squares technique, with matrix inversion performed using singular value decomposition, SVD [9]. The use of SVD improves robustness when the problem is close to singular, due, for instance, to an inadequate model order selection in an image patch, or induced by a large image noise level.

In table 1 are presented the root mean squared (RMS) errors for different irregular interpolation methods. Polynomial models with orders ranging from 1 to 3 have been considered. For comparison, the error corresponding to the direct selection of the nearest pixel (SEQ_NN) has been included, together with two distance-based interpolators: the first processing step of our previous method (MLP-PNN), and a method recently proposed in the literature [10], which makes use of inverse distance weights. As it can be observed, interpolation with second order local polynomials provides distinct advantages over distance-based methods at low and medium noise levels. Increasing polynomial order over this point result only in marginal gains in performance. In view of these results, a second order polynomial has been considered in our subsequent work.

2.1 Dimensionality Reduction by PCA. Eigenimages of Coefficients

The dimension of network input space is given by the model dimension (6 for a second order polynomial) times the size of the considered local neighborhood. Principal component analysis (PCA) has been applied to reduce the dimensionality of this space. PCA [11] is a linear technique that yields minimal representational error (in terms of mean squared error, MSE) for a given reduction in the dimensionality space. PCA operates by projecting the input data onto an orthogonal basis of the desired dimension, where the basis vectors are the eigenvectors of the input data covariance matrix, ranked in order of decreasing eigenvalues.

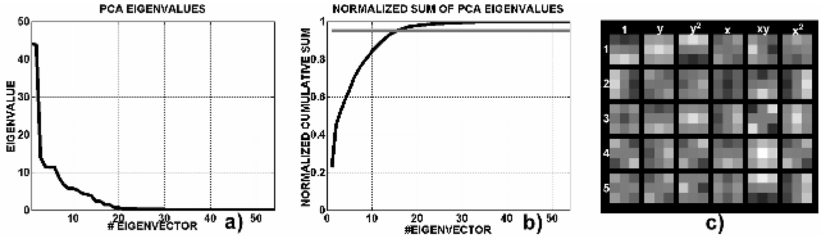


Fig. 1. Results of the PCA process, for a 3x3 local neighborhood: a) eigenvalues sorted in decreasing order; b) percentage of input variance maintained after dimensionality reduction, as a function of the number of used eigenvectors; c) first five eigenimages.

In figure 1 a, b) are presented, for a 3x3 local neighborhood, the obtained eigenvalues sorted in decreasing order, and the percentage of the input variance represented after dimensionality reduction, as a function of the number of used eigenvectors. Using a common heuristic approach, such as maintaining enough components to explain 95% of the input variance, will lead to the selection of the first 16 eigenvectors to build the projection basis. In the next section we will see that heuristics of this type are not appropriate for this problem, as use of low variance eigenvectors can have a considerable impact on the final prediction error.

In figure 1 c) are presented the five eigenvectors of highest eigenvalue, with components rearranged in the form of eigenimages. The preliminary experiments conducted show their stability under variations of input noise and image content, defining basic patterns of spatial variation of local polynomial models in images examined at sub-pixel level. The projection of the local image models on this set of spatial patterns is used as input data to the network, as described in the next paragraph.

2.2 Neural Network Training Results

To estimate the HR pixels, it has been used a multi-layer perceptron (MLP) architecture [11], with two layers of neurons. The hidden layer is composed of 10 neurons with hyperbolic tangent activation functions. These neurons are connected to a single neuron in the output layer, with a linear activation function. The output of this neuron provides a zero-mean, unit-variance estimation of the high-resolution central pixel. Renormalization provides the final value. A diagram of the system is presented in figure 2 a).

The network has been trained using a conjugate gradient descent method on synthetic data sets with various noise levels. The training data has been generated synthetically from a set of 23 high-resolution images of urban scenes acquired by the *Quickbird* satellite imaging system.

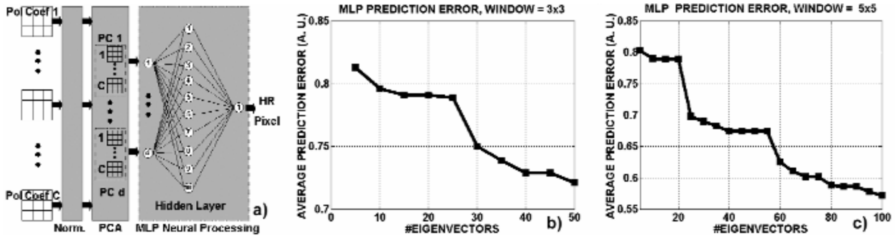


Fig. 2. a) Diagram of the superresolution neural processing stage; b) network average prediction error, measured on a validation test set, as a function of the dimensionality of the network input space for a 3x3 neighborhood; c) average prediction error for a 5x5 neighborhood.

The neural system has been trained for several input sizes, to study quantitatively the effect on the predicting error of augmenting the input data with the value corresponding to each sorted principal component. In figure 2 is presented the evolution of this error as a function of the number of eigenvectors retained in the dimensionality reduction step, for local windows of sizes 3x3 and 5x5. As it is apparent, the error decreases non-gradually, with almost flat zones interleaved with step decreases in the prediction error. Furthermore, these step changes occurred also in zones where the representational error of the dimensionality reduction step might be seen in principle as negligible (see figure 1, for a 3x3 window), highlighting the inadequacy of an MSE-based criteria in this case to select the dimension of the network input space.

3 Experimental Results

The accuracy and stability of the proposed method has been tested with both synthetic and outdoor sequences of different image content. Here, we present the results of applying superresolution on synthetic image sequences with various noise levels, to enable the quantitative comparison, in terms of RMS, of the prediction error of different methods. The sequences have been generated from a high-resolution image of an urban area acquired with the IKONOS satellite imaging system. This image has not been used during the training process. From this image, two sequences of 25 images (corresponding to 1s in the CCIR standard) were generated, one corresponding to the noiseless case, and the second one corrupted highly by addition of Gaussian noise (σ = 20 gray levels).

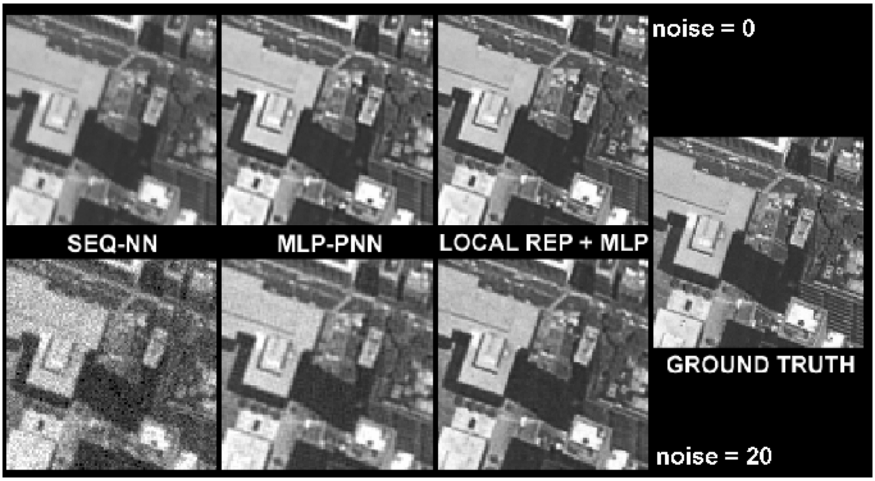


Fig. 3. Superresolution results, for a scaling factor of 2, on a synthetic sequence of an urban area using different methods. In the first row, are presented the results for a noiseless input sequence. The second row contains the results for a sequence corrupted with Gaussian noise of σ = 20 gray levels. Each column contains the results of a different method. The ground truth is presented at the right of the image.

On figure 3 are presented the results obtained with three superresolution methods. In the first column (SEQ-NN) are shown the results obtained estimating the HR pixels as the value of the nearest projected sequence pixel. This method produces jagged edges in the noiseless case, and no input data noise reduction. Both effects are reduced, as can be seen in the second column, when using the MLP-PNN technique, with networks specialized in the appropriate noise range. Finally, the use of the proposed method, with 3x3 neighborhoods and 40 eigenvectors, improves both image definition and noise rejection at all input noise levels, as can be seen in the third column of the figure. These perceptual results are supported by the RMS error figures reported in table 2.

Table 2. High-resolution RMS reconstruction errors for several methods

	ZOOM	SEQ-NN	MLP-PNN	LOCALREP+MLP
noiseless sequence (σ=0)	19.39	13.04	11.24	8.28
noisy sequence (σ=20)	24.08	23.58	14.60	11.06

4 Discussion

A superresolution method based on neural processing of local image representations has been developed. The method process representation coefficients on local windows to estimate the HR image values. The dimensionality of input data to this network is firstly reduced by application of a PCA technique. The eigenimages obtained have been shown to be stable for a wide range of noise levels and image contents, representing intrinsic properties of image structure at sub-pixel levels. Variations of prediction error with input size have been examined, showing a non-gradual, structured behavior, reflecting the inadequacy of MSE heuristics in this problem. The relation of the results with those provided by non-linear methods, such as ICA [12], will be investigated in future work. The experimental results obtained show the accuracy and robustness of the developed method.

Acknowledgments

This work has been partially supported by grants BFI2003-07276 and TIN2004-04363-C03-03, and by PROFIT project FIT-330100-2004-91.

References

1. S. Borman, R. Stevenson, “Super-resolution from image sequences-a review”, *Midwest Symposium on Circuits and Systems* (1998).
2. S. C. Park, M. K. Park, M. G. Kang, “Super-resolution image reconstruction: a technical overview”, *IEEE Signal Processing Magazine*, 21-35 (2003).
3. R.Y.Tsai and T.S.Huang (Ed.), “Multiframe image restoration and registration”, in *Advances in Computer Vision and Image Processing* volume 1, pages 317-339, JAI Press Inc.(1984).
4. R. C. Hardie, K. J. Barnard, J. G. Bognar, E. E. Armstrong, and E. A. Watson, "High-resolution image reconstruction from a sequence of rotated and translated frames and its application to an infrared imaging system", *Optical Engineering*, **37**(1), 247-260 (1998).
5. R. R. Schultz and R.L. Stevenson. “Extraction of high-resolution frames from video sequences”, *IEEE Trans. Image Processing*, vol. 5, nº 6, pp. 996-1011 (1996).
6. C. Miravet and F. B. Rodríguez, “A hybrid MLP-PNN architecture for fast image super-resolution”, Joint 13th International Conference on Artificial Neural Networks / 10th International Conference on Neural Information Processing (ICANN/ICONIP), *Lecture notes in Computer Science*, vol. 2714, Springer-Verlag, pp. 417-424 (2003).
7. C. Miravet and F. B. Rodríguez, “A two-step neural network based algorithm for fast high-resolution image reconstruction”, *Image and Vision Computing* (submitted).

8. M.Irani, S.Peleg, “Improving resolution by image registration”, CVGIP: *Graphical Models and Image Processing*. 53, 231-239 (1991).
9. W. Press, S. Teukolsky, W. Vetterling, B. Flannery, *Numerical recipes in C*, Cambridge University Press, 2º Ed. (1992)
10. M. S. Alam, J. G. Bognar, R. C. Hardie and B. J. Yasuada, “Infrared image registration and high-resolution reconstruction using multiple translationally shifted aliased video frames”, *IEEE Transactions on instrumentation and measurement*, vol. 49, nº 5 (2000).
11. A.Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press (1995).
12. A. Hyvärinen, J. Karhunen, E. Oja, *Independent Component Analysis*. Wiley-InterScience (2001).

The Emergence of Visual Object Recognition

Alessio Plebe and Rosaria Grazia Domenella

Department of Cognitive Science, University of Messina, Italy
{aplebe, rdomenella}@unime.it

Abstract. The model here proposed simulates the development of the object recognition capability, assuming that recognition does not imply any sort of explicit geometrical reconstruction and emerges as result of interactions between epigenetic influences and basic neural plasticity mechanisms. The model is a hierarchy of artificial neural maps, mainly based on the LISSOM architecture, achieving self-organization through simulated intercortical lateral connections.

1 Introduction

This work proposes a model of the development of the object recognition capability, one of the most valuable outcome of the human visual system, yet the less understood. This work assumes that recognition does not imply any sort of geometrical reconstruction, as posed in early works [14], it is fully driven by the two dimensional view captured by the retina, as supported by several neurocognitive [6], neurophysiological [13], and theoretical [5] studies.

A special focus of this model is the investigation on how the recognition capability can emerge, assuming that the processing functions involved in recognition are not genetically determined and hardwired in the neural circuits, but are the result of interactions between epigenetic influences and some very basic neural plasticity mechanisms. This view agrees with a more general explanation of the representational nature of the neural system [16], but is especially supported in the specific case of vision [10].

2 Modeling Self-organization in Cortical Maps

The first mathematical model of how the visual cortex can spontaneously develop its mature organization was proposed in [19]. The mechanism, generally referred as *self-organization*, is based on the self-reinforcing local interaction, corresponding to Hebbian plasticity, constrained by competition, that takes into account the limitation of biological resources. The two mechanisms are modeled by von der Malsburg in systems of differential equations, simulating visual organizations like retinotopy, ocular dominance and orientation sensitivity.

Later the self-organization mechanism has been largely popularized by Kohonen, who invented a much simpler yet efficient model called SOM (*Self-Organizing Maps*) [11]. The learning rule is on a *winner-take-all* basis: if the

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 507–512, 2005.

input data are vectors $\boldsymbol{v} \in \mathbb{R}^N$, the SOM will be made of some M neurons, each associated with a vector $\boldsymbol{x} \in \mathbb{R}^N$ and a two dimensional (in vision applications) coordinate $\boldsymbol{r} \in \{< [0,1],[0,1] >\} \subset \mathbb{R}^2$. For an input v there will be a winner neuron w satisfying:

$$w = \arg \min_{i \in \{1,\ldots,M\}} \{\|\boldsymbol{v} - \boldsymbol{x}_i\|\}\,. \tag{1}$$

Once identified the winner, during training the neural vectors are updated using:

$$\varDelta\boldsymbol{x}_i = \eta e^{-\frac{\|\boldsymbol{r}_w - \boldsymbol{r}_i\|^2}{2\sigma^2}} (\boldsymbol{v} - \boldsymbol{x}_i)\,, \tag{2}$$

where η is the learning rate, σ the amplitude of the neighborhood affected by the updating.

Even if the SOM model has been used for simulating certain properties of the visual cortex, it appears to bee far too oversimplified for a more realistic reproduction of human vision, for example lacking explicit modeling of intracortical connections. A recent model called LISSOM (*Laterally Interconnected Synergetically Self-Organizing Map*) attempts to include lateral connections still preserving the simplicity of the SOM [17, 1]. In this model each neuron is not just connected with the afferent input vector, but receives excitatory and inhibitory inputs from several neighbor neurons on the same map:

$$a_i^{(k)} = f\left(\gamma_{\mathrm{X}}\boldsymbol{x}_i \cdot \boldsymbol{v} \ + \gamma_{\mathrm{E}}\boldsymbol{e}_i \cdot \boldsymbol{y}_i^{(k-1)} \ + \gamma_{\mathrm{H}}\boldsymbol{h}_i \cdot \boldsymbol{z}_i^{(k-1)}\right), \tag{3}$$

where a_i is the activation of a neuron i at time k, vectors $\boldsymbol{y}_i$ and $\boldsymbol{z}_i$ are the activations of all neurons in the map with a lateral connections with neuron i of, respectively, excitatory or inhibitory type. Vectors $\boldsymbol{e}_i$ and $\boldsymbol{h}_i$ are composed by all connections strengths of the excitatory or inhibitory neurons projecting to i. The vectors $\boldsymbol{v}$ and $\boldsymbol{x}_i$ are the input and the neural code. The scalars γ_{X}, γ_{E}, and γ_{H}, are constants modulating the contribution of afferents. The map is characterized by the matrices $\mathbf{X}, \mathbf{E}, \mathbf{H}$, which columns are all vectors $\boldsymbol{x}$, $\boldsymbol{e}$, $\boldsymbol{h}$ for every neuron in the map. The function f is any monotonic non-linear function limited between 0 and 1. The final activation value of the neurons is assessed after a certain settling time K.

All afferent connections to a neuron i adapt by following the rule:

$$\varDelta\boldsymbol{x}_i = \frac{\boldsymbol{x}_i + \eta a_i \boldsymbol{v}}{\|\boldsymbol{x}_i + \eta a_i \boldsymbol{v}\|} - \boldsymbol{x}_i, \tag{4}$$

similarly for weights $\boldsymbol{e}$ and $\boldsymbol{h}$.

3 The Model

The model is made of several artificial cortical layers, the overall scheme of the model is visible on the left of Fig. 1. It has been trained and experimented mainly

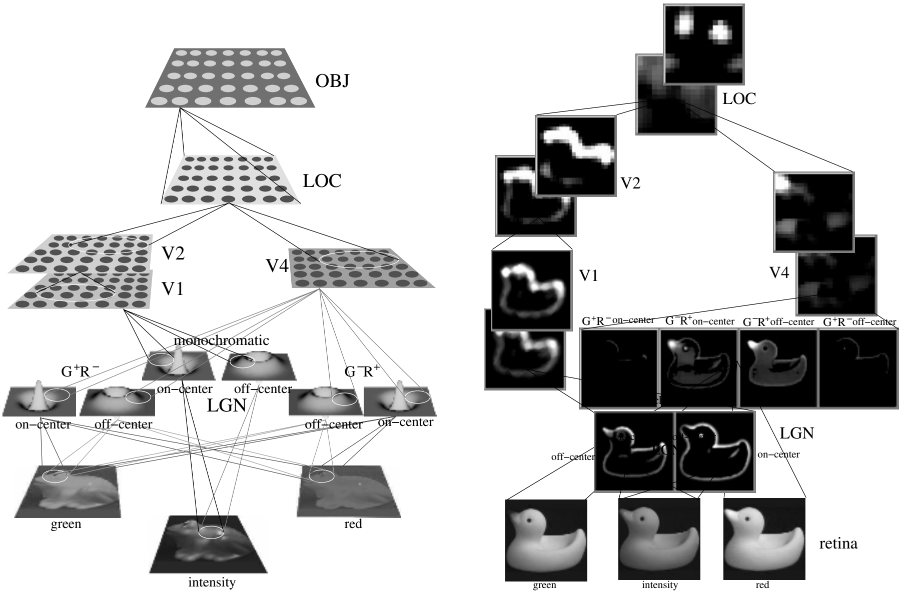


Fig. 1. Scheme of the model architecture (left) and a sample process (right)

with the set of natural images in the COIL-100 benchmark library [15], a collection of 100 ordinary objects, each seen under 72 different perspectives. There are two distinct pathways, one monochromatic connected to the intensity retinal photoreceptors, and another sensitive to the green and red photoreceptors. The lower maps of the model are called LGN with relation to the biological Lateral Geniculate Nucleus, the function performed includes in fact also the contribution of ganglion cells [4]. There are three pairs of on-center and off-center sheets, for intensity and red-green opponents.

The cortical map named V1 collects its afferents from the monochromatic sheets pair in the LGN, and is followed by the map V2, which has a lower resolution and larger receptive fields. Biological V1, is known to be the place of an overlap of many different organizations like retinotopy, ocularity, color and orientation sensitivity [18, 12]; the main phenomena reproduced by this model is the development of orientation domains. The training uses artificial elliptical blobs in the first 10000 steps, followed by natural images for other 10000 steps, according to the cooperative role of spontaneous activity and exposition to natural images in the biological ontogenesis of vision [2]. The size of the excitation connections is reduced to a half from the beginning to the end of the training, this is an other typical reconfiguration of the biological visual cortical circuitry during development.

The color path proceeds to V4, named as the biological area especially involved in color processing [20]. The main feature of the cortical color process is color constancy, that has been also proven to be an emergent capability in infants [3]. This map has the same resolution of V2. During the training of V4, done

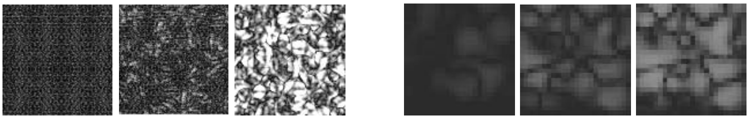


Fig. 2. Development of organizational domains in V1 (left) and V4 (right). The sensitivity to orientation or hue constancy is shown in gray scale.

with synthetic blobs of uniform color, at the beginning there is low sensitivity, peaked in the middle range between red and green, at the end the color sensitivity of all patches is uniformly distributed along the hue range. The development of domains is shown in Fig. 2.

The paths from V4 and V2 rejoin in the cortical map LOC, which has larger receptive fields, and is the last area of LISSOM type. It is known that knowledge of non-visuotopic areas in humans is currently poor [8], and scarcely comparable with primates [7]. An area that recently has been suggested as strongly involved in object recognition is the so-called LOC (Lateral Occipital Complex) [9]. The response properties of cells in this area seems to fulfill the requirement for an object-recognition area: sensitivity to moderately complex and complex visual stimuli, and reasonable invariance properties.

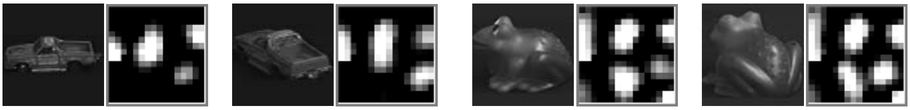


Fig. 3. Invariance properties of LOC map in response to COIL images rotated by 60°

Table 1. Correlations between images affected by viewpoint transformation (middle column), and the corresponding LOC map (right column), averaged over all 100 objects.

type of transformation	input image	LOC map
rotation of 30°	0.781	0.903
rotation of 60°	0.648	0.756
size downscaling of 80%	0.637	0.794
size downscaling of 70%	0.547	0.655
translation of 10%	0.463	0.586
translation of 20%	0.207	0.397

The model LOC is trained using all COIL-100 objects, and 25% of the different views, with the other maps of the model already trained. At the end it achieves a remarkable invariance with respect to viewpoint, as visible in some examples in Fig. 3. The global figures of LOC invariance with respect to several transformations, over all objects, is shown in Tab. 1. As said in the introduction, there is no representation independent from the view, therefore the invariance is not absolute but within a limited range, very different configuration of the same object will have different representations in LOC.

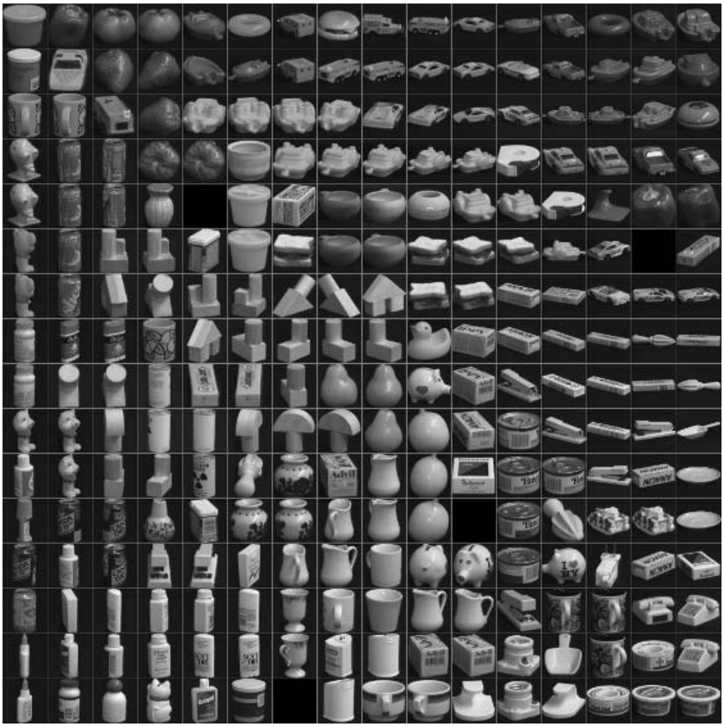


Fig. 4. Organization of objects in the OBJ map of the model. Each neuron of the map is labeled using the image with an output vector nearest to the neuron vector.

The highest map in the model is called OBJ, and is of SOM type. It processes as vector input the whole content of LOC, ignoring the spacial organization of the data. The use of a SOM architecture is justified by the abstract nature of this last map, which is not actually located in any specific part of the human brain.

The organization of all objects in OBJ, shown in Fig. 4, is the overlap of several coexisting ordering principles: color, shape, symmetries; producing a consistent categorization of most objects.

4 Conclusions

A model for simulating the emergence of visual recognition capability has been presented. Several simplifications make the model still far from the complexity of human recognition: the segregation of processes in areas, the lack of back-projections, and of course the simplification of the neural computations. Despite these limitations, the model is able to achieve recognition capabilities, reproducing some of the fundamental computational steps, without any explicit modeling of the processing functions necessary for this goal, only thanks to basic neural mechanisms.

References

[1] J. A. Bednar. *Learning to See: Genetic and Environmental Influences on Visual Development*. PhD thesis, University of Texas at Austin, 2002. Tech Report AI-TR-02-294.

[2] B. Chapman, M. P. Stryker, and T. Bonhoeffer. Development of orientation preference maps in ferret primary visual cortex. *Journal of Neuroscience*, 16:6443–6453, 1996.

[3] J. L. Dannemiller. A test of color constancy in 9- and 20-weeks-old human infants following simulated illuminant changes. *Developmental Psychology*, 25:171–184, 1989.

[4] J. E. Dowling. *The Retina: An Approachable Part of the Brain*. Cambridge University Press, Cambridge (UK), 1987.

[5] S. Edelman. *Representation and Recognition in Vision*. MIT Press, Cambridge (MA), 1999.

[6] S. Edelman and H. H. Bülthoff. Orientation dependence in the recognition of familiar and novel views of 3d objects. *Vision Research*, 32:2385–2400, 1992.

[7] D. C. V. Essen, J. W. Lewis, H. A. Drury, N. Hadjikhani, R. B. Tootell, M. Bakircioglu, and M. I. Miller. Mapping visual cortex in monkeys and humans using surface-based atlases. *Vision Research*, 41:1359–1378, 2001.

[8] M. J. Farah and G. K. Aguirre. Imaging visual recognition: PET and fMRI studies of the functional anatomy of human visual recognition. *Trends in Cognitive Sciences*, 3:179–186, 1999.

[9] K. Grill-Spector, Z. Kourtzi, and N. Kanwisher. The lateral occipital complex and its role in object recognition. *Vision Research*, 41:1409–1422, 2001.

[10] L. C. Katz and E. M. Callaway. Development of local circuits in mammalian visual cortex. *Science*, 255:209–212, 1992.

[11] T. Kohonen. *Self-Organizing Maps*. Springer-Verlag, Berlin, 1995.

[12] C. E. Landisman and D. Y. Ts'o. Color processing in macaque striate cortex: Relationships to ocular dominance, cytochrome oxidase, and orientation. *Journal of Neurophysiology*, 87:3126–3137, 2002.

[13] N. K. Logothesis, J. Puals, and T. Poggio. Shape representation in the inferior temporal cortex of monkeys. *Current Biology*, 5:552–563, 1995.

[14] D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W. H. Freeman, San Francisco (CA), 1982.

[15] H. Murase and S. Nayar. Visual learning and recognition of 3-d object by appearence. *International Journal of Computer Vision*, 14:5–24, 1995.

[16] S. R. Quartz and T. J. Sejnowski. The neural basis of cognitive development: a constructivist manifesto. *Behavioral and Brain Science*, 20:537–596, 1997.

[17] J. Sirosh and R. Miikkulainen. Topographic receptive fields and patterned lateral interaction in a self-organizing model of the primary visual cortex. *Neural Computation*, 9:577–594, 1997.

[18] B. H. Tootell, E. Switkes, M. S. Silverman, and S. L. Hamilton. Functional anatomy of the macaque striate cortex. II. retinotopic organization. *Journal of Neuroscience*, 8:1531–1568, 1988.

[19] C. von der Malsburg. Self-organization of orientation sensitive cells in the striate cortex. *Kibernetic*, 14:85–100, 1973.

[20] S. Zeki. Colour coding in the cerebral cortex: The reaction of cells in monkey visual cortex to wavelenghts and colours. *Neuroscience*, 9:741–765, 1983.

Implicit Relevance Feedback from Eye Movements

Jarkko Salojärvi[1], Kai Puolamäki[1], and Samuel Kaski[1,2]

[1] Neural Networks Research Centre, Helsinki University of Technology,
P.O. Box 5400, FI-02015 HUT, Finland
forename.surname@hut.fi

[2] Department of Computer Science, University of Helsinki,
P.O. Box 68, FI-00014 University of Helsinki, Finland
forename.surname@cs.helsinki.fi

Abstract. We explore the use of eye movements as a source of implicit relevance feedback information. We construct a controlled information retrieval experiment where the relevance of each text is known, and test usefulness of implicit relevance feedback with it. If perceived relevance of a text can be predicted from eye movements, eye movement signal must contain information on the relevance. The result is that relevance can be predicted to a considerable extent with discriminative hidden Markov models, and clearly better than randomly already with simple linear models of time-averaged data.

1 Introduction

A search engine could be improved by an algorithm which models the interests of a user. Such an algorithm would be *proactive*; it would predict the needs of the user and adapt its own behavior accordingly [1]. The usual way to learn the interests of the user would be to ask, after each document, whether the user found it relevant, and to learn the user's preferences from the answers. However, giving this kind of explicit feedback is laborious.

Alternatively, relevance can be inferred from implicit feedback derived traditionally from document reading time, or by monitoring other behavior of the user (such as saving, printing, or selecting of documents). The problem with the traditional sources is that the number of feedback events is relatively small.

In this paper we explore whether the traditional sources of implicit relevance information could be complemented with eye movements. We construct an experimental information retrieval setup where relevance of the texts is known. The measured eye movement data corresponding to each text will then have a known label, and the data can be used as a learning data set for machine learning methods. Machine learning will be used for selecting a good set of features and for learning time series models to predict relevance of new measurements.

We make an assumption that human attention patterns correlate with relevance; at the simplest, people tend to pay more attention to objects they find relevant or interesting. The reason why gaze direction provides an indicator of

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 513–518, 2005.

the focus of attention lies in the physiology of the eye. Accurate viewing is possible only in the central *fovea* area (only 1–2 degrees of visual angle) where the density of photoreceptive cells is highly concentrated. A scene needs therefore to be inspected with a sequence of alternating *saccades* (rapid eye movements) and *fixations* (the eye is fairly motionless). Information on the environment is mostly gathered during fixations, and fixation duration is known to be correlated with the complexity of the object under inspection. A simple physiological reason for this is that the amount of information the visual system is capable of processing is limited. In other words, we assume that (visual) attention lies there where the amount of gathered information is larger. This is justified in a multitude of psychological experiments [2].

Related Work. Eye movements have traditionally been exploited as an alternative input modality in user interfaces, for instance in eye typing (cf. [3]). Another increasingly popular application area is usability studies, where the goodness of a user interface or a web page is evaluated by monitoring natural behavior of the users. This form of implicit feedback information gathered from eye movements has been analyzed using features computed for larger areas of interest [4], such as images or captions of text.

As far as we know, eye movements have been used in information retrieval applications in only two previous studies. The first is our earlier preliminary work [5,6], which is extended in this paper by thorough experiments with a large number of subjects, better equipment that solves our earlier calibration problems, and more detailed analysis of the results. The goal of the second related work [7] was different: to investigate with quantitative measures how users behave in a real, less-controlled information retrieval task.

2 Information Retrieval Experiment

In a typical information retrieval setup the user types in keywords to a search engine and is then given a list of results, for instance titles of scientific articles, that possibly contain the information the user is looking for. Some of the proposed titles may be totally irrelevant, some of them handle the correct topic, and only few are links to articles that the user eventually reads. Our experimental setting for collecting eye movement data was designed to simulate this natural situation, with the difference that in our case the relevance is known.

The subject was first shown a question, and then a list of ten sentences, one of which contained the correct answer (C). Five of the sentences were known to be irrelevant (I), and four relevant to the question (R). The task of the subject was to identify the correct answer, press ‘enter’ (this ended the eye movement measurement), and then type in the associated number in the following display. Each of the eleven test subjects carried out 50 assignments. The assignments were in Finnish, the mother tongue of the subjects. Eye movements were measured with a Tobii 1750 eye tracker with a screen resolution of 1280x1024 and a sampling rate of 50 Hz.

Preprocessing. The raw eye movement data (x and y coordinates of the gaze direction) was segmented into a sequence of fixations and saccades by a window-based algorithm (software from Tobii).[1] Each fixation was first assigned to the nearest word. A set of 22 features were computed from the eye movement trajectory for each word [10]; similar features are used in psychological studies of reading [2]. The resulting feature vector sequence was then segmented to subsequences corresponding to different sentences, and a label was assigned to each subsequence according to the known class of the sentence.

Sentence-specific averages of the features was then computed. Linear Discriminant Analysis (LDA) was applied to the averaged features in order to select the set of features that best predict relevance for left-out data. The LDA also provided a baseline classification accuracy.

In the time-series models, described in more detail below, the resulting set of features were modeled with the following exponential family distributions: (1) One or many fixations within the word (binomial). (2) Logarithm of total fixation duration on the word (assumed Gaussian). (3) Reading behavior (multinomial): skip next word, go back to already read words, read next word, jump to an unread line, or last fixation in an assignment.[2]

3 Models

The simplest method to classify the eye movement data is to disregard the time dependency between data samples and compute averages of the eye movement features. This gives sentence-specific feature vectors, one per sentence. In this paper, the averaged vectors were classified with Linear Discriminant Analysis. More fine-grained cues of relevance can be sought with models that take into account the time series nature of the eye movement data.

Hidden Markov Models. The simplest model that takes the sequential nature of data into account is a two-state hidden Markov model (HMM). A separate model was optimized individually for each class, by maximizing the log-likelihood of the data Y given the model and its parameters Θ, that is, $\log p(Y|\Theta)$. The HMMs are trained with the Baum-Welch (BW) algorithm [8]. In a prediction task the models of different classes were combined into a maximum a posteriori (MAP) prediction.

Discriminative Hidden Markov Models. In speech recognition, where HMMs have been extensively used for decades, the current state-of-the-art HMMs are discriminative. Discriminative models aim to predict the relevance $B =$

[1] 20-pixel window size and a minimum duration of 80 ms.

[2] Feature selection was carried out with methods that use averaged data (from eigenvectors of LDA), since currently there are no methods that can simultaneously both do this and model the time series. We therefore chose to model a representative set of features which can be used to construct the best discriminating averaged measures.

$\{I, R, C\}$ of a sentence, given the observed eye movements Y. Formally, we optimize $\log p(B|Y, \Theta)$. In discriminative HMMs, a set of states or a certain sequence of states is associated with each class. This specific state sequence then gives the probability of the class, and the likelihood is maximized for the teaching data, versus all the other possible state sequences in the model [9]. The parameters of the discriminative HMM can be optimized with an extended Baum-Welch (EBW) algorithm, which is a modification of the original BW algorithm.

We model eye movements with a two-level discriminative HMM, where the first level models transitions between sentences, and the second level transitions between words within a sentence. Viterbi approximation is used to find the most likely path through the second level model (transitions between words in a sentence), and then the discriminative Extended Baum-Welch optimizes the full model.

Voting. The HMMs produce probabilities for the relevance classes (I, R, C) for each viewed sentence. However, the users may look at a sentence several times, and the resulting probabilities need be combined in a process we call *voting.*

We constructed a log-linear model for combining the predictions. Assume that the sentence-specific probability distribution, $p(B|Y_{1...K})$, can be constructed from the probability distributions of the kth viewings of the sentence, $P(B|Y_k)$ (obtained as an output from a Markov model) as a weighted geometric average, $p(B|Y_{1...K}, \alpha) = Z^{-1} \prod_k p(B|Y_k)^{\alpha_{Bk}}$, where Z is a sentence-specific normalization factor and the parameters α_{Bk} are non-negative real numbers, found by optimizing the prediction for the training data. The predicted relevance of a sentence is then the largest of $p(I)$, $p(R)$ and $p(C)$.

It is also possible to derive a simple heuristic rule for classification by assuming that decision of relevance is made only once while reading the sentence. We will call this rule maxClass, since for each sequence we will select the maximum of the predicted relevance classes (with ordering $I < R < C$). A simple baseline for the voting schemes is provided by classifying all the sequences separately (i.e., no voting).

4 Results

The prediction accuracy was assessed with 50-fold cross validation, in which each of the assignments was in turn used as a test data set. In order to test how the method would generalize to new subjects, we also ran an 11-fold cross validation where each of the subjects was in turn left out. Table 1 lists the classification accuracies, that is, the fraction of the viewed sentences in the test data sets for which the prediction was correct. The methods generalize roughly equally well both to new assignments and to new subjects. The performance of the two different voting methods (log-linear and maxClass) seem to be nearly equal, with log-linear voting having a slight advantage.

Table 2 shows the confusion matrix of the discriminative HMMs. Correct answers (C) are separated rather efficiently. Most errors result from misclassifying

Table 1. Prediction accuracies of different models. The baseline is given by the the "dumb model," which classifies all sentences to the largest class I. Differences between LDA and dumb classifier, and simple HMMs and LDA, tested significant ($P < 0.01$, McNemar's test), as well as the difference between discriminative HMM and simple HMMs in the case of leave-one-assignment-out validation (with log-linear voting). Left column: obtained by 50-fold cross-validation where each of the assignments was left out in turn as test data. Right column: Obtained by 11-fold cross-validation where each of the subjects was in turn left out to be used as test data.

Method	**Accuracy (%)** (leave-one-assignment-out)	**Accuracy (%)** (leave-one-subject-out)
Dumb	47.8	47.8
LDA	59.8	57.9
simple HMMs(no vote)	55.6	55.7
simple HMMs(maxClass)	**63.5**	**63.3**
simple HMMs(loglin)	**64.0**	**63.4**
discriminative HMM(loglin)	**65.8**	**64.1**

Table 2. Confusion matrix showing the number of sentences classified by the discriminative HMM, using loglinear voting, into the three classes (columns) versus their true relevance (rows). Cross validation was carried out over assignments. The percentages (in parentheses) denote row- and column-wise classification accuracies.

	Prediction		
	I (62.4 %)	R (61.8 %)	C (90.1 %)
I (77.3 %)	1432	395	25
R (43.6 %)	845	672	24
C (92.2 %)	17	21	447

relevant sentences (R) as irrelevant (I). It is also possible to compute precision and recall measures commonly used in information retrieval by treating correct answers as the relevant documents. The resulting precision rate is 90.1 %, and recall rate 92.2 %.

5 Conclusion

Our results show that relevance information can be inferred from eye movement signals. Both the time series nature of the data and the discriminative nature of the task should be taken into account when constructing models for eye movements. An interesting topic for further research would be to inspect whether the HMM is capable of differentiating cognitive processes associated with reading.

The work will be continued in the form of a PASCAL eye movement challenge [10]. This will hopefully result in a toolbox of robust and efficient methods for relevance extraction.

Acknowledgments

The authors thank Jaana Simola for carrying out the eye movement measurements, and Lauri Kovanen for implementing part of the code. We also thank Maria Sääksjärvi and Johanna Gummerus, Center for Relationship Marketing and Service Management at Swedish School of Economics and Business Administration (Hanken), for allowing us to use their eye tracker. This work was supported in part by Academy of Finland, decision 79017, and the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views. All rights are reserved because of other commitments.

References

1. Tennenhouse, D.: Proactive computing. Commun. ACM **43** (2000) 43–50
2. Rayner, K.: Eye movements in reading and information processing: 20 years of research. Psychological Bulletin **124** (1998) 372–422
3. Ward, D.J., MacKay, D.J.: Fast hands-free writing by gaze direction. Nature **418** (2002) 838
4. Goldberg, J.H., Stimson, M.J., Lewenstein, M., Scott, N., Wichansky, A.M.: Eye tracking in web search tasks: design implications. In: ETRA '02: Proceedings of the symposium on Eye tracking research & applications, ACM Press (2002) 51–58
5. Salojärvi, J., Kojo, I., Simola, J., Kaski, S.: Can relevance be inferred from eye movements in information retrieval? In: Proceedings of WSOM'03, Workshop on Self-Organizing Maps. Kyushu Institute of Technology, Kitakyushu, Japan (2003) 261–266
6. Salojärvi, J., Puolamäki, K., Kaski, S.: Relevance feedback from eye movements for proactive information retrieval. In Heikkilä, J., Pietikäinen, M., Silvén, O., eds.: workshop on Processing Sensory Information for Proactive Systems (PSIPS 2004), Oulu, Finland (2004)
7. Granka, L.A., Joachims, T., Gay, G.: Eye-tracking analysis of user behavior in WWW search. In: Proceedings of SIGIR'04. ACM Press (2004) 478–479
8. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE **77** (1989) 257–286
9. Povey, D., Woodland, P., Gales, M.: Discriminative MAP for acoustic model adaptation. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP'03). Volume 1. (2003) 312–315
10. Salojärvi, J, Puolamäki, K., Simola, J., Kovanen, L., Kojo, I., Kaski, S.: Inferring relevance from eye movements: feature extraction. Helsinki University of Technology, Publications in Computer and Information Science, Report A82 (2005). `http://www.cis.hut.fi/eyechallenge2005/`

Image Segmentation by Complex-Valued Units

Cornelius Weber and Stefan Wermter

Hybrid Intelligent Systems, SCAT, University of Sunderland, UK

Abstract. Spike synchronisation and de-synchronisation are important for feature binding and separation at various levels in the visual system. We present a model of complex valued neuron activations which are synchronised using lateral couplings. The firing rates of the model neurons correspond to a complex number's absolute value and obey conventional attractor network relaxation dynamics, while the firing phases correspond to a complex number's angle and follow the dynamics of a logistic map. During relaxation, we show that features with strong couplings are grouped by firing in the same phase and are separated in phase from features that are coupled weakly or by negative weights. In an example, we apply the model to the level of a hidden representation of an image, thereby segmenting it on an abstract level. We imply that this process can facilitate unsupervised learning of objects in cluttered background.

1 Introduction

Object recognition is a key task in everyday situations and for robotics applications. Unsupervised learning of object classes from natural data is performed by young living beings and has a chance of becoming a convenient and flexible method of learning to categorise sensory data by an artificial agent. A hierarchy of increasingly complex feature detectors is one aspect of the visual recognition process. In many models, such a feature extracting step performs an almost linear vector transformation. So in order to achieve noticeable achievements in their serial application, a strong non-linearity must be introduced at every level.

The non-linear response properties observed in cortical cells are explained in model studies by intra-area horizontal connections. A mathematical advantage to implementing these as an attractor network is that its activations recover noisy input with maximum likelihood [1]. Contrast-invariant orientation tuning curves and shift invariant responses can be obtained [2], as in V1 neurons.

Unsupervised learning of objects is possible if objects are shown on a plain background, but still fails with a noisy background [3]. While the competitive effect of the attractor network reduces background noise, in realistic conditions further percepts are the rule in addition to the object to be learnt. We therefore aim to separate these simultaneous percepts in the dimension of *phase* in order to separate an object from its background. In a hierarchical model this would allow higher levels to learn only the object at certain phases or only background elements at other phases and will facilitate unsupervised learning of objects.

Detailed spiking neuron models are attractive for segmentation purposes. In addition to the neurons' firing *rate* their code provides information that can

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 519–524, 2005.

be used for mutual binding and segregation. Computationally it is efficient to incorporate these additional capabilities of spikes in a single variable per neuron which we call a neuron's *phase*. A process to adjust this variable efficiently is deterministic chaos with a dual role of *(i)* supplying a process of pattern creation by synchronising the phases of coupled units and at the same time *(ii)* revolting against convergence into stereotyped synchronised states [4].

Our approach has the following characteristics: *(i)* Coupling strengths are represented by connection weights that can be trained to represent correlated activations and which can be negative. *(ii)* The weighting of the neural inputs is performed by complex number algebra. Complex-valued neural networks have advantages for chaotic and brain-like systems, image processing and quantum devices [6]. We identify the absolute value of a complex neuron activation with its firing rate and the phase of its activation with the phase at which its spikes are emitted. *(iii)* The *rates* follow a conventional update dynamics of a recurrent neural net. *(iv)* A logistic map provides the chaotic dynamics to synchronise and separate the *phases*, as in the "Divide and Conquer" model [5]. *(v)* Finally, we show an application where an image is segmented on its "hidden", first cortical representation on area V1, similar to the competitive layer model [7].

2 Real Valued Relaxation Procedure

We use one layer of fully connected units to define a recurrent update dynamics for the neurons' rates and phases. After some, possibly random, initialisation of the firing rates, the rate r_k of any unit k is governed over time steps t by

$$r_k(t+1) \;=\; f(\textstyle\sum_j w_{kj} r_j(t) - \theta_k) \tag{1}$$

where w_{kj} is the connection weight from unit j to unit k and θ_k is its threshold. The transfer function is $f(x) = 1/(1+e^{-x})$. A learning rule for the weights and thresholds is given in [8] and weights which sustain a bell-shaped hill of activation rates will have a Mexican hat shaped profile. Thus, we have a continuous valued attractor network and since weights are approximately symmetric, $w_{kj} \approx w_{jk}$, our intuition is that activations r_k relaxate to a stable state corresponding to a minimum of some energy function. In the following, we will introduce a second variable, the phase φ_k, with different dynamics.

3 Complex Valued Interactions

A complex number as displayed in Fig. 1 a) can be written as $z \;=\; x+iy \;=\; r\,e^{i\varphi}$ with $i^2 \;=\; -1$ and the relations $r^2 \;=\; x^2+y^2$ and $\tan\varphi \;=\; \frac{y}{x}$. We express our neuronal activation as $z_k = r_k\,e^{i\varphi_k}$ where the complex number's length r_k is the neuron's firing rate and its phase φ_k is the phase at which the neuron spikes. Similar phases of two neurons would correspond to similar firing times if their rates were the same, however, we regard these phases as abstract.

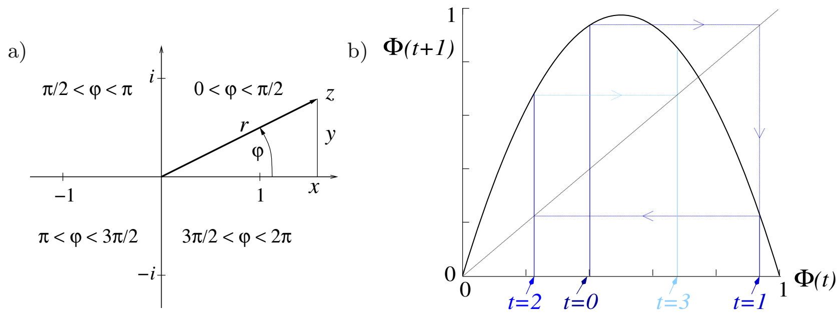


Fig. 1. a) A complex number z can be expressed by Cartesian coordinates x, y or polar coordinates r, φ. We identify its length r with a neuronal firing rate and angle φ with a firing phase. The ranges of φ are displayed as used in the text. **b)** The logistic map. Iterations according to $\varphi(t+1) = 3.9\,\varphi(t)\,(1 - \varphi(t))$, indicated for $t = 0 \ldots 3$ starting at $\varphi(t\!=\!0) = 0.4$, will lead to a desired chaotic behaviour.

The dynamics of the rates follows Eq. 1 and is independent of the phases. In the following we will first define how phases between neurons interact (Eq. 2) and then impose a local update dynamic on every neuron (Eq. 3).

At the beginning of the relaxation procedure, the neurons' phases φ_k are initialised with random values between 0 and 2π. Then at each relaxation step a neuron receives an influence from all other neurons, which we express as:

$$z_k^{wf} = \textstyle\sum_j w_{kj}\, r_j\, e^{i\varphi_j} \tag{2}$$

This weighted field is a complex number which is a sum of the complex number contributions by the other neurons weighted by the connection weights w_{kj} which are real values. Using $e^{i\varphi_j} = \cos\varphi_j \, + \, i\, \sin\varphi_j$ it can be expressed as:

$$z_k^{wf} \;=\; \textstyle\sum_j w_{kj} r_j \, \cos\varphi_j \, + \, i \, \sum_j w_{kj} r_j \, \sin\varphi_j \;\equiv\; x_k^{wf} \, + \, i\, y_k^{wf}$$

We obtain the phase of the neuron's weighted field as: $\varphi_k^{wf} = \mathrm{atan}\, \frac{y_k^{wf}}{x_k^{wf}}$ which we shift to range between 0 and 2π according to Fig. 1 a).

4 Logistic Coupled Map

The logistic map maps a value between 0 and 1 to another, different value within this interval, as shown in Fig. 1 b). Iterative application leads to desired chaotic development of these values for most settings of the map parameter A between 3.57 and 4.0. Nevertheless, if several values undergoing this mapping are coupled, they can maintain proximity and thus display structured mutual behaviour while displaying chaotic individual behaviour [4].

Since the logistic map takes values ranging between 0 and 1 while the neurons' phases range from 0 to 2π, we make the technical definitions: $\Phi_k \equiv \frac{\varphi_k}{2\pi}$, $\Phi_k^{wf} \equiv \frac{\varphi_k^{wf}}{2\pi}$.

We will now use the scaled phase Φ_k^{wf} of the weighted field at each neuron k in order to determine its scaled phase Φ_k at the next iteration time step. This is done via the logistic map:

$$\Phi_k(t+1) = A\,\Phi_k^{wf}(t)\,(1 - \Phi_k^{wf}(t)) \tag{3}$$

where its actual phase value φ_k is scaled back to the range from 0 to 2π. We have set $A = 3.9$. Having obtained the phases at the next time step, another iteration for the phases is performed starting with computing the weighted field (Eq. 2). While the rates develop concurrently according to Eq. 1 to a stable state, the phase values never converge.

5 Network Activation with Synchronising Phases

Figure 2 a) shows the activations of a network with 25 units as their rates have converged to a stable state according to Eq. 1. This is in no way influenced by the phases but only by the weight profile, which is also displayed for one neuron.

The weight profile with strong positive weights between neighbouring units should synchronise the phases between such connected units. A single unit's phase behaves random-like from one time step to the next. Fig. 2 b) shows a plot of time averaged phase differences between pairs of units, while the network is maintaining the rates shown in Fig. 2 a). It shows that adjacent units within the hill of activation have similar phases, while adjacent units at its boundary have differing phases. The phases are thus clustering regions of strong activity that are linked by strong positive weights (phase influence is weighted by weights *times* rates, cf. Eq. 2). Regions with negative connections to such a cluster have differing phases, as we see units with zero activation sharing an own phase. Note that since adjacency is defined functionally by mutual strong positive connections, long-range connections could mediate synchronisation over large distances.

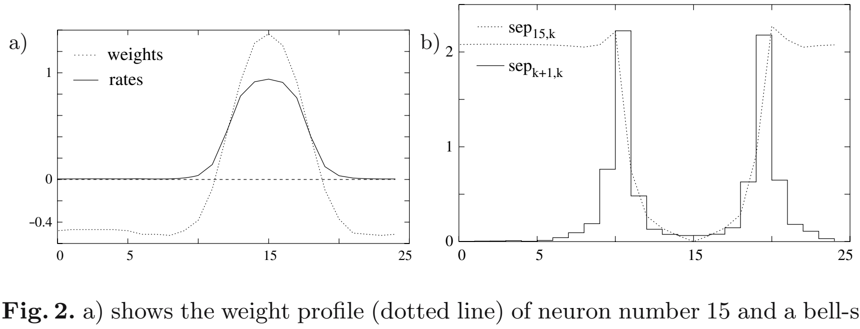


Fig. 2. a) shows the weight profile (dotted line) of neuron number 15 and a bell-shaped hill of activation *rates* as sustained by the network (solid line). b) shows the average *phase* separation $sep_{k+1,k} = \langle|\varphi_{k+1} - \varphi_k|\rangle$ between neighbouring units (solid line) and $sep_{15,k} = \langle|\varphi_{15} - \varphi_k|\rangle$ between any unit and unit 15 (dotted line), where the $\langle . \rangle$-brackets denote a time average over 500 iteration steps while sustaining the rates in a).

6 Segmentation in a Feature Space

In the following experiment we apply the attractor network to the activations of neurons which have been trained to extract features from natural images and which thereby resemble V1 "simple cells" [2]. Given an image as input, these units have sparse activations with values between 0 and 1 while responding to edges and colour features. V1 lateral weights have been trained on the same set of natural images to memorise these codes in the attractor activation patterns. The resulting weights are short-range excitatory and long-range inhibitory along cortical distance, as well as in feature space of orientation and spatial frequency [2]. This leads, during relaxation, to focused patterns of the activation rates on the simulated V1, after initialising with a somewhat irregular activity pattern obtained from presenting an image. From this V1 representation, a virtual reconstruction of the image can be obtained by projecting these rates back to the image. Fig. 3 a) shows an example.

Fig. 3 c) shows the hidden units' activations where each frame shows only those units' activations r_k which have a phase φ_k within a range shown in Fig. 3 b). The active units in each frame are thus a subset of all active units shown in Fig. 3 a), middle. It can be seen that within any selected phase range, preferably units from a certain region are active. The functions of the neurons within these active clusters can be seen by projecting their rates to the image. As a result, Fig. 3 d) shows partial reconstructions of the image which is hereby segmented into elements which belong together by having similar phases on the model V1. At different phases we find elements of the background (frames 1 and 2) or of the ball at the right (frames 4 and 5). This implies a segmentation which accounts for learnt proximities (via the V1 lateral weights) in an abstract representation of an image with the potential of separating objects from their background.

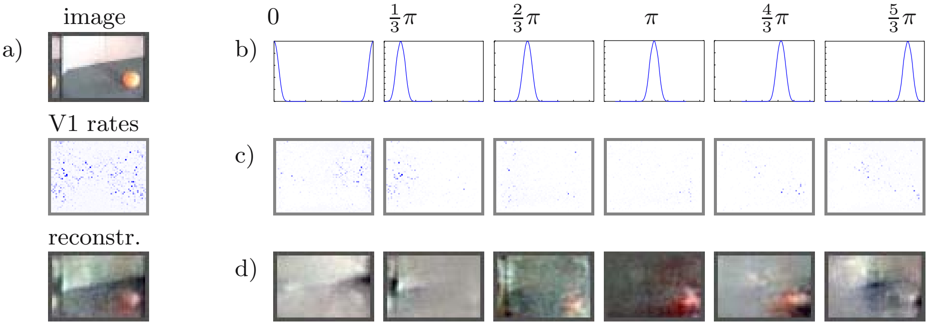


Fig. 3. a) shows full representations. Top, the original image, middle, the rates of the full hidden code and bottom, the reconstruction of the image from the full hidden code. b),c),d) show partial phase-dependent representations. b) shows Gaussian-like functions on an axis of 0 to 2π used to determine which phases contribute to the hidden code presented in c) and thus to the image reconstruction in d). c) shows the partial hidden code corresponding to selected phases and d) shows their partial reconstruction from those units in c) which are active at the selected phases.

7 Discussion

We have demonstrated a network of simplified *rate* and *phase* coding neurons which segments a neural code efficiently using the connection strengths between the units. The computational load is that of a network in which two activation values develop concurrently. The most demanding operations are the scalar products in Eqs. 1 and 2, while all other computations are local.

Since image segmentation involves top-down directed information flow, how does our model for intra-area lateral connections deal with this? Previously we have extended the lateral connections to link a "what"- to a "where" area for object localisation [9]. In the cortex, such lateral connections correspond to those originating from pyramidal cells in layers 2/3 of the cortex and arriving in the same layer, possibly in a different area. Hierarchically arranged areas have characteristically asymmetric connections, however, they also have characteristic horizontal connections and only with increasing hierarchical level difference the intensity of these horizontal connections decreases [10]. Thus, connections of a horizontal character may relay top-down information. If we would apply the lateral connections of our model to a larger hierarchical model, therefore, we might observe top-down influences such as stabilisation of consistent attractors.

Acknowledgements. This work is part of the MirrorBot project supported by the EU in the FET-IST programme under grant IST-2001-35282.

References

[1] Deneve, S., Latham, P., Pouget, A.: Reading population codes: a neural implementation of ideal observers. Nature Neurosci. **2** (1999) 740–5

[2] Weber, C.: Self-organization of orientation maps, lateral connections, and dynamic receptive fields in the primary visual cortex. In Dorffner, G., Bischof, H., Hornik, K., eds.: Proc. ICANN, Springer-Verlag Berlin Heidelberg (2001) 1147–52

[3] Stringer, S., Rolls, E.: Position invariant recognition in the visual system with cluttered environments. Neural Networks **13** (2000) 305–15

[4] van Leeuwen, C., Steyvers, M., Nooter, M.: Stability and intermittency in large-scale coupled oscillator models for perceptual segmentation. J. Mathematical Psychology **41** (1997) 319–44

[5] Raffone, A., van Leeuwen, C.: The "divide and conquer" model of image segmentation: object-bounded synchrony propagation in coupled map lattices. In: Proc. KES Knowledge-Based Intelligent Information & Engineering Systems. (2002)

[6] Hirose, A., ed.: Complex-Valued Neural Networks: Theories and Applications. World Scientific Publishing Co. (2003)

[7] Ontrup, J. Wersing, H., Ritter, H.: A computational feature binding model of human texture perception. Cognitive Processing **5** (2004) 32–44

[8] Zhang, K.: Representation of spatial orientation by the intrinsic dynamics of the head-direction cell ensemble: A theory. J. Neurosci. **16** (1996) 2112–26

[9] Weber, C., Wermter, S.: Object localization using laterally connected "what" and "where" associator networks. In: Proc. ICANN/ICONIP, Springer (2003) 813–20

[10] Felleman, D., Van Essen, D.: Distributed hierarchical processing in the primate cerebral cortex. Cerebral Cortex **1** (1991) 1–47

Cellular Neural Networks for Color Image Segmentation

Anna Wilbik

Systems Research Institute,
Polish Academy of Sciences,
ul. Newelska 6, 01-447 Warsaw, Poland
wilbik@ibspan.waw.pl

Abstract. In this paper we apply cellular neural networks for color image segmentation. Color aerial photographs will be analyzed. Two types of color models: RGB and HSV will be taken into account and compared. In resulting images we will distinguish some objects like houses, roads, trees and others. The selection of the objects will be based on the color value. We show that the choice of color model influences the results.

1 Introduction

Aerial photographs are used mainly for creating and drawing orthophotomaps. In order to prepare these maps some panchromatic photos of professional quality are used. They are costly to obtain, since specific tools are required. However, simple photos taken from the air may be also useful. Such images are used for places identification in air-sports like aviation, gliding, paragliding.

Neural networks and cellular neural networks (CNNs) are regarded as useful tools for image processing (eg. [7,18]). CNNs were introduced by L.O. Chua and L. Yang [5,6]. Recent years several applications of CNNs for image processing, like image restoration (eg. [5,6]), image segmentation (eg. [8,12,16,17]), object recognition (eg. [13]), or visual reconstruction (eg. [14]) have been proposed and tested. CNNs appeared to be useful in medical image processing [1,2] and for monitoring volcanoes [3].

Color image processing causes more problems than monochrome image processing, as the color photos have some nonlinear information. During color image processing, the color images are transformed into three monochrome images: red, green and blue (eg. [8]). Each monochrome image is then processed independently.

In this paper we propose to use a different method of transformation of color image to three monochromatic ones, namely we use HSV color model. In the experiment it will be shown that the results are often better than in case when RGB model is considered, as some new objects (like trees) stand out. Inspired by results from [12] we apply CNN, for converting monochromatic images to black-and-white ones, for image segmentation.

The paper is structured as follows. In Section 2 a general idea of cellular neural networks will be presented. In Section 3 basic color models are discussed.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 525–530, 2005.

The experiment and its results are described in Section 4. The paper is completed by some concluding remarks.

2 Cellular Neural Networks

A cellular neural network can be described as a n-dimensional array of dynamic systems called cells, denoted as $c(i, j)$ [15]. A cell is connected to another one ($c(i, j)$), if it belongs to the neighborhood $N_r(i, j)$ of the cell $c(i, j)$ defined by:

$$N_r(i, j) = \{c(k, l) : |k - i| \leq r \wedge |l - j| \leq r\},$$

where r is the radius of the neighborhood [5]. Moreover, all state variables are continuous-valued signals [15]. An example of two-dimensional cellular neural network is depicted in Figure 1.

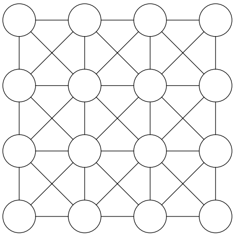


Fig. 1. A topology of a two-dimensional cellular network

Basically, it is assumed that the signal is transformed constantly in time. However, Discrete Time Neural Network (DTCNN), based on an assumption that the values of stated and output signals are changed in discrete, evenly spaced time points, was introduced in [9]. In the computer simulation a DTCNN is used, beacuse of its convenience. The dynamic of DTCNN can be characterized by a set of the following equations [11]:

$$\begin{aligned} x_{ij}[(n+1)\Delta t] = x_{ij}[n\Delta t] \ &+ \sum_{k=-r}^{r} \sum_{l=-r}^{r} A_{i,j;i+k,j+l}\ y_{i+k,j+l}[n\Delta t] \ + \\ &+ \sum_{k=-r}^{r} \sum_{l=-r}^{r} B_{i,j;i+k,j+l}\ u_{i+k,j+l} \ + \ I \end{aligned} \tag{1}$$

$$y_{ij} = f(x_{ij}) = \frac{1}{2}\left(|x_{ij} + 1| - |x_{ij} - 1|\right) \tag{2}$$

$$x_{ij} = E_{ij} \tag{3}$$

$$|x_{ij}(t = 0)| \leq 1 \tag{4}$$

$$|u_{ij}(t)| \leq 1 \tag{5}$$

Equation (1) is state equation for a cell, (2) and (3) describe output and input equations, respectively. Constraint conditions are represented by (4) and (5).

Information may be interchanged through both outputs and external inputs of the cell. The signals are weighted by parameters $A_{i,j;i+k,j+l}$ and $B_{i,j;i+k,j+l}$, which are called feedback operator and control operator, respectively. I is independent current source, which is constant in time.

A scheme of a cell is depicted in Figure 2, where y_{ij} is output signal of a cell, u_{ij} is an input signal and x_{ij} is a state of a cell [10].

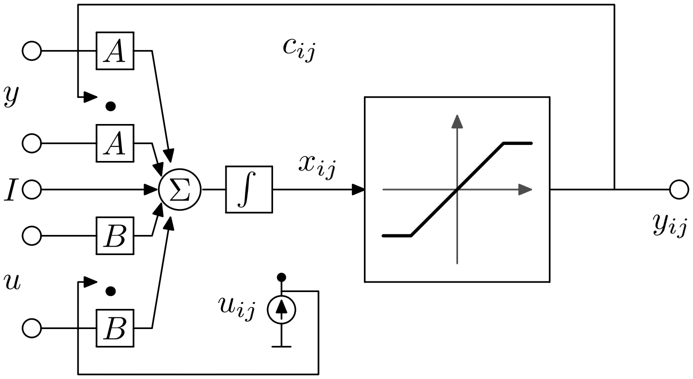


Fig. 2. Block scheme of a cell

Values of weights and the scheme of connection are equal for all the cells in the network. This means that for any two cells $c(i,j)$ and $c(p,q)$ we have [10]:

$$A_{i,j;i+k,j+l} = A_{p,q;p+k,q+l} \qquad B_{i,j;i+k,j+l} = B_{p,q;p+k,q+l}$$

The result of the performance of the network is the set of outputs signals, when the network reaches state of equilibrium, described by the following equation:

$$\forall i,j \qquad y_{ij}(t) = y_{ij}(t+1)$$

3 Color Models

Color images provide more information than monochrome images. The techniques used for gray level images are extended for color images by using components of RGB model or their transformations [4].

Model RGB is one of the first practical models of area of colors and contains a recipe for creation of colors. This model is resulting from the receiving abilities of eye and is based on the fact that the impressions of almost all the colors in eye can be evoked by mixing in the fixed proportions of three selected clusters of light of the properly chosen width of spectrum. Basing on this model the pictorial lamps (kinescopes) work up to nowadays.

The RGB model is convenient for display, however there are some disadvantages like high correlation among the components. Moreover, colors are only approximations, satisfactory enough for humaneye perception.

HSV color model is more user-oriented model then RGB. In HSV model the color can describe itself by means of a single representative of one monochromic wave of light Hue with the specification of Saturation measure for wave, and the specification of Value for Brightness level.

In HSV model it is assumed that the elements of model are orthogonal, independently from each other, except Hue definiteness for the case Saturation equals 0.

4 Description of the Experiment

In this experiment we will segment color photos. At the begining every pixel of considered image is decomposed to three components (Red, Green, Blue for RGB model and Hue, Saturation and Value for HSV model). Every image transormed in that way can be presented as a monochromatic one, with only one components' value changing. Such images constitute the network's input.

We use DTCNN with the following cloning template[1]:

$$A = \begin{bmatrix} 0 & 0.1 & 0 \\ 0.1 & 0.9 & 0.1 \\ 0 & 0.1 & 0 \end{bmatrix} \qquad B = 0 \qquad I = 0$$

This temptate converts monochromatic image into black–and–white one.

After transformation of all 3 layers (compponents), the results are putted together. This way regions of 8 colors are created.

During the experiment several aerial photos were used. Due to space limitation we restrict ourselves to consider only one photo — a roundabout in Warsaw (Figure 3). The top left image is a monochromatic version of the original color photo. Top right one presents a result of applying the network to the monochromatic image. The bottom left image was obtained with RGB model. The bottom right one was returned when HSV model was considered.

On all resulting images roads can be easily distinguished. However, when HSV model was considered, the roads differ themselves from the trees and grass. Pavements have sharp edges only when RGB model was used. The railway track is poorly seen on all three images. The original image contained houses and cars as well. Except from rare cases, almost any house cannot be distinguished on the resulting image. Cars are mostly removed or treated like noise, as they are very small.

We investigated also a graphic, where only basic colors: red, green, blue, cyan, magenta, yellow, black and white were used. When HSV model was considered cyan, yellow, red and green were not distinguished. Also, object in blue and magenta colors are not recognised.

[1] Cloning tempate is repeating structure of connections between the cells and their weights.

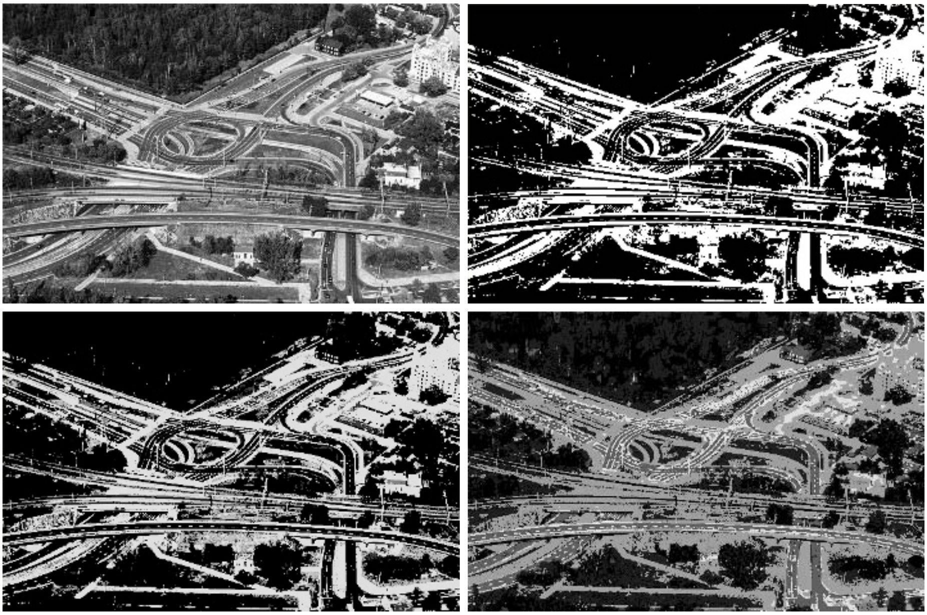


Fig. 3. Results — monochromatic version of original photo (top left), and after applying to the network (top right), result of usage RGB model (bottom left), result of usage HSV model (bottom right)

5 Conclusions

In the paper we applied DTCNN for color image segmentation and compared RGB and HSV color models.

The presented network is very simple, nevertheless the results are satisfactory. Better results could be obtained if the noise is removed. Only two values is enough for all, except the hue component. This component takes originally values from 0 to 360 and has the biggest influence on the resulting color. To achieve better results another template should be used, which enables us to have more resulting values than only two.

The results are better for HSV model, because in the real world images under consideration the colors are soft and dull. More objects can be distinguished. However, the resulting image use other colors to indicate objects, what may cause some problems for interpreting it. If an artificial image with vivid (basis) colors was given to the network, the better results were obtained for the RGB model.

In our further research we will focus on noise removing, usage of other cloning templates and classifying the selected objects.

Acknowledgements

We would like to Mr. Tomasz Stępień (`www.foto-tom.pl`) for the aerial photos used in researches.

References

1. I. Aizenberg, N. Aizenberg, J. Hiltner, C. Moraga, E. Meyer zu Bexten: Cellular neural networks and computational intelligence in medical image processing, *Image and Vision Computing* Vol. 19, No. 4, pp. 177-183, (2001).
2. P. Arena, A. Basile, M. Bucolo, L. Fortuna: Image processing for medical diagnosis using CNN, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* Vol. 497, No. 1, pp. 174-178, (2003).
3. L. Bertucco, M. Coltelli, G. Nunnari, L. Occhipinti: Cellular neural networks for real-time monitoring of volcanic activity, *Computers & Geosciences* Vol. 25, No. 2, pp. 101-117, (1999).
4. H.D. Cheng, X.H. Jiang, Y. Sun, J. Wang: Color image segmentation: advances and prospects, *Pattern Recognition*, Vol. 34, No. 12, pp. 2259-2281, (2001).
5. L.O. Chua, L. Yang: Cellular neural networks: Theory *IEEE Transaction on Circuits and Systems*, Vol. 35, pp. 1257-1272, (1988).
6. L.O. Chua, L. Yang: Cellular neural networks: Applications *IEEE Transaction on Circuits and Systems*, Vol. 35, pp. 1273-1290, (1988).
7. M. Egmont-Petersen, D. de Ridder, H. Handels: Image processing with neural networks — a review, *Pattern Recognition*, Vol. 35, No. 10, pp. 2279-2301, (2002).
8. Y. Gaobo, Z. Zhaoyang: Video object segmentation for head–shoulder sequences in the cellular neural networks architecture, *Real-Time Imaging*, Vol. 9, No. 3, pp. 171-178, (2003).
9. H. Harrer, J.A. Nossek: Discrete-time Cellular Neural Networks, *International Journal of Circuit Theory and Applications*, Vol 20, No. 5, pp. 453-467, (1992).
10. T. Kacprzak, K. Ślot: *Sieci neuronowe komórkowe*, Wyd. Naukowe PWN (1995) (in Polish).
11. R. A. Kosiński: *Sztuczne sieci neuronowe*, WNT (2002) (in Polish).
12. F. Kurugollu, B. Sankur, A. E. Harmanci: Image segmentation by relaxation using constraint satisfaction neural network, *Image and Vision Computing*, Vol. 20, No. 7, pp. 483-497, (2002).
13. M. Milanova, U. Büker: Object recognition in image sequences with cellular neural networks, *Neurocomputing* Vol. 31, No. 1, pp. 125-141, (2000).
14. M. Milanova, P.E.M. Almeida, J. Okamoto Jr., M. Godoy Simőes: Applications of cellular neural networks for shape from shading problem *Lecture Notes in Artificial Intelligence*, pp. 52-63, Springer-Verlag (1999).
15. A. Slavova: Applications of some mathematical methods in the analysis of cellular neural networks, *Journal of Computational and Applied Mathematics*, Vol. 114, No. 2, pp. 387-404, (2000).
16. D.L. Vilarino, V.M. Brea, D. Cabello, J.M. Pardo: Discrete-time CNN for image segmentation by active contours, *Pattern Recognition Letters* Vol. 19, No. 8, pp. 721-734, (1998).
17. D.L. Vilarino, D. Cabello, J.M. Pardo, V.M. Brea: Cellular neural networks and active contours: a tool for image segmentation, *Image and Vision Computing* Vol. 21, No. 2, pp. 189-204, (2003).
18. L. Wang, J. P. de Gyvez, E Sánchez-Sinencio: Time Multiplexed Color Image Processing Based on a CNN with Cell-State Outputs, *IEEE Transactions on very large csale Integration (VLSI) Systems*, Vol. 6, No. 2, (1998).

Image Segmentation Using Watershed Transform and Feed-Back Pulse Coupled Neural Network

Yiyan Xue and Simon X. Yang

University of Guelph, Guelph, Ontario, N1G 2W1, Canada

Abstract. This paper presents a novel approach for image segmentation with the fusion of morphological watershed transform(WST) and feedback pulse coupled neural network (FPCNN). FPCNN is used as a preprocessor to locate the markers in the image automatically. Controlled by markers, WST can be applied to segment the image without over-segmentation problem.

1 Introduction

Watershed transform (WST) [1] is a classical algorithm for image segmentation due to its simplicity and that it can generate close-contour objects for high level vision tasks such as object detection or recognition. However, when used alone, WST always leads to over-segmentation problem, which could even render the result useless. Marker-controlled WST can help to resolve this problem, but it is difficult to locate markers automatically. Pulse coupled neural work (PCNN) [2] is a model inspired by the study of small mammalian visual cortex. With each neuron corresponding to an image pixel, PCNN can pulse groups of pixels as outputs, based on spatial proximity and brightness similarity of the pixels in an image. This unique feature makes PCNN a useful tool for image processing tasks. However, PCNN is complex in theoretical analysis, the grouping of pixels based on similarity is ambiguous, and many parameters of PCNN need to be tuned up for satisfactory result. The complementary advantage and limitation of WST and PCNN imply a promising direction to fuse these two methods together for image segmentation. This paper presents a method to use a modified feedback PCNN model as an image pre-processor to locate the markers quickly and automatically. After the markers are located, WST is executed in a controlled manner that the over-segmentation problem can be resolved.

2 Marker Based Watershed Transformation

WST for image segmentation is based on the topographic interpretation of a grey scale image. An image can be regarded as a topographic relief on which the elevation of a point is represented by the gradient of a pixel. Assuming there is a hole that is punched in each regional minimum and water is rising through these holes, in order to prevent the merging of water, dams are built to separate

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 531–536, 2005.

adjacent regions. The process of water flooding and dams building will continue until the highest elevation level of the whole relief is reached. At the end of the flooding, watershed lines are formed to surround each local minimum and the segmentation of an image can be achieved. The implementation of above process in an image usually starts with detecting and labelling regional minima of the gradients. The pixels without labels are either assimilated into labelled regions or assigned new labels if they meet the criteria as new regional minima. It continues until a completely labelled image is generated, with each labelled region representing an image segment.

When WST is applied alone to a real image, over-segmentation problem is usually unavoidable because of noises and gradient irregularity. Morphological filters could help to alleviate but cannot resolve the problem. A good solution comes from object markers or background markers [1], which can be used to constrain the amount of catchment basins before WST is applied. However, it is a difficult task to locate the markers automatically. In most cases, markers are selected by man-machine interaction. In this paper, we present a method to use FPCNN to find the makers automatically, which is demonstrated to be a simple and general solution.

3 Use FPCNN to Find the Object Markers

The original PCNN model was developed by Eckhorn et al. [2] based on the study of cat's visual cortex. Linbald and Kinser [3] refined the PCNN model in discrete form for computer simulation. In PCNN, each neuron is characterized by five equations that are updated iteratively to produce a series of binary outputs, which constitute binary images in an image processing application. In order to acquire object markers for real-time processing, we use a modified feedback PCNN (FPCNN) model as illustrated in Fig. 1. There are three parts in the model: the receptive fields, the modulation product and the pulse generator. The major difference of this model from the original PCNN model is that the feeding input is replaced by the input stimulus and the output feeds back to modify the input.

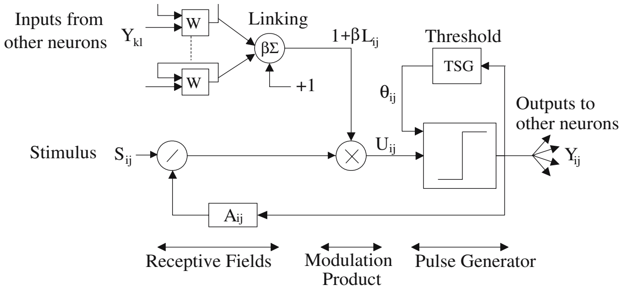


Fig. 1. Architecture of the modified FPCNN

In the receptive field, the linking input is denoted in discrete form as,

$$L_{ij}[n] = e^{-\alpha_L} L_{ij}[n-1] + V_L \sum_{kl} W_{ijkl} Y_{kl}[n-1] \tag{1}$$

where $L_{ij}[n]$ is the linking input of the ijth neuron in a 2D array of neurons, n is the iteration number, the memory of previous state $L_{ij}[n-1]$ decays with coefficient α_L; Y_{kl}'s are the outputs of neighouring neurons, W_{ijkl}'s represent the synaptic weights that are usually defined as local Gaussian distribution. The constant V_L is used to scale the correlation of the neighbouring neurons.

The modulation product of the input stimulus $S_{ij}[n]$ and linking input $L_{ij}[n]$ results in the total neuron internal activity $U_{ij}[n]$, which is calculated by,

$$U_{ij}[n] = S_{ij}[n](1 + \beta L_{ij}[n]) \tag{2}$$

where β is the linking strength that controls the combination.The feeding input in the original PCNN model is replaced by the input stimulus in this equation because the leaky integrator has minor impact to the internal activity, comparing to the input $S_{ij}[n]$ that is iteratively updated by the neuron output in this FPCNN model.

The pulse generator compares the internal activity with the threshold to determine if the output shall fire pulse (one) or not (zero), i.e.,

$$Y_{ij}[n] = \begin{cases} 1 & \text{if } U_{ij}[n] > \theta_{ij}[n]; \\ 0 & \text{otherwise .} \end{cases} \tag{3}$$

Because of the output feedback, the threshold will be boosted to a high value if the neuron fires, otherwise it decays until the neuron fires again. This dynamic change is defined by,

$$\theta_{ij}[n] = e^{-\alpha_\theta} \theta_{ij}[n-1] + V_\theta Y_{ij}[n-1] \tag{4}$$

where α_θ is the decaying coefficient and V_θ is a large constant in order to make sure the threshold can be increased to a higher value than the average magnitude of internal activity. Otherwise the pulsing process may stop after a few iterations.

Kinser and Johnson [4] developed the feedback concept for PCNN based on the investigation of the rat's olfactory bulb. The rat broadly classifies the smell at first and then the feedback transforms the coarse signal to finer signal for further classification. The feedback signal $A_{ij}[n]$ describes the modification of input $S_{ij}[n]$ by the feedback signal from the previous iteration, which is given as,

$$A_{ij}[n] = e^{-\alpha_A} A_{ij}[n-1] + V_A Y_{ij}[n-1] \tag{5}$$

$$S_{ij}[n] = S_{ij}[n-1]/A_{ij}[n-1] \tag{6}$$

where α_A is the decaying constant effecting on the memory of previous state of A_{ij},V_A is the adjusting constant.

Because of the local interconnections among the neurons, a neuron would stimulate its neighbours to fire when it fires. If a group of neurons have similar inputs, one neuron can trigger the whole group to fire together. The edges have different neighbouring characteristics than the interior of segments, therefore the edges tend to fire at different time. The group pulsing effect forms the base of PCNN's segmenting ability. However, according to [5], there are limitations to achieve good segmentation purely by PCNN. To overcome those limitations and to avoid the complexity to tune up PCNN parameters, we consider to use PCNN to do coarse segmentation only for markers which can have arbitrary shapes as long as they are enclosed in segments. The markers are to be used by following WST algorithm to accomplish the final segmentation. By using FPCNN, the group pulsing can be accelerated. Since the output signal feeds back to the input and the threshold in an opposite direction, the process of group pulsing could propagate through the whole image in a few iterations. In sense of performance, FPCNN model suits this application better than the general PCNN model.

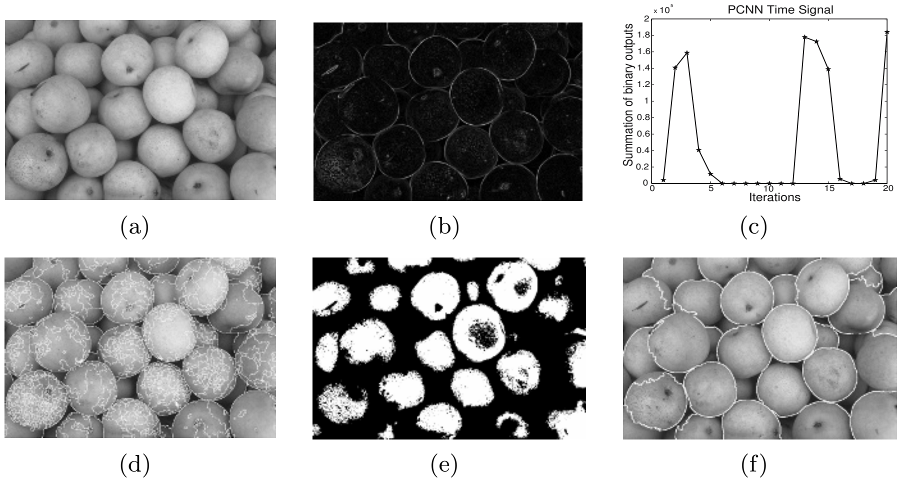
The equations (1)-(6) are computed iteratively, resulting in a series of binary images as outputs. Each binary image contains blobs as the results of group pulsing. These blobs actually segment the image coarsely, therefore they could be used as markers for further processing by WST. However, since there is more than one binary image output from FPCNN, which one(s) should be used? At this stage, we can consider to use the time signals of FPCNN to make selections, which is given by

$$G[n] = \sum_{i,j} Y_{ij}[n]. \tag{7}$$

As a function of time, the summation of all the binary outputs produces the time signals to describe the propagation of pulsing waves through the image. With a few empirical rules based on the time signals, the useful binary image(s) could be selected from a number of outputs. The desired binary image is usually acquired from the iteration that produces the second highest magnitude of time signal. This can be explained as following: when all the segments fires, based on Eq. (7), the highest magnitude of time signal is acquired, which also means that blobs may just overlap the segments. Since the markers for WST are expected to be disconnected and contained within the segments, the binary image corresponding to the second highest magnitude usually indicate part of each segment fires, therefore it could be used to locate the markers. This gives a general rule to identify the useful FPCNN binary image for WST. Actually, since every binary image contains more or less some markers. Multiple WST could also be performed on the basis of a few FPCNN binary images. The results of multiple WST can be overlapped for a complete segmentation.

4 Experiment Results

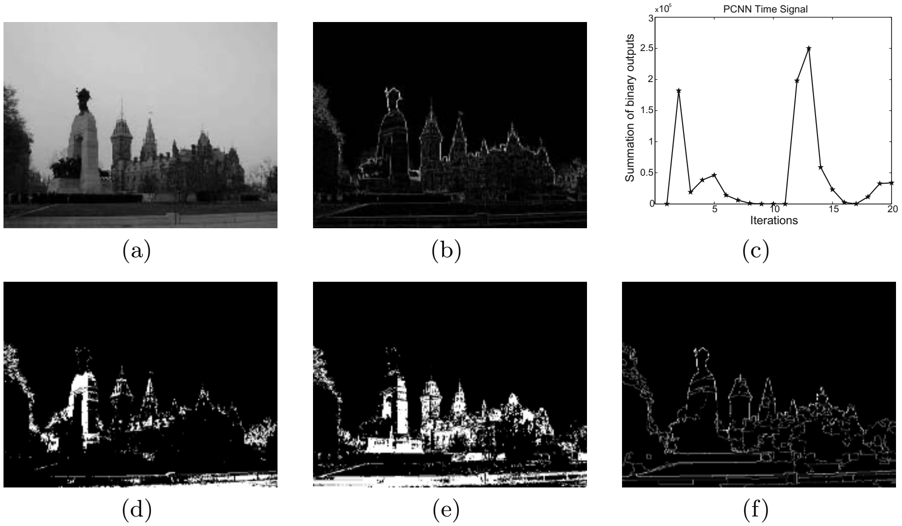
In order to demonstrate the generality of proposed algorithm, the tests on two different types of images are presented in Fig. 2 and Fig. 3. Since the test image Fig. 2(a) is comparatively simple, only one FPCNN binary image is selected according to the time signal. Correspondingly, WST is executed only once. For



(a) (b) (c)

(d) (e) (f)

Fig. 2. image segmentation on 'pears.png' using proposed algorithms. (a) Test image; (b) Gradient image; (c) FPCNN time signals; (d) Over-segmentation caused by direct WST without markers, but with threshold filter; (e) Object markers generated by FPCNN binary output from the 2nd iteration; (f) Overlapping of watershed lines and the original image.



(a) (b) (c)

(d) (e) (f)

Fig. 3. image segmentation on 'scene.jpg' using proposed algorithms. (a) Test image; (b) Gradient image; (c) FPCNN time signals; (d) FPCNN binary image from the 3rd iteration; (e) FPCNN binary image from the 4th iteration; (f) The watershed lines generated by two WST outputs.

a more complicated image Fig. 3(a), two FPCNN binary images are used and the WST was applied twice. In both cases, the iteration of FPCNN were set to 20 times in order to demonstrate the time signals. The FPCNN parameters are set to be the same in both tests, they are: $\beta = 0.2, \alpha_L = 0.8, \alpha_A = 0.15, \alpha_\theta = 0.06, V_L = 1, V_\theta = 5, V_A = 1$. Since the FPCNN is only used as per-processor for object markers, the final segmentation result is not sensitive to these FPCNN parameters.

The over-segmentation problem is demonstrated by Fig. 2(d), which is the output of direct WST applying to the gradient image Fig. 2(b). Even with a threshold filter, the result is still unacceptable. According to the time signals shown in Fig. 2(c), the binary image Fig. 2(e) generated from the second FPCNN iteration is selected according to the rule mentioned afore. It is obvious the markers are well formed in this binary image. Based on the gradient image Fig. 2(b) and the markers in Fig. 2(e), WST is performed for segmentation. The overlapping of segmentation result and the original image is demonstrated in Fig. 2(f).

For a more complex image, a few binary images could be selected according to the FPCNN time signals, and multiple WST can be applied. In Fig. 3(b), a few initial iterations of FPCNN produce two waves on the time signals. The 3rd and 4th binary outputs are corresponding to the second highest amplitudes of these two waves. It denotes that the markers could be well formed from the 3rd and 4th binary images shown by Fig. 3(d) and Fig. 3(e) respectively. Based on the gradient image Fig. 3(b) and the markers, WST is performed twice to generate two sets of watershed lines, which can be overlapped for a complete segmentation, as illustrated by Fig. 3(f).

5 Summary

The feedback PCNN model can be used as a pre-processor to find markers in an image. With the markers located automatically by FPCNN, watershed transform can be applied in a controlled manner to generate closed-contour segmentations, and the over-segmentation problem of WST can be resolved.

References

1. Beucher, S., Lantuejoul, C.: Use of watersheds in contour detection. In: Int. Workshop Image Processing, Real-time Edge and Motion, Det./Est., vennes, France. (1979)
2. R. Eckhorn, H.J. Reitboeck, M.A., Dicke, P.: Feature linking via synchronization among distributed assemblies: simulation of results from cat cortex. Neural Computing **2** (1990) 293–307
3. Lindbald, T., Kinser, J.M.: Image processing using pulse-coupled neural networks. Springer (1998)
4. Kinser, J.M., Johnson, J.L.: Stabilized input with a feedback pulse-coupled neural network. Opt. Eng. **35** (1996) 2158–2161
5. Kuntimad, G., Ranganath, H.: Perfect image segmentation using pulse coupled neural networks. IEEE Trans. Neural Networks **10** (1999) 591–598

Adaptive Switching Median Filter with Neural Network Impulse Detection Step

Pavel S. Zvonarev, Ilia V. Apalkov, Vladimir V. Khryashchev, and Andrey L. Priorov

Digital Circuits and Signals Laboratory,
Department of Physics,
Yaroslavl State University,
150000 Russia, Yaroslavl, Sovetskaya st., 14
dcslab@uniyar.ac.ru

Abstract. A new neural network adaptive switching median (NASM) filter is proposed to remove salt-and-pepper impulse noise from corrupted image. The algorithm is developed by combining advantages of the known median-type filters with impulse detection scheme and the neural network was included into impulse detection step to improve its characteristics. Comparison of the given method with traditional filters is provided.

1 Introduction

Images are often corrupted by impulse noise due to errors generated in noisy sensors or communications channels. It's important to eliminate noise in images before edge detection, image segmentation or object recognition procedures. The well known median filter and its derivatives have been recognized as effective means of impulse noise removal [1-3]. The success of median filters is based on two main properties: edge preservation and efficient noise attenuation with robustness against impulsive-type noise. Edge preservation is essential in image processing due to the nature of visual perception [4].

However, median filtering also removes very fine details and sometimes changes signal structure. The main reason is that the median filter uses only rank-order information of the input data within the filter window, and discards its original temporal-order information. To avoid the damage of "good" image pixels the switching scheme is introduced [5]. The idea of this median filter modification is based on impulse noise detection. If the impulses can be detected and their positions are correctly located in the image, it is feasible to replace the impulses by the best estimates using only uncorrupted pixels. Self-organizing neural networks [6], fuzzy techniques [7] or other methods can be used on the detection step.

In the work [8] authors present a median-based switching filter, which is called progressive switching median (PSM), where both the impulse detector and the noise filter are applied progressively in iterative form. The main advantage of such method is that some impulse pixels located in the middle of large noise blotches can also be properly detected and filtered. Another interesting approach for impulse noise removal is adaptive median (AM) filter [9]. It has variable window size for removal of impulses while preserving sharpness. Simulations on test images with PSM and

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 537 – 542, 2005.

AM filters confirm that these algorithms are superior to standard median filters in removing impulse noise.

The new based-median neural network adaptive switching (NASM) filter is introduced in this paper. It uses advantages of filters considered above. The neural network was included into impulse detection step to improve algorithm characteristics.

The main tasks of this work are the development and testing of complex NASM algorithm and its comparison with different median type filters modifications with impulse detection scheme. The paper is organized as follows: in Section 2 we describe the proposed NASM algorithm. Section 3 shows the simulation results. Conclusions and directions for future work are given in Section 4.

2 Impulse Noise Detection and Removing

The noise considered in this work is bipolar salt-and-pepper impulsive noise which means fixed values 0 (pepper) and 255 (salt) for all the impulses. This model is mathematically expressed as

$$x_i = \begin{cases} 0 & \textit{with probability } p_n \\ 255 & \textit{with probability } p_p \\ \varphi_i & \textit{with probability } 1-(p_n+p_p) \end{cases} \tag{1}$$

where $p_n = p_p = 0{,}5R$, φ_i denotes the uncorrupted (good) pixel values, 0 – fixed value of the negative impulses, 255 – fixed value of positive impulses, R – noise ratio$(0 \le R \le 1)$ and x_i denotes the pixel values of the degraded image.

Impulse detection procedure includes two steps: the first step is the preliminary impulse detection and the second step is the neural network impulse detection which is used to correct the preliminary result. The preliminary impulse detector can find almost all impulses for salt-and-pepper noise model, but some "good" pixels with values equal salt or pepper could be detected as impulses too. Network allows distinguishing such pixels from impulses and it is used to obtain final result of impulse detection.

Preliminary impulse detection algorithm uses two images. The first represents corrupted image $\{x_i\}$, which displays values of pixel at position $\boldsymbol{i} = (i_1, i_2)$. The second is a binary flag image $\{f_i\}$, where the binary value f_i is used to indicate whether the pixel $\boldsymbol{i}$ has been detected as an impulse, i.e., $f_i = 0$ means the pixel $\boldsymbol{i}$ is good and $f_i = 1$ means it is an impulse. In the beginning, we assume that all the image pixels are good, i.e., $f_i \equiv 0$.

Then for each pixel x_i we find the minimum and maximum values of the samples in a $W_D \times W_D$ window (W_D is an odd integer not smaller than 3). If we use Ω_i^W to represent the set of the pixels within a $W \times W$ window centered about $\boldsymbol{i}$

$$\begin{aligned} \Omega_i^W = \{\boldsymbol{j} = (j_1, j_2) \big| & i_1 - (W-1)/2 \le j_1 \le j_1 + (W-1)/2, \\ & i_2 - (W-1)/2 \le j_2 \le j_2 + (W-1)/2\} \end{aligned} \tag{2}$$

then we have

$$\begin{aligned} \min_i &= \min\{x_i \mid \boldsymbol{j} \in \Omega_i^{W_D}\}, \\ \max_i &= \max\{x_i \mid \boldsymbol{j} \in \Omega_i^{W_D}\}. \end{aligned} \tag{3}$$

After this we use simple measurement to detect impulses

$$f_i = \begin{cases} 0, & \text{if } \min_i < x_i < \max_i \\ 1, & \text{else.} \end{cases} \tag{4}$$

The received binary flag image $\{f_i\}$ is the result of preliminary detection step.

Neural network impulse detection algorithm uses three images. The first represents corrupted image $\{y_i\}$. The second is a binary flag image $\{f_i\}$ obtained by preliminary impulse detection. And the third is also a binary flag image $\{g_i\}$ which is used to write the final impulse detection result. At start we assume $\{g_i\}$ equal preliminary impulse detection result, i.e., $g_i \equiv f_i$.

For each pixel detected as an impulse by preliminary detection we apply a network.

There are two premises for network topology selection. The first is the size of input vector and the second is the size of output vector. During examination of training data we found that the most information consists in seven local characteristics of pixel. They are pixel value, medians and dispersions for different neighborhoods of estimated pixel. Here we use only pixels with $f_i = 0$ for calculating of such values. Let M^W denote the number of the pixels with $f_i = 0$ in the $W \times W$ window. If M^W is even, the median calculates as arithmetic mean of two middle elements of sorted data. Then, if M^3 more then 0, we preset elements of input network vector v

$$\begin{gathered} v_0 = y_i \\ v_1 = \mathrm{Med}\{y_j \mid f_j = 0, \boldsymbol{j} \in \Omega_i^3\} - y_i \qquad v_2 = \mathrm{Disp}\{y_j \mid f_j = 0, \boldsymbol{j} \in \Omega_i^3\} - y_i \\ v_3 = \mathrm{Med}\{y_j \mid f_j = 0, \boldsymbol{j} \in \Omega_i^5\} - y_i \qquad v_4 = \mathrm{Disp}\{y_j \mid f_j = 0, \boldsymbol{j} \in \Omega_i^5\} - y_i \\ v_5 = \mathrm{Med}\{y_j \mid f_j = 0, \boldsymbol{j} \in \Omega_i^7\} - y_i \qquad v_6 = \mathrm{Disp}\{y_j \mid f_j = 0, \boldsymbol{j} \in \Omega_i^7\} - y_i\,. \end{gathered} \tag{5}$$

The dimension of output vector in compliance with current task was selected to be equal to one in the case of two different output states (“good” and “bad”). In our algorithm three-layer network with S_D neurons in hidden layer is used. During experimental work we found out that S_D can’t be less than five.

Let D_i denote the output value of network with range [0;1] for pixel at position $\boldsymbol{i}$, where D_i approaches to 1 confirms that the pixel was detected as an impulse correctly and if D_i approaches to 0 means that the pixel with high probability is “good”. Then we use simple measurement to pass corrected solution about pixel

$$g_i = \begin{cases} 0, & \text{if } f_i = 1;\ D_i < 0{,}1;\ M^3 > 0 \\ g_i, & \text{else.} \end{cases} \tag{6}$$

The received binary flag image $\{g_i\}$ is the result of impulse detection procedure.

After this we use the filtration procedure of PSM algorithm which is described in work [8] with size of window equal 3.

3 Simulation Results

In experiments, the original test images are corrupted with fixed valued salt-and-pepper where negative and positive values are 0 and 255 respectively with equal probability. Mean square error (MSE) is used to evaluate the restoration performance in our experiments. MSE is defined as

$$MSE = \frac{1}{N} \sum_{i} (u_i - \varphi_i)^2 \tag{7}$$

where N is the total number of pixels in the image, u_i and φ_i are the pixel values at position $\boldsymbol{i}$ in the test and the original images respectively.

To implement NASM filter we need to define three parameters: W_D, S_D and W_F. They are not sensitive to noise ratio and the best results for the most of the test images were obtained with $W_D = 7$, $S_D = 5$ and $W_F = 3$.

Average representation of neural network influence on the algorithm performance for the set of test images is shown on Fig. 1, where ASM and NASM algorithms with switched off and switched on neural network detection step respectively.

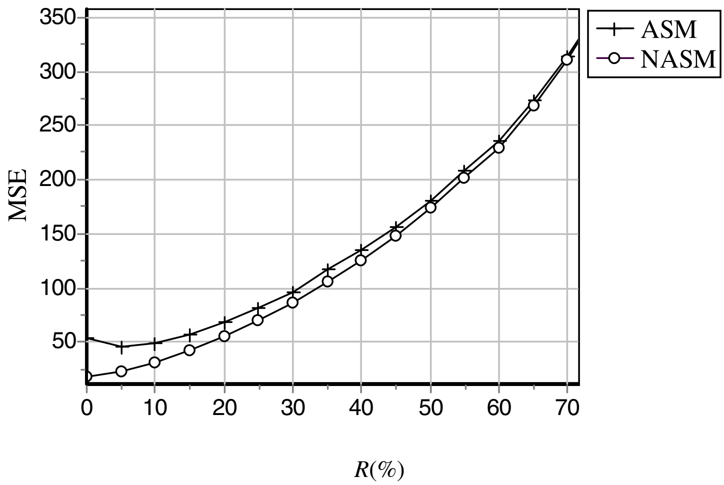


Fig. 1. Representation of the positive influence of neural network on the algorithm performance

A comprehensive evaluation is reported in Fig. 2 and Fig. 3 that compare MSE for two images with different detail degree corresponding to six different algorithms: 1) simple median filter with window size 3×3, 2) AM filter for window size from 3 to 15, 3) PSM filter, 4) iterative median (IM) filter (iterative apply the simple median filter) with 3×3 window and 10 iterations, 5) center weighted median filter (CWM) with window size 5×5 and a center weight of 3 [10], 6) proposed NASM filter.

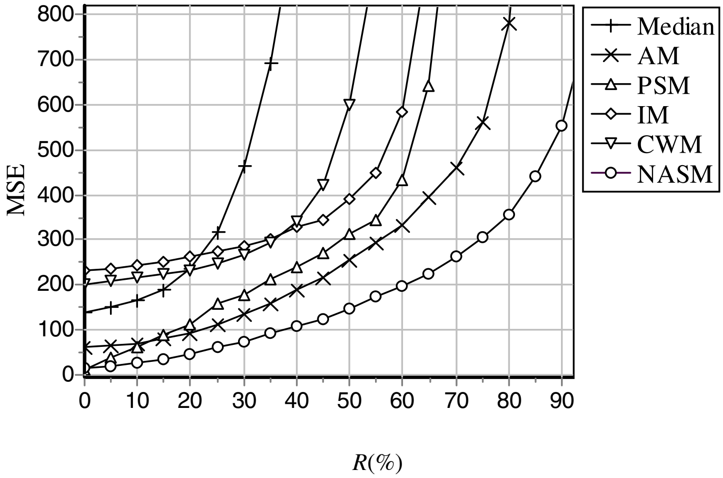


Fig. 2. A comparison of different median-based noise removal algorithms for the test image "Stream and Bridge"

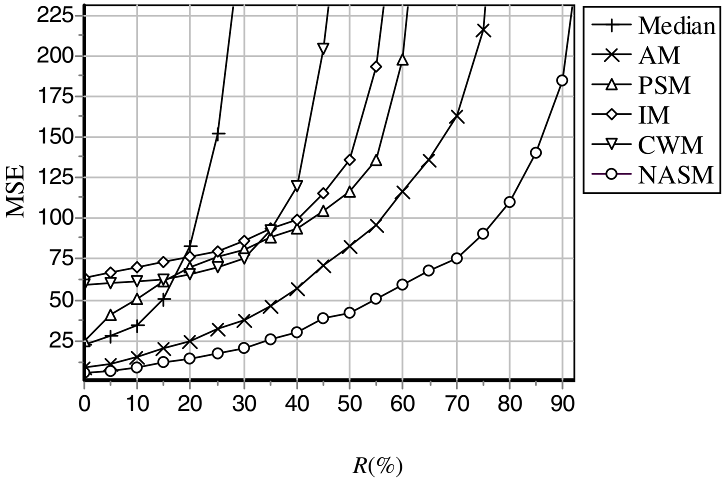


Fig. 3. A comparison of different median-based noise removal algorithms for the test image "Peppers".

The MSE curves demonstrate that in the presence of salt-and-pepper type of noise NASM algorithm is better than the other median-based methods on both images on all range of noise ratio. The algorithm was tested on many others images and the similar results were obtained.

4 Conclusion

The idea of new impulse noise removal algorithm has arisen on the basis of analysis of two known effective algorithms PSM and AM. The basic problem of PSM algorithm is the filtration of highly corrupted images by the salt-and-pepper noise. In this case noisy pixels are grouped in blocks. PSM algorithm is unable to remove them. AM filter cannot distinguish "good" pixels of image with values equal to salt or pepper values from impulses, that is why AM filter defects borders of objects in image. The algorithm submitted in this work includes advantages of the filters considered above, eliminating their basic lacks. Also algorithm was improved with a help of neural network. Neural network positive effect appears especially when original image corrupted with salt-and-pepper noise has "good" pixels with values equal to salt or pepper values. Network provides the possibility to distinguish such pixels from impulses. Proposed algorithm demonstrates high results on overwhelming majority of test images. It removes the most of noise pixels while preserving details and edges of the objects even in highly corrupted images. This property is important for further processing (edges detection or objects recognition).

References

1. I. Pitas and A. Venetsanopoulos, Nonlinear Digital Filters: Principles and Applications. Boston, MA: Kluwer, 1990.
2. S. Mitra and G. Sicuranza, Nonlinear Image Processing. Academic Press, 2000.
3. T. Nodes and N. Gallagher, "Median Filters: Some Modifications and Their Properties," IEEE Trans. ASSP, vol. ASSP-30, no. 5, 1982.
4. L. Yin, R. Yang, M. Gabbouj, Y. Neuvo, "Weighted Median Filters: A Tutorial," IEEE Trans. Circuits Systems, vol. 43. no. 3, pp. 157-192, 1996.
5. T. Sun and Y. Neuvo, "Detail-Preserving Median Based Filters in Image Processing," Pattern Recognit. Lett., vol. 15, pp. 341-347, 1994.
6. H. Kong and L. Guan, "A Neural Network Adaptive Filter for the Removal of Impulse Noise in Digital Images," Neural Networks, vol. 9, no. 3, pp.373-378, 1996.
7. D. Zhang and Z. Wang, "Impulse Noise Detection and Removal Using Fuzzy Techniques," Electron. Lett., vol. 33, pp. 378-379, 1997.
8. Z. Wang and D. Zhang, "Progressive Switching Median Filter for the Removal of Impulse Noise from Highly Corrupted Images," IEEE Trans. Circuits Systems – II, vol. 46, no. 1, pp. 78-80, 1999.
9. H. Hwang and R. Haddad, "Adaptive Median Filters: New Algorithms and Results," IEEE Trans. on Image Processing, vol. 4, no. 4, pp. 499-502, 1995.
10. S. Ko and Y. Lee, "Center Weighted Median Filters and Their Applications to Image Enhancement," IEEE Trans. Circuits Systems, vol. 38, no. 9, pp. 984-993, 1991.

Human Face Detection Using New High Speed Modular Neural Networks

Hazem M. El-Bakry

University of Aizu, Aizu Wakamatsu, Japan 965-8580
d8071106@u-aizu.ac.jp

Abstract. In this paper, a new approach to reduce the computation time taken by neural networks for the searching process is introduced. Both fast and cooperative modular neural networks are combined to enhance the performance of the detection process. Such approach is applied to identify human faces automatically in cluttered scenes. In the detection phase, neural networks are used to test whether a window of 20x20 pixels contains a face or not. The major difficulty in the learning process comes from the large database required for face / nonface images. A simple design for cooperative modular neural networks is presented to solve this problem by dividing these data into three groups. Such division results in reduction of computational complexity and thus decreasing the time and memory needed during the test of an image. Simulation results for the proposed algorithm on Bio database show a good performance.

1 Introduction

The goal of this paper is to solve the problem of requiring large database to build an automatic system in order to detect the location of faces in scenes. This paper explores the use of modular neural network (MNN) classifiers. Non-modular classifiers tend to introduce high internal interference because of the strong coupling among their hidden layer weights [3,5]. As a result of this, slow learning or over fitting can occur during the learning process. Sometimes, the network could not be learned for complex tasks. Such tasks tend to introduce a wide range of overlap which, in turn, causes a wide range of deviations from efficient learning in the different regions of input space [3,5]. High coupling among hidden nodes will then, result in over and under learning at different regions [8]. Enlarging the network, increasing the number and quality of training samples, and techniques for avoiding local minina, will not stretch the learning capabilities of the NN classifier beyond a certain limit as long as hidden nodes are tightly coupled, and hence cross talking during learning [7]. A MNN classifier attempts to reduce the effect of these problems via a divide and conquer approach. It, generally, decomposes the large size / high complexity task into several sub-tasks, each one is handled by a simple, fast, and efficient module. Then, sub-solutions are integrated via a multi-module decision-making strategy. Hence, MNN classifiers, generally, proved to be more efficient than non-modular alternatives [2,5,6]. In section 2, a method for detection of human faces in photo images is presented. Also, an algorithm during the searching procedure is described. A fast searching algorithm for face detection which reduces the computational complexity of neural networks is presented in section 3.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 543 – 550, 2005.

2 Human Face Detection Based on Neural Networks

The human face is a complex pattern. Finding human faces automatically in a scene is a difficult yet significant problem. It is the first step in fully automatic human face recognition system. Face detection is the fundamental step before the face recognition or identification procedure. Its reliability and time response have a major influence on the performance and usability of the whole face recognition system. Training a neural network for the face detection task is challenging because of the difficulty in characterizing prototypical "nonface" images [1]. Unlike face recognition, in which the classes to be discriminated are different faces, the two classes to be discriminated in the face detection are "image containing faces" and "image not containing faces". It is easy to get a representative sample of images that contain faces, but much harder to get a representative sample of those which do not. Feature information needs to be stored in the database for the purpose of retrieval. Information retrieval can be done by using a neural network approach which has the potential to embody both numerical and structural face data. However, neural network approaches have been demonstrated only on limited database. The use of huge samples of face/nonface images makes the learning process very difficult for the neural network.

2.1 A Proposed Algorithm for Face Detection Using MNNs

First, in an attempt to equalize the intensity values of the face image, the image histogram is equalized. This not only reduce the variability of generated by illumination conditions, and enhance the image contrast but also increases the number of correct pixels that can be actually encountered [3]. The next component of our system is a classifier that receives an input of 20x20 pixel region of gray scale image and generates an output region ranging from 1 to -1, signifying the presence or absence of a face, respectively. This classification must have some invariance to position, rotation, and scale. To detect faces anywhere in the input, the classifier is applied at every location in the image. To detect faces larger than the window size, the input image is repeatedly reduced in size. The classifier is applied at every pixel position in the image and scale the image down by a factor of 1.2 for each step as shown in Fig. 1. So, the classifier is invariant to translation and scaling. To have rotation invariant, the neural network is trained for images rotated from 0° to 345° by a step of 15°. In order to train neural networks used in this stage, a large number of face and nonface images are needed. A sample of nonface images, which are collected from the world wide web, is shown in Fig. 2. So, conventional neural networks are not capable of realizing such a searching problem. As a result of this, MNNs are used for detecting the presence or absence of human faces for a given image. Images (face and nonface) in the database are divided into three groups which result in three neural networks. More divisions can occur without any restrictions in case of adding more samples to the database. Each group consists of 600 patterns (300 for face and 300 for nonface). Each group is used to train one neural network. Each network consists of hidden layer containing 30 neurons, and an output layer which contains only one neuron. Here, two models of MNNs are used. The first is the ensemble majority voting which gives a result of 82% detection rate for the Bio-Database. The other is the average voting which gives a better result of 89% detection rate.

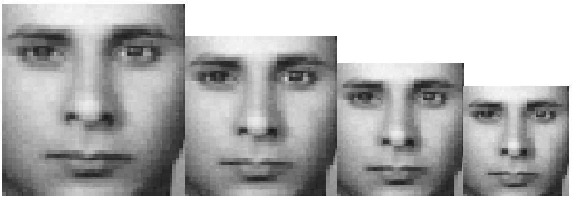


Fig. 1. Image resizing by a factor of 1.2 during face detection

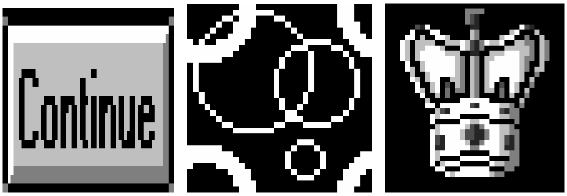


Fig. 2. Examples of nonface images

2.2 Enhancement of Recognition Performance

To enhance the detection decision, the detection results of neighboring windows can be used to confirm the decision at a given location. This will reduce false detection as neighboring windows may reveal the nonface characteristics of the data. For each location the number of detections within a specified neighborhood of that location can be counted. If the number is above a threshold, then that location is classified as a face. Among a number of windows, the location with the higher number of detections in range of one pixel is preserved, and locations with fewer detections are eliminated. In our case, a threshold of 4 is chosen. Such strategy improves the detection rate for the Bio-database to 96% (average voting), as a result of reducing the false detections. It is clear that, the use of MNNs and this enhancement has improved the performance over our previous results in [9], where non-modular neural networks are used, in which the best result on the same samples was 61% [9].

3 Fast Neural Networks for Face Detection

In subsection 2.1, modular neural network for object detection is presented using a sliding window to test a given input image. In this section, a fast algorithm for object detection (used with each of the neural networks presented in section 2.1) based on two dimensional cross correlations that take place between the tested image and the sliding window. Such window is represented by the neural network weights situated between the input unit and the hidden layer. The convolution theorem in mathematical analysis says that a convolution of f with h is identical to the result of the following steps: let F and H be the results of the Fourier transformation of f and h in the frequency domain. Multiply F and H in the frequency domain point by point and then transform this product into spatial domain via the inverse Fourier transform. As a result, these cross correlations can be represented by a product in the frequency domain. Thus, by using cross correlation in the frequency domain a speed up in an order of magnitude can be achieved during the detection process [1-3]. In the detection phase, a sub image I of size mxn (sliding window) is extracted from the tested image, which has a size PxT, and fed to the neural network. Let X_i be the vector of weights between the input sub image and the hidden layer. This vector has a size of mxn and can be represented as mxn matrix. The output of hidden neurons h(i) can be calculated as follows:

$$h_i = g\left(\sum_{j=1}^{m} \sum_{k=1}^{n} X_i(j,k)I(j,k) + b_i \right) \quad (1)$$

where g is the activation function and b(i) is the bias of each hidden neuron (i). Eq. 1 represents the output of each hidden neuron for a particular sub-image I. It can be obtained to the whole image Z as follows:

$$h_i(u,v) = g\left(\sum_{j=-m/2}^{m/2} \sum_{k=-n/2}^{n/2} X_i(j,k)\ Z(u+j, v+k) + b_i \right) \quad (2)$$

Eq. 2 represents a cross correlation operation. Given any two functions f and d, their cross correlation can be obtained by:

$$f(x,y) \otimes d(x,y) = \left(\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(x+m, y+n)d(m,n) \right) \quad (3)$$

Therefore, Eq. 2 may be written as follows:

$$h_i = g(Z \otimes X_i + b_i) \quad (4)$$

where h_i is the output of the hidden neuron (i) and h_i (u,v) is the activity of the hidden unit (i) when the sliding window is located at position (u,v) and (u,v) $\in$ [P-m+1,T-n+1]. Now, the above given cross correlation can be expressed in terms of Fourier Transform:

$$Z \otimes X_i = F^{-1}\left(F(Z) \bullet F^*(X_i)\right) \quad (5)$$

Hence, by evaluating this cross correlation, a speed up ratio can be obtained compared to conventional neural networks. Also, the final output of the neural network can be evaluated as follows:

$$O(u,v) = g\left(\sum_{i=1}^{q} w_o(i)\, h_i(u,v) + b_o \right) \quad (6)$$

O(u,v) is the output of the neural network when the sliding window located at the position (u,v) in the input image Z.

For a tested image of NxN pixels, the 2D-FFT requires ($5N^2\log_2N^2$) real computation steps [11]. The same number of computation steps is required for the weight matrix at each neuron in the hidden layer. The inverse 2D-FFT of the resulted dot product must be computed at each neuron in the hidden layer. As a result, q backward and (q+1) forward transforms have to be computed. Therefore, for a tested image, the total number of the 2D-FFT to compute is ($(2q+1)(5N^2\log_2N^2)$). Moreover, the input image and the weights should be multiplied in the frequency domain. Therefore, real computation steps of ($6qN^2$) should be added. Finally, a total of $O((q+1)(5N^2\log_2N^2)+6qN^2)$ computation steps must be evaluated for fast the neural algorithm. , for the weight matrix to have the same size as the input image, a number of zeros = (N^2-n^2) must be added to the weight matrix. This requires a total real number of real computation steps = q(N^2-n^2) for all neurons. Moreover, after computing the 2D-FFT for the weight matrix, the conjugate of this matrix must be obtained. So, a real number of computation steps =qN^2 should be added in order to obtain the conjugate of the weight matrix for all neurons. Also, a number of real computation steps equal to N is required to create butterflies complex numbers ($e^{-jk(2\Pi n/N)}$), where 0<K<L. These (N/2) complex numbers are multiplied by the elements of the input image or by previous complex numbers during the computation of 2D-FFT. To create a complex number requires two real floating point operations. Thus, the total number of computation steps required for fast neural networks becomes [11]:

$$\sigma=((2q+1)(5N^2\log_2 N^2)+q(8N^2-n^2)+N) \qquad (7)$$

Using sliding window of size nxn, for the same image of NxN pixels, (q(2n^2-1)(N-n+1)^2) computation steps are required when using traditional neural networks for the face detection process. The theoretical speed up factor η can be evaluated as follows:

$$\eta=\frac{q(2n^2-1)(N-n+1)^2}{(2q+1)(5N^2\log_2 N^2)+q(8N^2-n^2)+N} \qquad (8)$$

The speed up factor introduced in [10] for object detection which is given by:

$$K=\frac{qn^2}{(q+1)\log^2 N} \qquad (9)$$

is not correct for the reasons given in [11-23]. The relation between the image size and speed up ratio is shown in Fig. 3. Practical speed up ratio is shown in Fig.4 using MATLAB ver 5.3 and 700 MHz processor. A comparison between the classic and fast neural networks for different window size is illustrated in Fig. 5.

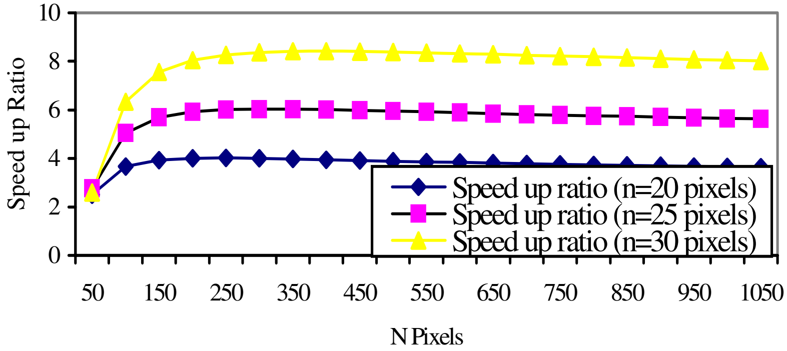


Fig. 3. The relation between the size of the input image and the speed up ratio

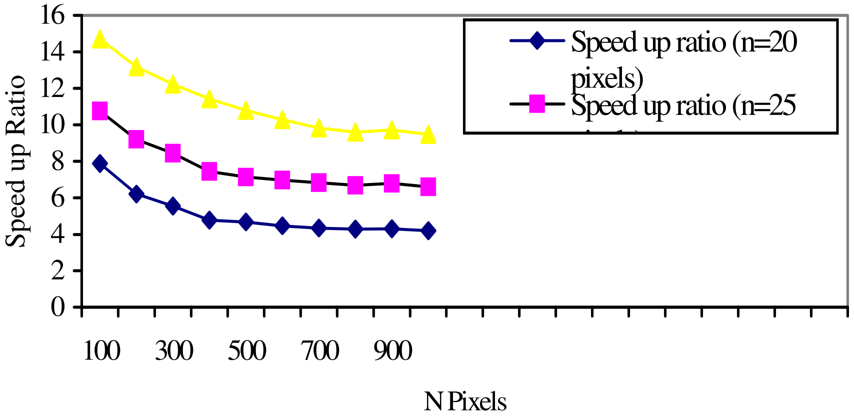


Fig. 4. Practical speed up ratio for images with different size

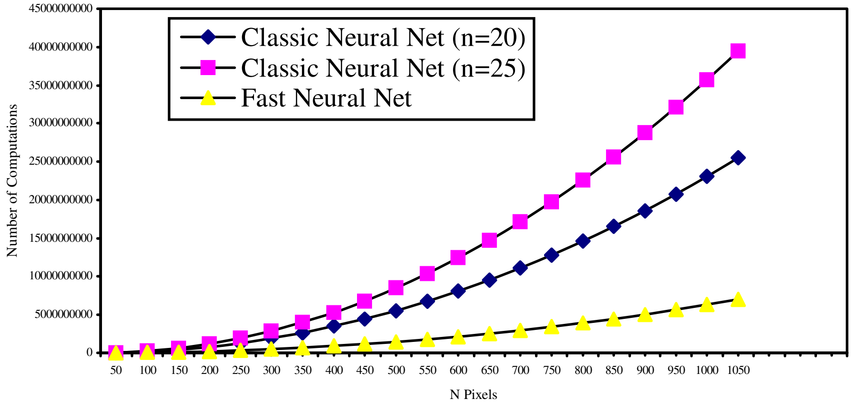


Fig. 5. A comparison between the number of computations taken by classic and fast neural networks for face detection

4 Conclusion

A fast modular neural network approach has been introduced to identify frontal views of human faces. Such approach can manipulate gray scale images of resolution 20x20 up to 500x500 pixels. The technical problem associated with large database (face/nonface) required for training neural networks has been solved using MNNs. A simple algorithm for fast face detection based on neural network and FFT is presented in order to speed up the execution time. Simulation results on Bio-database have shown that our algorithm is an efficient method for finding locations of faces when the size of the face is unknown as well as mirrored, noised, and occluded faces are detected correctly.

References

1. H. A. Rowley, S. Baluja, and T. Kanade, " Neural Network - Based Face Detection, " IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 20, No. 1, (1998) 23-38
2. R. Jacobs, M. Jordan, and A. Barto, " Task Decomposition Through Competition in a Modular Connectionist Architecture: The what and where vision tasks, " Neural Computation 3, (1991) 79-87
3. H. M. El-Bakry, "Automatic Human Face Recognition Using Modular Neural Networks," Machine Graphics & Vision Journal (MG&V), vol. 10, No. 1, (2001) pp. 47-73
4. R. Klette, and Zamperon, "Handbook of image processing operators, " John Wiley & Sonsltd, (1996)
5. H. M. El-Bakry, "Human Iris Detection Using Fast Cooperative Modular Neural Networks and Image Decomposition," Machine Graphics & Vision Journal (MG&V), Vol. 11, No. 4, (2002) 498-512
6. E. Alpaydin, " Multiple Networks for Function Learning, " Int. Conf. on Neural Networks, Vol.1 CA, USA, (1993) 9-14
7. A. Waibel, "Modular Construction of Time Delay Neural Networks for Speech Recognition, " Neural Computing 1, (1989) 39-46
8. K. Joe, Y. Mori, and S. Miyake, "Construction of a large scale neural network: Simulation of handwritten Japanese Character Recognition, " on NCUBE Concurrency Vol. 2, No. 2, (1990) 79-107
9. H. M. El-bakry, "Automatic Personal Identification Using Neural Networks, " The 24th international Conf. On Statistics computer Science, and its applications, Egypt, cairo, (1999) 405-416
10. S. Ben-Yacoub, "Fast Object Detection using MLP and FFT, " IDIAP-RR 11, IDIAP, (1997)
11. H. M. El-Bakry, "Fast Face Detection Using Neural Networks and Image Decomposition," Proc. of the 6th International Computer Science Conference, Active Media Technology, Dec. 18-20, 2001, Hong Kong – China, (2001) 205-215
12. H. M. El-Bakry, "Face Detection Using Fast Neural Networks and Image Decomposition," Proc. of INNS-IEEE International Joint Conference on Neural Networks, 14-19 May, 2002, Honolulu, Hawaii, USA (2002)
13. H. M. El-Bakry, "Comments on Using MLP and FFT for Fast Object/Face Detection," Proc. of IEEE IJCNN’03, Portland, Oregon, July, 20-24, (2003) 1284-1288
14. H. M. El-Bakry, and H. Stoyan, "Fast Neural Networks for Sub-Matrix (Object/Face) Detection," Proc. of IEEE International Symposium on Circuits and Systems, Vancouver, Canada, 23-26 May, (2004)
15. H. M. El-Bakry, "Fast Sub-Image Detection Using Neural Networks and Cross Correlation in Frequency Domain," Proc. of IS 2004: 14th Annual Canadian Conference on Intelligent Systems, Ottawa, Ontario, 6-8 June, (2004)
16. H. M. El-Bakry, and H. Stoyan, "Fast Neural Networks for Code Detection in a Stream of Sequential Data," Proc. of CIC 2004 International Conference on Communications in Computing, Las Vegas, Nevada, USA, 21-24 June, (2004)
17. H. M. El-Bakry, "Fast Neural Networks for Object/Face Detection," Proc. of 5th International Symposium on Soft Computing for Industry with Applications of Financial Engineering, June 28 - July 4, Sevilla, Andalucia, Spain, (2004)
18. H. M. El-Bakry, and H. Stoyan, "A Fast Searching Algorithm for Sub-Image (Object/Face) Detection Using Neural Networks," Proc. of the 8th World Multi-Conference on Systemics, Cybernetics and Informatics, 18-21 July, Orlando, USA (2004)

19. H. M. El-Bakry, and H. Stoyan, "Fast Neural Networks for Code Detection in Sequential Data Using Neural Networks for Communication Applications," Proc. of the First International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2004, Orlando, Florida, USA, Vol. IV, 21-25 July, (2004) 150-153
20. H. M. El-Bakry, and Q. Zhao, "Fast Object/Face Detection Using Neural Networks and Fast Fourier Transform," International Journal of Signal Processing, vol.1, no.3, (2004) 182-187
21. H. M. El-Bakry, and Q. Zhao, "A Fast Neural Algorithm for Serial Code Detection in a Stream of Sequential Data," International Journal of Information Technology, vol.2, no.1, (2005) 71-90
22. H. M. El-Bakry, and Q. Zhao, "Fast Pattern Detection Using Neural Networks Realized in Frequency Domain," Proc. of the International Conference on Pattern Recognition and Computer Vision, The Second World Enformatika Congress WEC'05, Istanbul, Turkey, 25-27 Feb., (2005) 89-92
23. H. M. El-Bakry, and Q. Zhao, "Fast Pattern Detection Using Normalized Neural Networks and Cross Correlation in the Frequency Domain," accepted and under publication in the EURASIP Journal on Applied Signal Processing.

Face Detection Using Convolutional Neural Networks and Gabor Filters

Bogdan Kwolek

Rzeszów University of Technology,
W. Pola 2, 35-959 Rzeszów, Poland
bkwolek@prz.rzeszow.pl

Abstract. This paper proposes a method for detecting facial regions by combining a Gabor filter and a convolutional neural network. The first stage uses the Gabor filter which extracts intrinsic facial features. As a result of this transformation we obtain four subimages. The second stage of the method concerns the application of the convolutional neural network to these four images. The approach presented in this paper yields better classification performance in comparison to the results obtained by the convolutional neural network alone.

1 Introduction

Detecting and locating human faces in an image or a sequence of images are important tasks in applications like intelligent human-computer interaction, model-based coding of video sequences at very low bitrates and content-based video indexing. Given an image of arbitrary size, the task is to detect the presence of any human face appearing in the image. Face detection in complex scenes is a challenging task since human faces may appear in different scales, orientations and with different head poses. Due to change of lighting condition, facial expressions, shadows, etc., the human face appearance could change considerably. Presence of glasses is another source of variations we have to take into account.

Several approaches, such as support vector machines [10], Bayesian classifiers [10], neural networks [7][5] have been proposed so far for detection of facial regions. The face knowledge-based detector [7] first preprocess the image subwindow and applies a neural network to detect whether it contains a face. The neural network has three types of hidden units: four looking at 10x10 pixel subregions, six looking at overlapping 20x5 pixel subregions and sixteen looking at 5x5 subregions. These subregions have been chosen to represent facial features that are important to face detection in pixel windows of size 20x20. To reduce the number of false positives multiple networks are applied. They have been trained in a similar manner but under different initial conditions and with different self-selected non-face examples. A skin-color detector has been used at the preprocessing stage to limit the amount of searching. Another neural network-based approach for finding frontal faces has been presented in work [8]. The algorithm uses a modified k-means clustering algorithm to extract the six

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 551–556, 2005.

face and non-face pattern centroids and their cluster covariance matrices from normalized input patterns. A multi-layer perceptron has been applied to classify the face and non-face patterns using different feature vectors of 12 distance measurements. The network contains 12 pairs of input units, one output unit and 24 hidden units. This work reported 79.9% to 96.3% detection rates with different data sets. The work [1] indicated that in the face recognition the face representation which is obtained on the basis of 2D Gabor filters is more robust against illumination variations than that of intensity-based. Gabor filter-based features have been used in several approaches to face recognition [10] and little work has been done to apply they to face detection. The work [5][2] presents results which were obtained during experiments with detection of faces in static images using convolutional neural networks.

Convolutional neural networks use the local receptive fields, shared weights and subsampling in order to extract and then to combine local features in a distortion-invariant manner. The feature extractor is created by the learning process and it is integrated into the classifier. The number of free parameters in a convolutional neural network is much less than in a fully-connected neural network with comparable classification capabilities due to the weight sharing. Our method uses Gabor filter-based features instead of the raw gray values as the input for a convolutional neural network to take advantage of both methods. The choice of Gabor filter responses is biologically motivated since they model the response of human visual cortical cells [3]. Gabor filters remove most of variation in lighting and contrast and can reduce intrapersonal variation. They are also robust against small shifts and small object deformations.

The remainder of the paper is organized as follows. In the next section we discuss the Gabor filter. In section 3. the components and details of convolutional neural network are presented. Section 4. reports results which were obtained in experiments. Finally, some conclusions follow in the last section.

2 Facial Features Extraction Using Gabor Filters

The main advantage of Gabor wavelets is that they allow analysis of signals at different scales, or resolution, and further they accommodate frequency and position simultaneously. The Gabor wavelet is essentially a sinewave modulated by a Gaussian envelope. The 2-D Gabor filter kernel is defined as follows:

$$f(x, y, \theta_k, \lambda) = exp\left[-\frac{1}{2}\left\{\frac{R_1^2}{\sigma_x^2} + \frac{R_2^2}{\sigma_y^2}\right\}\right] exp\left\{i\frac{2\pi R_1}{\lambda}\right\} \tag{1}$$

where $R_1 = xcos\theta_k + ysin\theta_k$ and $R_2 = -xsin\theta_k + ycos\theta_k$, σ_x and σ_y are the standard deviations of the Gaussian envelope along the x and y dimensions, λ and θ_k are the wavelength and orientation of the sinusoidal plane wave, respectively. The spread of the Gaussian envelope is defined in terms of the wavelength λ. θ_k is defined by $\theta_k = \frac{\pi(k-1)}{n}$, $k = 1, 2, ..., n$, where n denotes the number of orientations that are taken into account. For example, when $n = 2$, two values of orientation θ_k are used: 0^o and 90^o.

A Gabor filter response is achieved by convolving a filter kernel given by (1) with the image. The response of the filter for sampling point (x, y) is given by:

$$g(x, y, \theta_k, \lambda) = \sum_{u=-(N-x)}^{N-x-1} \sum_{v=-(N-y)}^{N-y-1} I(x+u, y+v) f(u, v, \theta_k, \lambda) \qquad (2)$$

where $I(x, y)$ denotes a NxN grayscale image.

In this work two different orientations and two different wavelengths are utilized. Therefore, different facial features are selected, depending on the response of each filter. In frontal or near frontal face image the eyes and mouth are oriented horizontally, while the nose constitutes vertical orientation. Fig. 1 depicts some Gabor filtered images of face samples of size 20x20. We can observe that the orientation properties of the face pattern have been highlighted. In particular, the eyes, nose and mouth have come out quite well. This does demonstrate the Gabor wavelet's capability to select localized variation in image intensity.

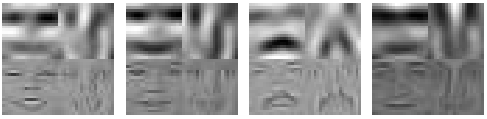


Fig. 1. Gabor filtered images

3 Convolutional Neural Architecture

A convolutional neural network [6] is a special kind of a feedforward neural network. It incorporates prior knowledge about the input signal and its distortions into its architecture. Convolutional neural networks are specifically designed to cope with the variability of 2D shapes to be recognized. They combine local feature fields and shared weights as well as utilize spatial subsampling to ensure some level of shift, scale and deformation invariance. Using the local receptive fields the neurons can extract simple visual features such as corners, end-points. These elementary features are then linked by the succeeding layers to detect more complicated features.

A typical convolutional network contains a set of layers each of which consists of one or more planes. Each unit in the plane is connected to a local neighborhood in the previous layer. The unit can be seen as a local feature detector whose activation characteristic is determined in the learning stage. The outputs of such a set of units constitute a feature map. Units in a feature map are constrained to perform the same operation on different parts of the input image or previous feature maps, extracting different features from the same image. A feature map can be obtained in a sequential manner through scanning the input image by a single unit with weights forming a local receptive field and storing the outputs of this unit in corresponding locations in the feature map. This operation is equivalent to a convolution with a small kernel. The feature map can be treated as a plane of units that share weights. The subsampling layers which usually follow layers with local, convolutional feature maps introduce a certain level of invariance to distortions and translations. Features of decreasing spatial resolution and of increasing complexity as well as globality are detected by the units in the successive layers.

The convolutional neural network we use consists of 6 layers. Layer C1 performs a convolution on the Gabor filtered images using an adaptive mask. The weights in the convolution mask are shared by all the neurons of the same feature map. The receptive fields of neighboring units overlap. The size of the scanning windows was chosen to be 20x20 pixels. The size of the mask is 5x5 and the size of the feature map of this layer is 16x16. The layer has 104 trainable parameters. Layer S2 is the averaging/subsampling layer. It consists of of 4 planes of size 16 by 16. Each unit in one of these planes receives four inputs from the corresponding plane in C1. Receptive fields do not overlap and all the weights are equal within a single unit. Therefore, this layer performs a local averaging and 2 to 1 subsampling. The number of trainable parameters utilized in this layer is 8. Once a feature has been extracted through the first two layers its accurate location in the image is less substantial and spatial relations with other features are more relevant. Therefore layers S1 and C2 are partially connected, and the task of such a configuration is to discover the relationships between different features.

Layer C2 is composed of 14 feature maps. Each unit contains one or two receptive fields of size 3x3 which operate at identical positions within each S1 maps. The first eight feature maps use single receptive fields. They form two independent groups of units responsible for distinguishing between face and nonface patterns. The remaining six feature maps take inputs from every contiguous subsets of two feature maps in S1. This layer has 140 free parameters. Layer S2 plays the same role as the layer S1. It is constructed of 14 feature maps and has 28 free parameters. In the next layer each of 14 units is connected only to the corresponding feature map of the S2 layer. It has 140 free parameters. Finally, the output layer has one node that is fully connected to the all the nodes from the previous layer. The network contains many connections but relatively few free trained parameters. Weight sharing allows to considerably reduce the number of free parameters and improves the generalization capability.

Training of our network has been realized in a supervised manner by using the back-propagation algorithm which has been adapted for convolutional neural networks. The partial derivatives of the activation function with respect to each connection have been computed, as if the network were a typical multi-layer one. Then the partial derivatives of all the connections that share the same parameter have been added to construct the derivative with respect to that parameter.

The recognition performance of a learned system is dependent on the size and quality of the training set. The face detector was trained on 3000 non-face patches collected from about 1500 images and 1500 faces covering out-of-plane rotation in the range $-20^o, ..., 20^o$. All faces were manually aligned by eyes position. For each face example the synthesized faces were generated by random in-plane rotation in the range $-10^o, ..., 10^o$, random scaling about $\pm 10\%$, random shifting up to ± 1 pixel and mirroring. All faces were then cropped and re-scaled to windows of

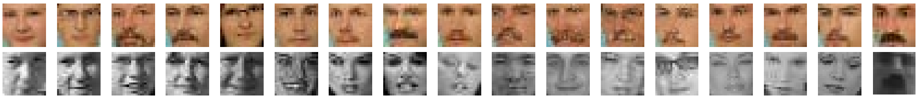


Fig. 2. Some examples from the training gallery

size 20x20 pixels while preserving their aspect ratio, see Fig. 2. Such a window size is considered in the literature as the minimal resolution that can be used without loosing critical information from the face pattern. The training collection contains also images acquired from our video cameras. The most of the training images which were obtained from WWW are of very good quality. The images obtained from cameras are of second quality, see also the last subimage from sequence demonstrated in Fig. 2. To provide more false examples we utilized a training with bootstrapping [7]. By using bootstrapping we iteratively gathered examples which were close to the boundaries of face and non-face clusters in the early stages of training. The activation function in the network was a hyperbolic tangent. Training the face detector took around 60 hours on a 2.4 GHz Pentium IV-based PC. There was no overlap between the training and test images.

4 Experimental Results

The experiments described in this section were carried out with a commercial binocular Megapixel Stereo Head. The depth map covering a face region is usually dense because human face is rich in details and texture. Thanks to such a property the stereovision provides a separate source of information and allows us to avoid expensive scaling down the subimages during searching for faces at all scales. A skin color detector is the first classifier in our system [4]. This fast classifier discards most of non-skin regions and therefore provides a focus of attention strategy guiding the searching for faces to only promising regions. It could detect almost all the promising regions. To find the faces the detector moves a scanning subwindow by a pre-determined number of pixels within only skin-like regions. The output of the face detector is then utilized to initialize our face/head tracker [4]. Fig. 3 depicts a typical scenario in which the face detector has been tested. The detector operates on images of size 320x240 and can process 2-5 images per second depending on the image structure.

To estimate the recognition performance we utilized only the static gray images. We obtained a detection rate of 87.5% on a test data-set containing 1000 face samples and 10000 non-face samples. Using only the convolutional network we obtaied a detection rate of 79%. This is a result of relatively simple structure of the network. It is worth to notice that such a relatively simple architecture of the network has been chosen to provide face detection in real-time using the available computational resources. The system achieves a much better recognition performance that using the convolutional neural network alone. It is much easier to train a convolutional neural network using a Gabor filtered input images than a network which uses raw images or histogram equalized images.

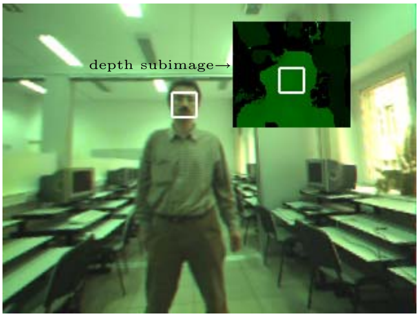


Fig. 3. Face detection

5 Conclusion

The experimental results we have obtained are very promising both in detection rates and processing speed. The Gabor filter has been used to capture efficient features for a convolutional neural network. The system achieves a much better recognition performance that using the convolutional neural network alone. The advantage of the proposed approach is that it achieves high face detection rates and real-time performance due to no exhaustive searching on the whole image.

References

1. Adini, Y., Moses, Y., Ullman, S.: Face recognition: The problem of compensating for changes in illumination direction, IEEE Trans. on Patt. Anal. Mach. Intell., vol. 19, no. 7 (1997) 721–731
2. Garcia, Ch., Delakis, M.: A neural architecture for fast and robust face detection, Int. Conf. on Pattern Recognition (2002) 44–47
3. Jones, J., Palmer, L.: An evaluation of the two dimensional Gabor filter model of simple receptive fields in cat striate cortex, Journal of Neurophysiology, vol. 58 (1987) 1233–1258
4. Kwolek, B.: Stereovision–based head tracking using color and ellipse fitting in a particle filter, 8th European Conf. on Computer Vision, LNCS 3024 (2004) 192–204
5. Lawrence S., Giles C.L., Tsoi A., Back, A.: Face recognition: A convolutional neural network approach, IEEE Trans. on Neural Networks, vol. 8, no. 1 (1997) 98–113
6. LeCun, Y., Bengio, Y.: Convolutional networks for images, speech, and time-series, In: M.A. Arbib, ed., The handbook of brain theory and neural networks, MIT Press (1995)
7. Rowley, H.A, Baluja, S., Kanade, T.: Neural network-based face detection, IEEE Trans. on Patt. Anal. Mach. Intell., vol. 20, no. 1 (1998) 23-38
8. Sung, K.K., Poggio, T.: Example based learning for view-based human face detection, IEEE Trans. on Patt. Anal. Mach. Intell., vol. 20, no. 1 (1998) 39–50
9. Yang, M-H., Kriegman, D., Ahuja, N.: Detecting faces in images: A survey, IEEE Trans. on Patt. Anal. Mach. Intell., vol. 24, no. 1 (2002) 34–58
10. Zhang, J., Yan, Y., Lades, M.: Face recognition: Eigenface, elastic matching, and neural nets, Proc. of IEEE, vol. 85 (1997) 423-435

Face Identification Performance Using Facial Expressions as Perturbation

Minoru Nakayama[1] and Takashi Kumakura[2]

[1] CRADLE, Tokyo Institute of Technology,
Ookayama, Meguro-ku, Tokyo 152-8552, Japan
[2] Human System Science, Tokyo Institute of Technology,
Ookayama, Meguro-ku, Tokyo 152-8552, Japan

Abstract. The paper presents improvements in face identification performance using synthesized images as a perturbation method. Three facial expression features, smiles, anger and screams, were extracted from images of actual facial expression using the eigenspace method. Synthesized facial images based on these features were added to learning data of a personal identification model using support vector machines (SVM). The performance of this model was significantly higher than that of a model trained without facial expression images, but significantly lower than that of a model using actual expression images. The results suggest that identification performance also depends significantly on facial expression.

1 Introduction

Personal identification systems using facial images have been studied variously. To achieve optimum performance using facial image recognition, as many images of an individual as possible should be gathered. However it is not easy to get enough facial images suitable to develop an identification system. The reason why the system requires many different images is to enable it to obtain some feature and variation of the facial images. In particular, transformations of the face which cause face rotation and other changes, such as frowns and open mouths should be considered [1,2,3].

Due to these robustness for the factors, additional training data is often generated to obtain robustness by putting perturbation on an original set of data [4], using the perturbation method. Effectiveness of this method for the factor of face rotation or conditions of illumination have been examined.

The aim of this paper is to examine whether face identification performance improves when additional training data is generated using perturbation methods, such as putting facial expression features on a neutral facial image. The expression features of the facial image are achieved using the eigenspace method [5], and then synthesized facial images are obtained using linear summation of an expression feature and a neutral facial image. The identification performance is compared to sets of training data of neutral facial images, synthesized facial images, and actual images of facial expressions.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 557–562, 2005.

2 Method

2.1 Facial Image Data

To extract expression features and to test identification performance, facial images of the AR Face database [6] were analyzed. This database contains four facial expression images of individuals: smiles, anger, screams and neutral expressions. The data consists of multiple images of individuals, taken on each of two consecutive days. The images of 113 individuals were selected as appropriate facial images for the analysis.

The expression features were extracted from images of 30 individuals which were selected at random. The images of the remaining 83 individuals taken each day were applied to training and identification tests respectively. As a result, 240 images (30 images × 4 expressions × 2 days) for feature extraction, 581 images (83 images × 4 neutral expressions and 3 other expressions) for training, and 332 images (83 images × 4 expressions) for identification tests were extracted.

The target facial images were standardized as 30 × 30 pixel, 256 gray scale images, clipped from the original images. The center position of the clipping was located at the mid-point between the right and left eyes.

2.2 Expression Feature Extraction

Kurozumi et al. [5] suggest that it is possible to extract expression features from facial images and create synthesized facial expression images based on the eigenspace method. Using the following method, we obtained these features from an image set for analysis.

An image set for analysis is defined as F, and the number of images which are included for a class $f \in F$ is M_f. A facial image is converted to an N dimensional vector x which consists of 256 gray scale level values for each pixel. The m th image ($1 \leq m \leq M_f$, where m is an integer), which is included for a class f, is noted as x_{fm}.

The between-class variance S_B, and within-class variance S_W are derived from following equations respectively.

$$S_B = \frac{1}{M} \sum_{f \in F} M_f (\bar{x}_f - \bar{x})(\bar{x}_f - \bar{x})^t \tag{1}$$

$$S_W = \frac{1}{M} \sum_{f \in F} \sum_{m=1}^{M_f} (x_{fm} - \bar{x}_f)(x_{fm} - \bar{x}_f)^t \tag{2}$$

Where $\bar{x}$ gives a mean of x_{fm} across f and m, $\bar{x}_f$ gives a mean of x_{fm} across m on an f, and $M = \sum_{f \in F} M_f$.

In this paper, each expression feature is extracted from an image subset of neutral and another facial expressions, such as neutral expressions and smiles for example.

An image vector x_{fm} is approximated using $K(K \leq M-1)$ normalized orthogonal bases, $\phi_k = (\phi_{1k}, \cdots, \phi_{Nk})^t, k = 1, \cdots, K$.

$$\tilde{x}_{fm} = \sum_{k=1}^{K} \sigma_{kfm}\phi_k + \bar{x} \tag{3}$$

$$\sigma_{kfm} = (x_{fm} - \bar{x})^t \phi_k \tag{4}$$

The k th σ_{kfm} is given as the above equation (4). Here, $S = S_B - S_W$ and ϕ_k are derived by following the eigenvalue problem [5].

$$S\phi_k = \lambda_k \phi_k \tag{5}$$

In this analysis, λ_k is the difference between between-class and within-class variances of σ_{kfm}, and it is defined as the degree of separation. An eigenvector ϕ_k, which corresponds to the maximum eigenvalue λ_k, represents features of smile expressions (ϕ_f).

Eigenvectors as features of scream expressions and anger are obtained using the same procedure.

3 Experiment

3.1 Procedure for Synthesizing Facial Expression Images

It is assumed that a synthesized facial image is achieved using the following equation, if the neutral expression is a mean of all facial expression images for individuals.

$$\tilde{x} = x_0 + zV_f \tag{6}$$

Here, V_f is a transformed feature vector from ϕ_f for expression f, which is adjusted for 256 gray scale presentation. A weight z for synthesizing a facial expression image is controlled as $z = -20, -16, -12, -8, -4, 0, 4, 8, 12, 16, 20$ by observing synthesized facial images.

3.2 Identification Model Training

The personal identification model has been developed by SVM, using the SVMTorch package [7,8]. The training/test data consists of 256 gray scaled 900 dimensions (30 × 30 pixels) with identification numbers. The SVM was trained with gaussian kernel, using the parameter $STD = 7000$.

Three training data sets were prepared on the first day from 83 individuals facial images, to examine the effectiveness of the perturbation method in generating additional synthesized facial expression images from a neutral expression set.

As a result, "neutral" sets consist of 332 images (83 × 4 shots); "synthesized" sets consist of 10,956 images (83 × 4 shots × 11 synthesized × 3 expressions); and "real" sets consist of 581 images (83 × 3 expressions and 4 neutral expressions).

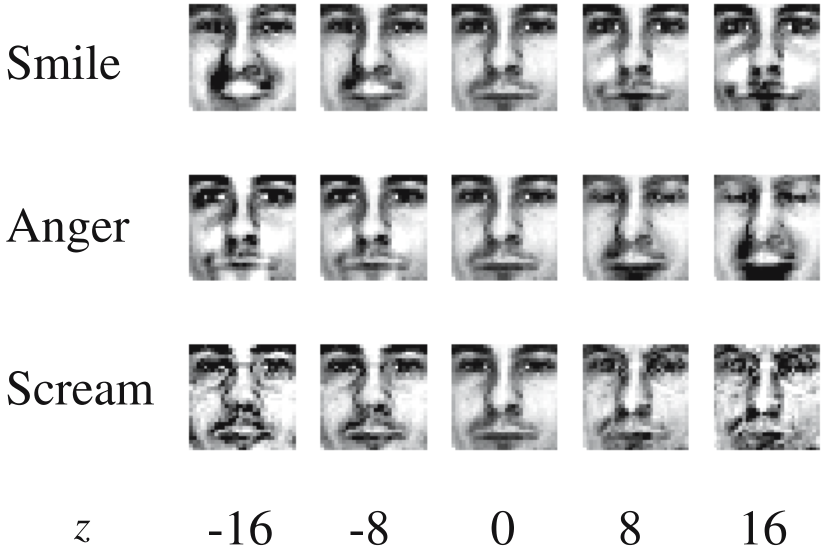


Fig. 1. Samples of synthesized face image

4 Experimental Results

The personal identification performance for each model was evaluated with a test data set which consisted of 332 facial images from the second day (83 × 3 expressions and a neutral expression).

The performance of personal identification using biometrics is generally evaluated using indices of False Acceptance Ratio (FAR) and False Rejection Ratio (FRR) [9]. In this paper, identification performance is evaluated by indices of recall rate and precision rate which are often used for document retrieval performance. Here, recall rate (R) means the performance in identifying a person, and precision rate (P) means the performance in omitting other persons. Additionally, a harmonic mean of R and P is defined as $F1(R,P) = \frac{2RP}{R+P}$ [10].

The recall rate and precision rate for each model are summarized in the first rows of Table 1 and Table 2 respectively. The results suggest that both rates increase gradually from “neutral” to “synthesized” and to “real” models. To examine the performance difference across the three models, one-way ANOVA was conducted for the recall and precision rates. As a result, the source of a model is significant for both rates and for the $F1$ measure (recall: $F(2, 164) = 51.4, p < 0.01$; precision: $F(2, 164) = 61.5, p < 0.01$; $F1$: $F(2, 164) = 63.9, p < 0.01$). There are significant differences for all indices amongst the three models. This suggests that identification performance of “synthesized” models is significantly higher than “neutral” models, and significantly lower than “real” models however. The result provides evidence that face identification performance improves when additional training data is generated by using the perturbation method.

Table 1. Recall rate

	Identification Model		
Images	Neutral	Synthesized	Real
Total	.68	.77	.90
Neutral	.93	.95	.96
Smile	.69	.86	.93
Anger	.81	.84	.95
Scream	.30	.43	.77

Table 2. Precision rate

	Identification Model		
Images	Neutral	Synthesized	Real
Total	.60	.72	.86
Neutral	.89	.93	.95
Smile	.59	.79	.89
Anger	.74	.79	.93
Scream	.19	.35	.68

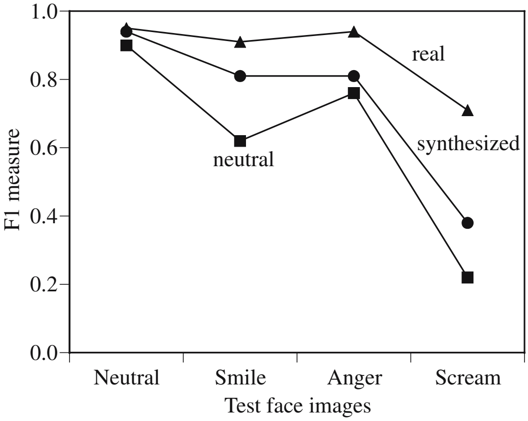


Fig. 2. $F1$ measures across facial expression of test images

The identification performance of each expression image is also summarized in Tables 1 and 2. It shows that the rates change with the models and facial expressions. The change in $F1$ measure is illustrated across expressions in Figure 2. The $F1$ for "real" models shows good performance without scream expressions. The "neutral" model performance is the worst without neutral expressions. The performance of "synthesized" models is better for smile and anger expressions, but the performance falls down for scream expressions. To obtain performance differences amongst models for each facial expression, one-way ANOVA was conducted. There are no significant differences for neutral expressions. For smile expressions, there is a significant difference ($p < 0.05$) in performance between "synthesized" and "neutral". However, this model has the same $F1$ level as the "neutral" model. For scream expressions, performance increases significantly in the order of "neutral", "synthesized", and "real".

5 Conclusion

This paper examines the idea that identification performance for facial expressions can be improved without gathering actual images of facial expressions. The additional data for model training was generated using the perturbation method by synthesizing face images which are based on facial expression features extracted using the eigenspace method.

The performance of three identification models, which were trained with "neutral", "synthesized" or "real" image sets, was compared. As a result, the identification performance of "synthesized" models is significantly better than "neutral" models across $F1$ measure, recall and precision rates. Also, the performance of the "synthesized" models is significantly better than the "neutral" model for all facial expressions without screams.

The results provide evidence that face identification performance can be improved when additional training data such as synthesized images is generated for facial expressions using the perturbation method.

The application of other techniques to the identification process will be the subject of further study.

References

1. N. Nakayama, S. Haruyama and S. Sakano, Face Recognition using Perturbation Method, IEICE Technical Report, PRMU98-90:45–51, 1998.
2. A. Inoue, S. Sakamoto and A. Sato, Face Recognition using Local Area Matching and Perturbed Subspace Method, *9th Symposium on Sensing via Image Information* (SSII 03), pages 555–560, 2003.
3. S. Clippingdale and A. Matsui, Perturbation of Deformable Face Templates Improves Robustness of Face Recognition to Facial Expression, IEICE Technical Report, PRMU2003-163:73–78, 2003.
4. D. DeCoste and B. Schölkopf, Training Invariant Support Vector Machines, *Machine Learning*, 46:161–190, 2002.
5. T. Kurozumi, Y. Shinza, Y. Kenmochi and K. Kotani, Facial Individuality and Expression Analysis by Eigenspace Method based on Class Features or Multiple Discriminant Analysis, *IEEE International Conference on Image Processing*(ICIP-99 Kobe), 25PP6A, 1999.
6. A. M. Martinez and R. Benavente, The AR Face Database. CVC Technical Report #24, 1998.
   http://rvl1.ecn.purdue.edu/˜aleix/aleix_face_DB.html.
7. R. Collobert and S. Bengio, SVMTorch: Support Vector Machines for Large-Scale Regression Problems, *Journal of Machine Learning Research*, 1:143–160, 2001.
8. R. Collobert, SVMTorch II, http://www.idiap.ch/
9. S. Bengio and J. Mariethoz, The expected performance curve: a new assessment measure for person authentication, *Proceedings of Odyssey 2004: The Speaker and Language Recognition Workshop*, 2004.
10. J. M. Pierre, On the Automated Classification of Web Sites, *Computer and Information Science*, Vol.6, nr0:1–12, 2001.

Discriminative Common Images for Face Recognition[*]

Vo Dinh Minh Nhat and Sungyoung Lee

Kyung Hee University – South of Korea
{vdmnhat, sylee}@oslab.khu.ac.kr

Abstract. Linear discrimination analysis (LDA) technique is an important and well-developed area of image recognition and to date many linear discrimination methods have been put forward. Basically, in LDA the image always needs to be transformed into 1D vector, however recently two-dimensional PCA (2DPCA) technique have been proposed. In 2DPCA, PCA technique is applied directly on the original images without transforming into 1D vector. In this paper, we propose a new LDA-based method that applies the idea of two-dimensional PCA. In addition to that, our approach proposes an method called Discriminative Common Images based on a variation of Fisher's LDA for face recognition. Experiment results show our method achieves better performance in comparison with the other traditional LDA methods.

Keywords: Fisherfaces, Linear discrimination analysis (LDA), Discriminative Common Image, face recognition.

1 Introduction

The Fisherface method [4] combines PCA and the Fisher criterion [9] to extract the information that discriminates between the classes of a sample set. It is a most representative method of LDA. Nevertheless, Martinez *et al.* demonstrated that when the training data set is small, the Eigenface method outperforms the Fisherface method [7]. Should the latter be outperformed by the former? This provoked a variety of explanations. Liu *et al.* thought that it might have been because the Fisherface method uses all the principal components, but the components with the small eigenvalues correspond to high-frequency components and usually encode noise [11], leading to recognition results that are less than ideal. In [5], Yu *et al.* propose a direct LDA (DLDA) approach to solve this problem. It removes the null space of the between-class scatter matrix firstly by doing eigen-analysis. Then a simultaneous diagonalization procedure is used to seek the optimal discriminant vectors in the subspace of the between-class scatter matrix. However, in this method, removing the null space of the between-class scatter matrix by dimensionality reduction would indirectly lead to the losing of the null space of the within-class scatter matrix which

[*] This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment).

Corresponding Authors: Vo Dinh Minh Nhat (vo_dinhminhnhat@yahoo.com), and SungYoung Lee (sylee@oslab.khu.ac.kr).

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 563 – 568, 2005.

contains considerable discriminative information. Rui Huang [10] proposed the method in which the null space of total scatter matrix which has been proved to be the common null space of both between-class and within-class scatter matrix, and useless for discrimination, is firstly removed. Then in the lower-dimensional projected space, the null space of the resulting within-class scatter matrix is calculated. This lower-dimensional null space, combined with the previous projection, represents a subspace of the whole null space of within-class scatter matrix, and is really useful for discrimination. The optimal discriminant vectors of LDA are derived from it. In [14], a common vector for each individual class is obtained by removing all the features that are in the direction of the eigenvectors corresponding to the nonzero eigenvalues of the scatter matrix of its own class. The common vectors are then used for recognition. In their case, instead of using a given class's own scatter matrix, they use the within-class scatter matrix of all classes to obtain the common vectors.

In [15], a new PCA approach called Two-dimensional PCA (2DPCA), is developed for image feature extraction. As opposed to conventional PCA, 2DPCA is based on 2D matrices rather than 1D vectors. That is, the image matrix does not need to be transformed into vector. Instead, an image covariance matrix can be constructed directly using original image matrices. So, in this paper we improve the LDA algorithm based on the idea of two-dimensional PCA.

Generally, in this paper we improve the LDA-based algorithm by apply the 2D approach into the Discriminative Common Vectors [14] algorithm. Our new method takes the advantages of both 2DPCA method [15] for dealing with high dimensional data to avoid singularity and LDA-based algorithm [14] for dealing with small sample size problem. The remainder of this paper is organized as follows: In Section 2, the traditional LDA methods are reviewed. The idea of the proposed method and its algorithm are described in Section 3. In Section 4, experimental results are presented for the Yale face image databases to demonstrate the effectiveness of our method. Finally, conclusions are presented in Section 5.

2 Linear Discriminant Analysis

Suppose that we have *N* sample images $\{x_1, x_2, ..., x_N\}$ taking values in an *n*-dimensional image space. Let us also consider a linear transformation mapping the original *n*-dimensional image space into an *m*-dimensional feature space, where *m* < *n*. The new feature vectors $y_k \in \mathbb{R}^m$ are defined by the following linear transformation :

$$y_k = W^T x_k \tag{1}$$

where $k = 1, 2, ..., N$ and $W \in \mathbb{R}^{nxm}$ is a matrix with orthonormal columns.

Different objective functions will yield different algorithms with different properties. While PCA seeks directions that are efficient for representation, Linear Discriminant Analysis seeks directions that are efficient for discrimination. Assume that each image belongs to one of C classes $\{C_1, C_2, ..., C_C\}$. Let N_i be the number of the samples in class $C_i (i = 1, 2, ..., C)$, $\mu_i = \frac{1}{N_i} \sum_{x \in C_i} x$ be the mean of the samples in class X_i, $\mu = \frac{1}{N} \sum_{i=1}^{N} x_i$ be the mean of all samples. Then the between-class scatter matrix S_b is defined as

$$S_b = \frac{1}{N} \sum_{i=1}^{C} N_i (\mu_i - \mu)(\mu_i - \mu)^T = \frac{1}{N} \Phi_b \Phi_b^T \tag{2}$$

and the within-class scatter matrix S_w is defined as

$$S_w = \frac{1}{N} \sum_{i=1}^{C} \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T = \frac{1}{N} \Phi_w \Phi_w^T \tag{3}$$

In LDA, the projection W_{opt} is chosen to maximize the ratio of the determinant of the between-class scatter matrix of the projected samples to the determinant of the within-class scatter matrix of the projected samples, i.e.,

$$W_{opt} = \arg\max_W \frac{\left|W^T S_b W\right|}{\left|W^T S_w W\right|} = [w_1 w_2 ... w_m] \tag{4}$$

where $\{w_i \,|i = 1, 2, ..., m\}$ is the set of generalized eigenvectors of S_b and S_w corresponding to the m largest generalized eigenvalues $\{\lambda_i \,|i = 1, 2, ..., m\}$, i.e.,

$$S_b w_i = \lambda_i S_w w_i \quad i = 1, 2, ..., m. \tag{5}$$

3 Discriminative Common Vectors and Discriminative Common Images

The Discriminative Common Vectors approach can be summarized as follows, details can be referenced at [14] :

- Step 1: Compute the nonzero eigenvalues and corresponding eigenvectors of S_w. Set $Q = [\alpha_1 ... \alpha_r]$, where r is the rank of S_w.
- Step 2: Choose any sample from each class and project it onto the null space of S_w to obtain the common vectors.

$$x_{c\_com} = x_i - QQ^T x_i \quad , x_i \in C_c, \quad c = 1, ..., C \tag{6}$$

- Step 3: Compute the eigenvectors w_k of S_{com}, corresponding to the nonzero eigenvalues. With S_{com} is defined below. There are at most $C-1$ eigenvectors that correspond to the nonzero eigenvalues. Use these eigenvectors to form the projection matrix $W=[w_1 w_2 ... w_{C-1}]$, which will be used to obtain feature vectors.

$$S_{com}=\sum_{c=1}^{C}(x_{c\_com}-\mu_{com})(x_{c\_com}-\mu_{com})^T$$

$$\mu_{com}=\frac{1}{C}\sum_{c=1}^{C}x_{c\_com} \tag{7}$$

In Discriminative Common Images approach, the image matrix does not need to be previously transformed into a vector, so a set of *N* sample images is represented as $\{X_1, X_2, ..., X_N\}$ with X$_i$. ∈ $\mathbb{R}^{kxs}$. Then the between-class scatter matrix S_b is re-defined as

$$S_b=\frac{1}{N}\sum_{i=1}^{c}N_i(\mu_{C_i}-\mu_X)(\mu_{C_i}-\mu_X)^T \tag{8}$$

and the within-class scatter matrix S_w is re-defined as

$$S_w=\frac{1}{N}\sum_{i=1}^{c}\sum_{X_k\in C_i}(X_k-\mu_{C_i})(X_k-\mu_{C_i})^T \tag{9}$$

with $\mu_X=\sum_{i=1}^{N}X_i \in \mathbb{R}^{kxs}$ is the mean image of all samples and $\mu_{C_i}=\frac{1}{N_i}\sum_{X\in C_i}X$ be the mean of the samples in class C_i. The common vectors in (8) now become common images, and defined as

$$X_{c\_com}=X_i-QQ^TX_i \quad ,X_i\in C_c, \quad c=1,...,C \tag{10}$$

Also, S_{com} in (9) is re-defined as

$$S_{com}=\sum_{c=1}^{C}(X_{c\_com}-\mu_{com})(X_{c\_com}-\mu_{com})^T$$

$$\mu_{com}=\frac{1}{C}\sum_{c=1}^{C}X_{c\_com}\,. \tag{11}$$

4 Experimental Results

This section evaluates the performance of our propoped algorithm Discrinative Common Images (DCI) compared with that of the original Fisherface algorithm, Direct LDA algorithm, and Discriminative Common Vectors (DCV) based on using Yale face database. The database contains 5760 single light source images of 10 subjects each seen under 576 viewing conditions (9 poses x 64 illumination conditions). For every subject in a particular pose, an image with ambient (background) illumination was also captured.

Table 1. The recognition rates on Yale databases

k	Fisherface	Direct LDA	DCV	DCI
2	77.95	79.97	83.01	**85.73**
3	86.29	86.59	89.11	**93.78**
4	91.62	92.32	92.97	**95.80**
5	93.31	93.98	94.42	**97.51**
6	95.36	95.80	96.87	**98.67**

Firstly, we tested the recognition rates with different number of training samples. $k(k=2,3,4,5,6)$ images of each subject are randomly selected from the database for training and the remaining images of each subject for testing. For each value of k, 50 runs are performed with different random partition between training set and testing set, and *Table 1* shows the average recognition rates (%) with Yale database.

5 Conclusions

A new LDA-based method for face recognition has been proposed in this paper. In this paper, we propose a method that applies the idea of two-dimensional PCA. In addition to that, we proposes a method called Discriminative Common Images based on a variation of Fisher's LDA for face recognition. By solving the small sample size problem and high dimensionality, this paper proposes a practical algorithm for applying LDA on image recognition applications, and shows the efficiency in face recognition application. It has the advantage of easy training, efficient testing, and good performance compared to other linear classifiers.

References

1. M. Turk, A Pentland: Eigenfaces for recognition. Journal of Cognitive Neuroscience, Vol. 3 (1991) 71-86.
2. W. Zhao, R. Chellappa, P.J. Phillips: Subspace Linear Discriminant Analysis for Face Recognition. Technical Report CAR-TR-914, 1999.
3. D. L. Swets, J. J. Weng: Using discrimination eigenfeatures for image retrieval. IEEE Trans. Pattern Anal. Machine Intell., vol. 18 (1996) 831–836.

4. P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman: Eigenfaces vs. fisherface: Recognition using class specific linear projection. IEEE Trans. Pattern Anal. Machine Intell., Vol. 19 (1997) 711–720.
5. H. Yu, J. Yang: A direct LDA algorithm for high-dimensional data with application to face recognition. Pattern Recognit., Vol. 34 (2001) 2067–2070.
6. M. Loog, R. P. W. Duin, R. Haeb-Umbach: Multiclass linear dimension reduction by weighted pairwise fisher criteria. IEEE Trans. Pattern Anal. Machine Intell., Vol. 23 (2001) 762–766.
7. A. M. Martinez, A. C. Kak: PCA versus LDA. IEEE Trans. Pattern Anal. Machine Intell., Vol. 23 (2001) 228–233.
8. D. H. Foley, J. W. Sammon: An optimal set of discrimination vectors. IEEE Trans. Comput., Vol. C-24 (1975) 281–289.
9. R. A. Fisher: The use of multiple measurements in taxonomic problems. Ann. Eugenics, Vol. 7 (1936) 178–188.
10. Rui Huang, Qingshan Liu, Hanqing Lu, Songde Ma: Solving the small sample size problem of LDA. Pattern Recognition, 2002. Proceedings. 16th International Conference on , Vol 3 (2002)
11. C. Liu, H. Wechsler: Robust coding scheme for indexing and retrieval from large face databases. IEEE Trans. Image Processing, Vol. 9 (2000) 132–137.
12. Chengjun Liu, Wechsler H.: A shape- and texture-based enhanced Fisher classifier for face recognition. IEEE Trans. Image Processing, Vol. 10(2001) 598–608.
13. L. Chen, H. M. Liao, M. Ko, J. Lin, G. Yu: A new LDA-based face recognition system which can solve the small sample size problem. Pattern Recognit., Vol. 33 (2000) 1713–1726.
14. Cevikalp H., Neamtu, M., Wilkes, M.; Barkana, A.: Discriminative common vectors for face recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on , Vol. 27 (2005) 4 – 13
15. Jian Yang, Zhang, D., Frangi, A.F., Jing-yu Yang : Two-dimensional PCA: a new approach to appearance-based face representation and recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on , Vol. 26 (2004) 131 – 137
16. "The Yale face database" http://cvc.yale.edu/projects/yalefaces/yalefaces.html.

Classification of Face Images for Gender, Age, Facial Expression, and Identity[*]

Torsten Wilhelm, Hans-Joachim Böhme, and Horst-Michael Gross

Department of Neuroinformatics and Cognitive Robotics,
Ilmenau Technical University, P.O. Box 100565, 98684 Ilmenau, Germany

Abstract. In this paper we compare two models for extracting features from face images and several neural classifiers for their applicability to classify gender, age, facial expression, and identity. These models are i) a description of face images by their projection on independent base images and ii) an Active Appearance Model which describes the shape and grey value variations of the face images. The extracted feature vectors are classified with Nearest Neighbor, MLP, RBF and LVQ networks, and classification results are compared.

1 Introduction

A growing number of applications rely on the ability to extract information about people from images. Examples are person identification for surveillance or access control, the estimation of gender and age for building user models or facial expressions recognition, which can give valuable information for the evaluation of man-machine interfaces. As the mentioned recognition tasks have been addressed in isolation in the past, there often exists a variety of methods for each. The presented work was done in the context of building a man-machine interface for a mobile service robot [5], where all the above mentioned information is of great interest. Furthermore, our hope was to identify one method that could be used universally.

2 State of the Art

A commonly used method for face image analysis is the subspace projection of the image data, where the subspace can be spanned by principal components, independent components of the training data. This method was used for a vast amount of approaches for person identification and automatic facial expression analysis [1]. Another widespread method for person identification and for gender estimation is the Elastic Graph Matching technique [10] [7]. Elastic Graph Matching describes faces in terms of spatial frequencies in local image areas, where the relation between these areas is defined by a graph structure. These models are adaptive and can adjust themselves to some degree to variations in the image data. Active Appearance Models have been used for person identification and facial expression analysis [2] [3]. They describe the statistical variations of shape and grey values in the training data and adapt themselves in an iterative process to a given face image. Up to now, very little work was reported on age estimation from image data.

[*] This work is partially supported by TMWFK-Grant # B509-03007 to H.-M. Gross.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 569–574, 2005.

3 Dataset Coding Age, Gender, Identity, and Facial Expressions

There exists a variety of face databases designed for single classification tasks, e.g. the Cohn Kanade database for facial expressions [8]. However, we wanted to test the performance of our methods for classification tasks as diverse as gender, age, facial expressions, and identity. To eliminate the influence of varying quality of the image data, we recorded our own database according to our requirements. This database consists of two parts. The first part used for the classification of age, gender, and identity contains 70 people with 7 images each, with neutral facial expression, different illuminations, and small deviations in head orientation. Identities are equally distributed in the age range between 10 and 60 and equally distributed over genders. The second part consists of 30 identities with 7 images each, which represent the basic emotions happiness, sadness, surprise, fear, anger, disgust and neutral as identified by Ekman in [4], see Fig. 1. Since evoking facial expressions with movies was shown to produce mixtures of basic emotions [6], we decided to ask probands to pose facial expressions according to the seven basic emotions. As people's ability to pose facial expressions varies significantly, all the recorded images were manually classified by 10 people and only a subset of all images was used where at least 7 people agreed on the facial expression. This problem could be avoided, if the image data was labeled with FACS codes, describing the activity of facial muscles instead of basic emotions. However, since FACS coding is very time consuming and has to be done by trained personnel, it was not yet possible to obtain FACS codes for our data.

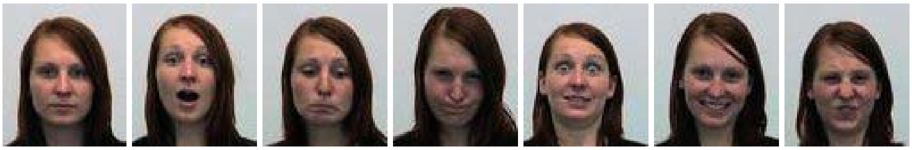


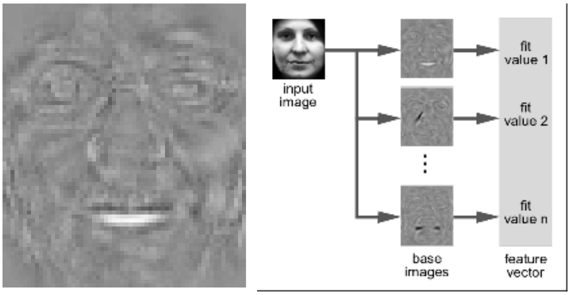
Fig. 1. Examples of the used data set for facial expressions. From left to right: neutral, surprise, sadness, anger, fear, happiness, disgust.

4 Feature Extraction

To provide the feature extraction methods with normalized data, we manually labeled the position of facial landmarks. We analyzed Independent Component Analysis (ICA) and Active Appearance Models (AAM) for feature extraction.

4.1 Independent Component Analysis

For ICA, only the centers of the eyes are used as facial landmarks. The ICA model depends on highly accurate aligned image data as the model does not adapt to a given face image. Thus, we applied affine transformations such that the center of the eyes are on the same position in every image. The size of these normalized images is 60 × 70 pixels. For building the ICA model, an observation matrix was built by using the vectorized images as rows. On this matrix we applied ICA and obtained independent



(a) (b)

Fig. 2. (a) Example of an independent base image. (b) When using the ICA model, a normalized input image is projected on the independent base images. The fit values for the base images form the feature vector to be classified.

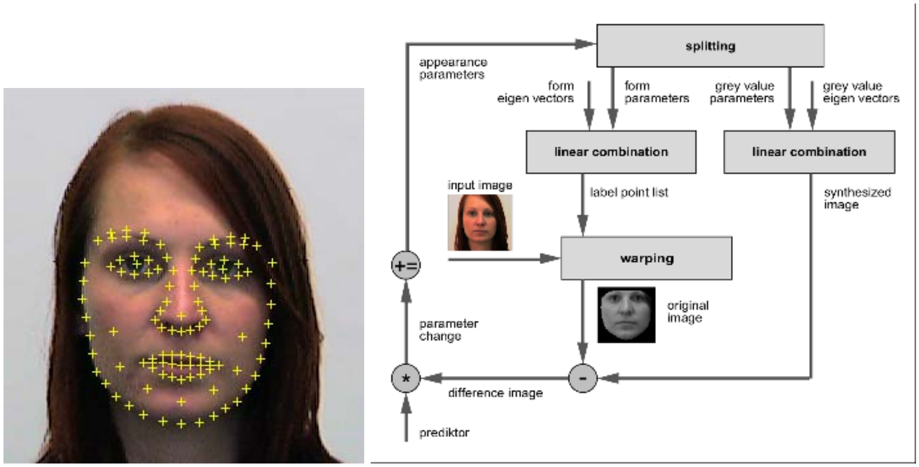
base images for the data, which represent local facial features, see Fig. 2(a). Any given normalized face image is represented as linear combination of independent base images, and the fit values constitute the data to be classified. For a more detailed description see [9]. The data flow when using the ICA model is depicted in Fig. 2(b).

4.2 Active Appearance Models

The facial landmarks for the Active Appearance Models (AAM) consist of 116 points along dominant outlines in the face, see Fig. 3(a). To construct an AAM, the mean shape of the training data is computed and the shape variation is determined by principal component analysis. In the next step, the training images are warped to the mean shape. In the same way as with the shape model, the mean grey value face is computed and the grey value variation is determined by principal component analysis. Finally, a predictor matrix is estimated by varying single appearance parameters and averaging their effects on the difference image. For details on Active Appearance Models see [2]. The data flow when using the AAM is depicted in Fig. 3(b). After adaptation of the model to a given face image, the resulting appearance parameters describe the shape and the grey value distribution of the given face and are used as feature vector for classification.

5 Classification

We used a leave-n-out strategy to train and test the classifiers. The partitioning was such, that every identity was in the test dataset exactly once. For gender, the test dataset consisted of 2 identities (male and female), for age of one identity per age group and for facial expressions of one person showing all facial expressions. For person identification we used 3 images from each person for training, 2 for validation, and 2 for testing. The results were averaged over the multiple training cycles performed for each recognition task. We compared the following network types: Multi Layer Perceptron (MLP), Radial Basis Function (RBF), Nearest Neighbor (NN), and Generalized Learning Vector Quantization (GLVQ). The number of inputs corresponds to the number of extracted features, that is, ICA fit values or appearance parameters, respectively. The MLPs had two trainable layers with 40 neurons in the hidden layer. The GLVQ network used 20 neurons per class. The centers of RBF networks were initialized by GLVQ training. GLVQ and RBF networks operated on normalized, NN and MLP on unnormalized feature vectors.



(a) (b)

Fig. 3. (a) Landmarks used for the construction of the Active Appearance Model. (b) Usage of the Active Appearance Model. The iterative search process begins with appearance parameter vector **0**, that is, with mean shape and mean grey value distribution. The position is initialized by the center of the eyes as with the ICA model. The input image is warped to the current shape to produce the form normalized *original image*. On the other hand, the current grey value parameters are used to produce the form normalized *synthesized image*. From the difference of these images, a parameter change is estimated for each appearance parameter with the goal to minimize the energy of the difference image. The search process converges, when this parameter change is **0**.

6 Results and Conclusions

Recognition rates are shown in Fig. 4. For gender and facial expressions, recognition rates are promising. Here, the best results were obtained with AAMs and MLP classifiers or ICA with Nearest Neighbor classifiers, respectively. For both feature extraction methods (ICA and AAM), the results for age classification are only slightly better than guessing (20% due to the used 5 classes). From the confusion matrixes it can be seen, that it is possible to distinguish young and old people, Table 1. For person identification it is often not feasable to have a fixed gallery represented by a neural classifier. Alternatively, two models either ICA fit values or active appearance parameters can be compared by the normalized dot product. Therefore, the false acceptance and false rejection rates were recorded for different similarity thresholds, see Fig. 5. Both methods achieve approximately the same equal error rates. Although the best recognition rates for gender and age estimation were obtained with AAMs, in the final system we deploy ICA with Nearest Neighbor classifiers only. This is a trade-off between recognition rates and processing time needed to adapt the AAM to the input image, which is about 2800ms compared to about 20ms for the subspace projection with ICA. Since the AAM proved its capability for various recognition tasks, our future work will focus on optimizing the AMM for processing speed. In contrast to the ICA subspace projection,

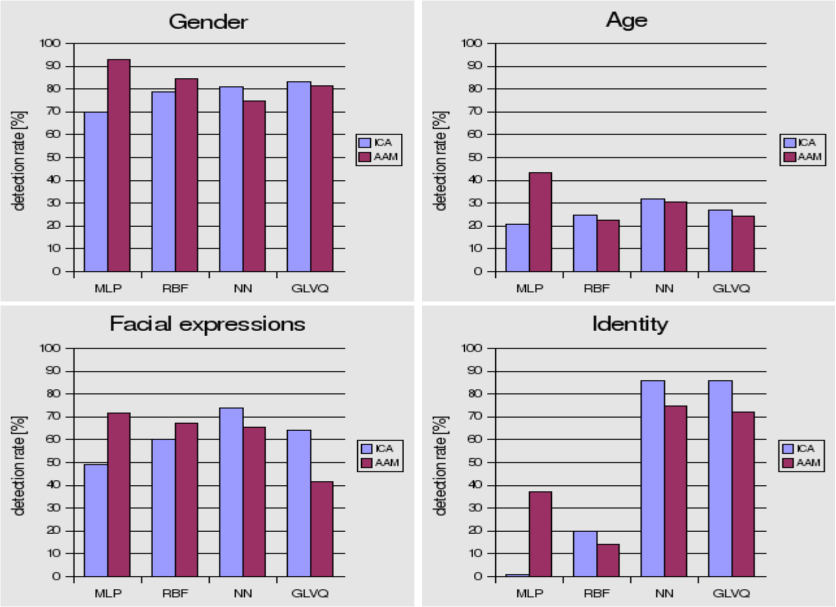


Fig. 4. Recognition rates for gender, age, facial expressions and identity on validation data for feature extraction with ICA or AAM, respectively. The best recognition rates w.r.t. gender, age and facial expression were obtained with AAM and MLP or ICA and NN, respectively. The high recognition rates for AAM with MLP classifiers suggest, that the AAM produces appearance parameters which are in contrast to the ICA fit values well clustered according to the recognition tasks. For person identification the ICA feature extraction performs significantly better than the AAM. Here MLP and RBF networks fail, because of the large number of clusters. However, in the final system, person identification is done by comparing two models and using a similarity threshold for acceptance or rejection, see Fig. 5.

Table 1. Confusion matrixes for age estimation. Horizontal: true class, vertical: estimated class (a) ICA + NN (b) AAM + MLP. Each age intervall contained 98 images to be classified. It can be seen, that both methods are roughly able to distinguish young from old people. Thus, when using the system only two or three clusters should be used.

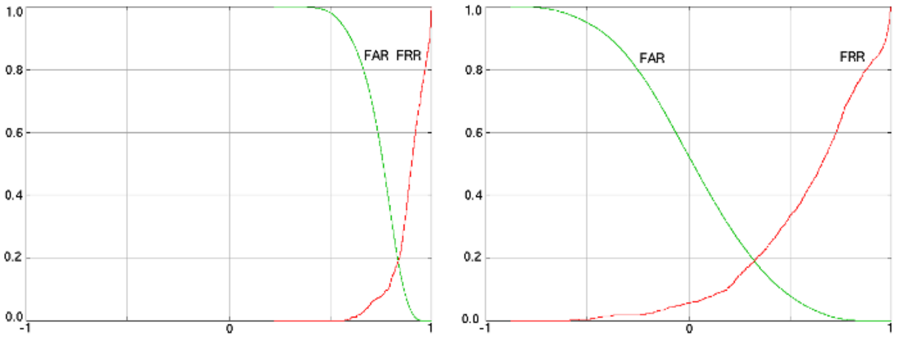
	10	**20**	**30**	**40**	**50**
10	48	15	18	10	7
20	9	36	22	6	25
30	18	32	20	15	13
40	6	19	29	23	21
50	7	32	21	19	19

(a)

	10	**20**	**30**	**40**	**50**
10	55	11	13	16	3
20	9	40	29	8	12
30	20	19	23	19	17
40	13	7	23	33	22
50	4	10	15	22	47

(b)

which relies on acurately aligned frontal views, the AAM is also able to handle and estimate head pose, provided this image variation is present in the training data.



(a) (b)

Fig. 5. False acceptance (FAR) and false rejection rate (FRR) curves for person identification with (a) ICA (b) AAM. The equal error rates are approximately the same with 0.2 for ICA and 0.19 for AAM, which suggests that both methods are equally well suited for person identification.

References

1. M.S. Bartlett. *Face image analysis by unsupervised learning*. Kluwer Academic Publishers, 2001.
2. Cootes, T.F., Edwards, G.J., and Taylor, C.J. Active appearance models. *Lecture Notes in Computer Science*, 1407:484–??, 1998.
3. Edwards, G.J., Taylor, C.J., and Cootes, T.F. Learning to identify and track faces in image sequences. In *ICCV*, pages 317–322, 1998.
4. Ekman, P. and Friesen, W.V. *Unmasking the face. A guide to recognizing emotions from facial clues*. Prentice-Hall, Englewood Cliffs, New Jersey, 1975.
5. Gross, H.-M., Boehme, H.-J., Key, J, and Wilhelm, T. The perses project - a vision-based interactive mobile shopping assistant. *Künstliche Intelligenz*, 4:34–36, 2000.
6. Gross, J.J. and Levenson, R.W. Emotion elicitation using films. *Cognition and Emotion*, 9:87–108, 1995.
7. Hong, H., Neven, H., and v. d. Malsburg, C. Online facial expression recognition based on personalized gallery. In *Intl. Conference on Automatic Face and Gesture Recognition*, pages 354–359. IEEE Comp. Soc., 1998.
8. Kanade, T., Cohn, J.F., and Tian, Y. Comprehensive database for facial expression analysis. *In Proc. of the Fourth IEEE Int. Conf. on Automatic Face and Gesture Recognition (FG'00), Grenoble*, pages 46–53, 2000.
9. Wilhelm, T. and Backhaus, A. Statistical and neural methods for vision-based analysis of facial expressions and gender. In *In Proc. IEEE Int. Conf. on System Man and Cybernetics*, pages 2203–2208, 2004.
10. Wiskott, L., Fellous, J.-M., Krüger, N., and von der Malsburg, C. Face recognition and gender determination. In *Int. Workshop on Automatic Face- and Gesture-Recognition, Zürich, June 26-28, 1995*, pages 92–97, 1995.

Classifying Unprompted Speech by Retraining LSTM Nets

Nicole Beringer[1], Alex Graves[1], Florian Schiel[2], and Jürgen Schmidhuber[1,3]

[1] IDSIA , Galleria 2, 6928 Manno-Lugano, Switzerland
{nicole, alex, juergen}@idsia.ch
[2] Schiel BAS Services, Schellingstr. 3, 80799 Munich, Germany
schiel@bas.phonetik.uni-muenchen.de
[3] TU Munich, Boltzmannstr. 3, 85748 Garching, Munich, Germany

Abstract. We apply Long Short-Term Memory (LSTM) recurrent neural networks to a large corpus of unprompted speech- the German part of the VERBMOBIL corpus. By training first on a fraction of the data, then retraining on another fraction, we both reduce time costs and significantly improve recognition rates. For comparison we show recognition rates of Hidden Markov Models (HMMs) on the same corpus, and provide a promising extrapolation for HMM-LSTM hybrids.

1 Introduction

It would be desirable to retrain an Automatic Speech Recognition (ASR) system on new data without losing the benefits of previous learning. For example, it may be necessary to adapt quickly to new input, or to use information gained from a previous task, e.g., recognizing read speech, in order to solve the next task, e.g., quasi-spontaneous (= unprompted) speech. In task/domain independent recognition [15], systems that are (pre-)trained under certain conditions and/or certain dialogue specifications are required to adapt to utterances recorded under different conditions or with different dialogue specifications. It has also become standard practice to train Hidden Markov Models (HMMs) on multiple corpora, in order to improve their robustness also with respect to new data. However, methods for adapting HMM's are complex, unintuitive, and time-consuming [11]. Most modern systems use a hybrid of HMMs and maximum likelihood linear regression to adapt to new training material.

Artificial neural nets (ANNs) lend themselves to a very simple form of retraining: train on one dataset, then continue training on another without resettting the weights. Recurrent Neural Nets (RNNs) are particularly promising for speech processing because they have the potential to learn a dynamic model of speech that incorporates multiple time scales without using time windows or fixed time delays. Unlike traditional RNNs, Long Short-Term Memory nets (LSTM) [10] can also handle long time lag correlations between inputs and errors, also in the context of speech applications [7]. Recent experiments with plain LSTM on speaker adaptation [8] suggest that retraining is fast and effective on **small**

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 575–581, 2005.

corpora, and that results of previous learning and generalization improve with retaining on randomly chosen subsets of the data. In this paper we apply this approach for the first time to Bidirectional LSTM [9] and a **large** corpus of unprompted speech.

The following section gives an overview of LSTM. Section 3 briefly describes the VERBMOBIL data used for both LSTM and HMM experiments. Section 4 describes the experimental setup. Section 5 analyses the experimental results of baseline and retrained LSTM for framewise phoneme prediction and gives results for the entire phonemes for plain HMMs on the same test set to show the task difficulty (unprompted speech). Section 6 provides an extrapolation of the framebased results for a HMM-LSTM hybrid (under development) based on previous comparisons of framewise and phoneme error rates on various corpora of read speech.

2 LSTM

“Long Short-Term Memory” [10,6] is a general purpose algorithm for extracting statistical regularities from noisy time series. It learns from scratch, typically with more adjustable parameters (the weights), a larger search space, and less initial bias [5] than HMMs, which incorporate prior linguistic knowledge.

2.1 Bidirectional LSTM (BLSTM)

The output of typical RNNs is based on the complete history of *previous* inputs. However, there are many sequence processing tasks where future inputs are also useful because reverse correlations exist. In speech, for example, the articulatory system is already preparing future utterances as it shapes the current one. A solution is bidirectional training [13,1,2]: the input is presented forwards and backwards to two separate recurrent nets, both of which are connected to the same output layer. In this way, errors can be injected as normal and backpropagated through the nets. Current results with BLSTM [9] show that it outperforms normal LSTM, as well as previous bidirectional RNNs on speech recognition tasks.

2.2 Retraining with Bidirectional LSTM

An in-depth investigation of retraining with LSTM [8] (i.e. presenting new data to an already trained network) showed that LSTM is capable of fast and effective relearning on speakers with widely varying vocal characteristics. The net was trained and successively retrained on disjoint subsets of the TIDIGITS database. The retraining time and difficulty diminished with repetition, and the net was able to transfer knowledge across several datasets. The final performance of the net was generally raised by having been previously trained on different datasets, and this improvement persisted over multiple retrainings.

Table 1. Basic numbers for the subsets in VM1 and VM2

	VM1_DEV	VM1_TEST	VM1_TRAIN	VM2_DEV	VM2_TEST	VM2_TRAIN
WORDS	15084	14615	285280	11905	9855	153438
TURNS	630	631	12600	592	592	11835
LEX	1537	1342	6472	1397	1264	5238
SPEAKER	35	33	629	13	13	119

3 Corpus Description

Our present investigation uses a database of *unprompted* speech- the VERBMOBIL (VM) corpus [17], which is more difficult to recognize than read speech such as the TIMIT corpus. The VM corpus is divided into VM1 (recordings up to 1996) and VM2 (recording after 1996). Both sets differ in recording conditions and tasks. The corpus consists mainly of three language portions: German, American English and Japanese. The German VM portion contains sufficient speech data for training and testing (35136 turns[1]). For this study only the German portion was used. The database-scenario deals with scheduling appointments with a business partner: real-life-situations with currently used speech. The "formal situation" setup ensures that speech contains fewer and weaker regional variants than it would contain if personal affairs were discussed.

The training (TRAIN), development (DEV) and test (TEST) sets currently used in our experiments on the VM corpus were created with the following constraints (see table 1 for exact numbers): each speaker is allowed in only one set (hard constraint), for each speaker there must be at least one complete dialogue (to allow speaker adaptation algorithms to be applied; hard constraint), speakers should be distributed equally across sexes in all sets (soft constraint), recordings should be distributed equally across recording sites in all sets (to cover possible accents preferences in one site; soft constraint).

The HMM system uses the full data for training and testing. The LSTM classification network uses only one fourth of the training set in its baseline training, another fourth of the training set is used for retraining. The full test set is used as described above.

4 Experimental Setup

4.1 HMM System for Evaluating the Task Difficulty

The HMM[12] phone recognizer was built up with the Hidden Markov Toolkit [18]. It uses the above defined subsets and a bigram trained solely on the training corpus ($VM1\_TRAIN + VM2\_TRAIN$). It was tested on the $VM1\_DEV + VM2\_DEV$ sets with the corresponding lexicon (total: 5540 lexical entries).

The acoustic models are based on 12 Standard MFCC + Energy + velocity + acceleration (39), Diagonal covariance matrices, 3-5 states per

[1] One turn in the VM database has about 22.8 words in average.

phoneme, 43 phoneme classes (extended German SAMPA) + garbage + voice garbage + silence + laugh + breath (48), Models initialized using the Munich Automatic Segmentation (MAU) tier of the BAS Partiture Format (BPF) from 1/4 of TRAIN, Re-estimation and splitting mixtures after 6 iterations on total TRAIN, testing after every two iterations on DEV, weight of language model fixed to 6.5; beam search width 100.0.

4.2 BLSTM: Experimental Setup

Preliminary experiments with LSTM standard nets with 25, 50, 100 and 200 blocks (2 cells each) showed that although the duration of the epochs doubled each time, comparable results occurred in far fewer epochs. Nevertheless all experiments converged at around 50% framewise phoneme correctness. When comparing LSTM bidirectional nets to standard nets with comparable weights (50 000) we found that BLSTM needs less epochs to obtain comparable results to standard nets and reaches higher framewise phoneme correctness (58.87%). Both bidirectional and standard nets reach their peak around the 120th epoch.

Based on these findings we used a two-step retraining procedure as follows: LSTM training and retraining sets were each around 1/4 of the whole VM training set. Both training and retraining set are distinct from each other but were randomly chosen from the whole training set. The whole VM test set was used.

Our bidirectional LSTM network contained two hidden LSTM layers (for the forward and reverse nets), each with 200 blocks of 2 cells. It had 26 input nodes and a softmax output layer containing 52 nodes. A cross entropy objective function was used. The input layer was connected to the hidden layers, both of which were connected to themselves and to the output layer. There were 907112 weights in total. Note that unlike HMMs BLSTM has no structural bias and more weights - a disadvantage according to the bias-variance dilemma [5].

5 Experimental Results

Our experiments are divided into two main parts: The first shows the recognition results of a plain HMM phone recognizer which was trained both on monophones and triphones (also across words). Part two gives the plain LSTM classification for frame by frame recognition results.

Monophones contain 512 Gaussian mixtures per state. Triphones have the same number of parameters as the monophone system, 8 mixtures per state and are trained also across word boundaries. HMMs were trained on the full training

Table 2. For comparison: Phoneme error rate for plain HMMs

System	training set size	phoneme error rate on the test set	epochs
Monophone	full	34.29%	52
Triphone crossword	full	35.49%	37

Table 3. Recognition results: frame by frame phoneme error rate for plain BLSTM

System	training set size	frame by frame phoneme error rate on the test set	epochs
baseline	1/4 (randomly chosen)	38.40%	50
retraining	1/4(distinct from baseline)	33.36%	67

set (BLSTM just on one fourth, retraining on another fourth). Both systems use the same test set. Table 3 shows the main results of the plain BLSTM net.

BLSTM retraining led to a 5% improvement on the full test set. Using 1/4 of the training set at a time greatly reduces total training time.

6 Predicting the Phoneme Error Rate: An Extrapolation for a HMM-LSTM Hybrid Approach

Although we cannot compare the framewise phoneme error of BLSTM directly with the phoneme error of the HMM we expect that a BLSTM-HMM hybrid (under construction) will outperform both plain BLSTM on frame by frame and plain HMMs on the phoneme level, inheriting the best of both worlds, namely reduction of training material and training time (BLSTM), as well as more built-in structural bias (HMMs). This expectation is encouraged by experiments on **read speech** by Chen and Jamieson [3], Shire [14], Waterhouse, Kershaw and Robinson [16], and Elenius and Blomberg [4]. They all achieved better results on the phoneme level using an ANN-HMM hybrid approach, as shown in table 4 for framewise and phoneme error rates for several systems on various corpora. *improvement factor* shows the relative ratio of framewise and phoneme error. *LIN* stands for Linear Input Network, *MLIN* for Mixtures of LINs for adaptation (*_2* = 2 experts; *_4* = 4 experts). *MLP* stands for Multilayer Perceptron nets[2].

As can be seen from table 4 the framewise errors are quite high for noisy input sequences (several microphones or enriched with background noise) as opposed to clean speech. The HMM part of the hybrids is able to drastically reduce the error on the phoneme level due to structural bias of the HMM. This means that on unprompted speech with background noise, speaker overlaps and other perturbations we can expect a much lower phoneme error.

With the **worst** improvement factor (1.15) of table 4 we can conservatively predict a phoneme error rate of 29.01% for a retrained BLSTM-HMM hybrid on VERBMOBIL (33.39% for the standard BLSTM respectively). An optimistic

[2] MLPs are supervised feedforward neural networks trained with the standard back-propagation algorithm. With one or two hidden layers, they can approximate virtually any input-output(= the desired response) map. They are widely used for pattern classification and can approximate the performance of optimal statistical classifiers in difficult problems.

[3] ARPA 1995 H3 multiple unknown microphones.

[4] NUMBERS95.

Table 4. Framewise and phoneme errors on read speech corpora

System	corpus	frame (plain ANNs)	phoneme (ANN- HMM hybrids)	improvement factor
Backprop [5]	Swedish speakers	30.0%	24.5%	1.22
RNN 0 pass [15]	MUM[2] Task	22.8%	18.1%	1.26
LIN 1 pass [15]	MUM Task	20.1%	16.5%	1.22
LIN 2 pass [15]	MUM Task	19.9%	15.9%	1.25
MLIN_2 1 pass [15]	MUM Task	19.2%	16.5%	1.16
MLIN_2 2 pass [15]	MUM Task	18.9%	16.1%	1.17
MLIN_4 2 pass [15]	MUM Task	18.2%	15.8%	1.15
MLIN_4 3 pass [15]	MUM Task	18.0%	15.7%	1.15
MLP [14]	clean speech[4]	28.97%	7.3%	3.97
MLP [14]	clean sp. no border[4]	29.80%	7.7%	3.87
MLP [14]	factory noise[3]	42.84%	15.5%	2.76
MLP [14]	factory noise no border[4]	42.88%	15.0%	2.86
RNN [3]	TIMIT	26.3%	20.21%	1.30

calculation with the **best** improvement factor (3.97) for read speech in table 4 would give us 8.4% for the retrained BLSTM-HMM hybrid (9.67% for the baseline respectively). Of course, to figure out the precise improvement we really have to implement a BLSTM-HMM hybrid.

7 Conclusions and Outlook

We examined the retraining ability of LSTM recurrent nets in a frame by frame phoneme classification task of unprompted speech. We compared recognition results of a normally trained BLSTM system to those of a retrained one. Retraining both significantly reduced both time costs and training set size and improved recognition results. An extrapolation based on previous work on read speech [16,3,14,4] promises significant additional improvements on the phoneme level through a BLSTM-HMM hybrid, which we are currently implementing.

Acknowledgements

This work was supported by the SNF under grant number 200020-100249.

References

1. Baldi, Brunak, Frasconi, Soda, and Pollastri. Exploiting the past and the future in protein secondary structure prediction. *BIOINF: Bioinformatics*, 15, 1999.
2. Jinmiao Chen and Narendra S. Chaudhari. Capturing long-term dependencies for protein secondary structure prediction. In Fuliang Yin, Jun Wang, and Chengan Guo, editors, *Advances in Neural Networks - ISNN 2004, International Symposiumon Neural Networks, Part II*, volume 3174 of *Lecture Notes in Computer Science*, pages 494–500, Dalian, China, 2004. Springer.

3. R. Chen and L. Jamieson. Experiments on the impementation of recurrent neural networks for speech phone recognition. *Proc. Thirtieth Annual Asilomar Conference on Signals, Systems and Computers*, pages 779–782, 1996.
4. K. Elenius and M. Blomberg. Comparing phoneme and feature based speech recognition using artificial neural networks. *Proc. ICSLP*, 1992.
5. S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4:1–58, 1992.
6. F. A. Gers and J. Schmidhuber. Long Short-Term Memory learns simple context free and context sensitive languages. *Proc. IEEE TNN*, 2001.
7. A. Graves, D. Eck, N. Beringer, and J. Schmidhuber. Biologically plausible speech recognition with LSTM neural nets. *Proc. Bio-ADIT*, 2004.
8. Alex Graves, Nicole Beringer, and Juergen Schmidhuber. Rapid retraining on speech data with lstm recurrent networks. Technical Report IDSIA-05-05, IDSIA, www.idsia.ch/techrep.html, 2005.
9. Alex Graves and Juergen Schmidhuber. Framewise phoneme classification with bidirectional lstm networks. In *International Joint Conference on Neural Networks, July-August 2005, under review*, 2005. Currently under review.
10. S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.
11. John McDonough and Alex Waibel. Performance comparisons of all-pass transform adaption with maximum likelihood linear regression. *Proc. ICSLP*, 2004.
12. L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. 77(2):257–286, 1989.
13. Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45:2673–2681, November 1997.
14. M. Shire. Relating frame accuracy with word error in hybrid ann-hmm asr. *Proc. EUROSPEECH*, 2001.
15. W. Wahlster. SmartKom: Symmetric multimodality in an adaptive and reusable dialogue shell. *Krahl, R., Günther, D. (eds): Proceedings of the Human Computer Interaction Status Conference*, 2003.
16. S. Waterhouse, D. Kershaw, and T. Robinson. Smoothed local adaptation of connectionist systems. *Proc. ICSLP*, 1996.
17. K. Weilhammer, F. Schiel, and U. Reichel. Multi-Tier annotations in the Verbmobil corpus. *Proc. LREC*, 2002.
18. S. Young. *The HTK Book*. Cambridge University Press, 1995.

Temporal Sound Processing by Cochlear Nucleus Octopus Neurons

Werner Hemmert, Marcus Holmberg, and Ulrich Ramacher

Infineon Technologies Inc., Corporate Research ST, 81730 Munich, Germany
{Werner.Hemmert, Marcus.Holmberg, Ulrich.Ramacher}@infineon.com

Abstract. The human auditory system excels in the detection of signals in background noise. We evaluate the principles of robust processing with a detailed inner ear model and a model of octopus neurons in the cochlear nucleus. These neurons reject steady-state excitation and fire on signal onsets with extremely high temporal precision. Spike-triggered reverse-correlation analysis revealed that octopus neurons fire preferentially if many coincident spikes follow a short interval of relative low excitation. The frequency spectrum of the reverse-correlation revealed that octopus neurons perform a band-pass analysis of the incoming signal, with the pass-band ranging from about 110 to 650 Hz. The low-frequency slope was approximately 6 dB/oct, which indicates that octopus neurons process the first derivative of the input signal. This mechanism not only removes steady-state activity, which accentuates onsets, but also enhances amplitude modulation in the frequency region predominant in speech.

1 Introduction

The spike-trains of auditory nerve fibers code spectral–, temporal– and spatial information, which is extracted at higher levels in the auditory pathway. The way information is coded and processed is crucial for speech recognition, especially in noisy environments. Octopus neurons are located in the cochlear nucleus, the first neuronal processing stage, and receive inputs from auditory nerve fibers (ANF). Unlike most other neurons, which exhibit sustained activity to continuous excitation, octopus neurons show onset inhibitory responses: they only fire on signal onsets. As they receive predominantly excitatory inputs, their membrane properties are thought to be responsible for suppressing sustained responses [1],[6],[8]. Octopus neurons are interesting because they suppress spontaneous- and uniform activity, a feature which is essential for sound processing in noise. Their detection of signal onsets is important for sound localization as well as for the perception of speech and music.

2 Modeling

In this paper we combine our realistic inner ear model [2], which codes sound signals into spike trains of the auditory nerve, with a model of octopus neurons.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 583–588, 2005.

2.1 Peripheral Sound Processing

The model of the peripheral hearing system consists of a simplified middle ear, a model of inner ear hydrodynamics followed by a compression stage, and an inner hair cell model. Here we describe the model very briefly, more details can be found in [2]. BM vibrations were calculated with a computationally efficient wave-digital filter model comprising of 100 sections at a sampling rate of 48 kHz. The inner ear's nonlinear amplification process was realized by adding four second-order resonators with time-varying quality factors at the output of each section. Quality factors were high at low levels to achieve amplification and were decreased at high levels to reach dynamic range compression. The sensory cells were modeled according to [7]. All together the model replicated bandwidths of human threshold tuning curves [4] and latest measurements of dynamic range compression [5] with great precision [2].

2.2 Octopus Neuron Model

Auditory nerve fibers project to neurons in the cochlear nucleus (CN). Here we focus on the function of so-called onset inhibitory units, which were identified as octopus neurons. They receive excitatory inputs from approximately 60 auditory nerve fibers and they fire if about 10% to 25% of their inputs are activated within 1 ms [1]. In this study, we only used outputs trains from high spontaneous rate (HSR) ANFs.

We used a single-compartment model with Hodgkin-Huxley-type ion channels. Rothman and Manis measured the properties of the five major conductances [6] and derived both steady-state and dynamic equations. We solved the differential equations of the ionic channels by replacing them with difference equations and iterating in the time domain. We used conductance values and time constants corrected for a body temperature of 38°C. Octopus neurons – and our model – exhibit large activated ionic conductances at rest of about 40 nS and a membrane capacitance of 12 pF. Due to their extraordinarily fast membrane time constant (0.3 ms) they act as coincidence detectors and they greatly enhance the precision of timing relative to a single ANF. Their electrical behavior is dominated by a low-threshold potassium channel (K_{LT}) with activation kinetics in the order of 2 ms, which is already activated at rest [8]. When their membrane is depolarized, they elicit an initial action potential, but thereafter K_{LT} compensates input currents and keeps the membrane potential below spiking threshold. For octopus neurons, action potentials of the auditory nerve elicit only extremely brief activation of postsynaptic currents (compare [6]).

In this study we connected 60 auditory nerve fibers originating from a single frequency channel (characteristic frequency: 1.4 kHz) of our model with an octopus neuron. We used only excitatory synapses. Activation by an action potential of the auditory nerve was modelled with a single exponentially decaying conductance (0.1 ms decay-time constant, [6]). We set the peak value to 8.5 nS; an action potential was elicited when 25% of the input fibers fired synchronously; a spike of an octopus neuron was counted when its membrane potential crossed 0 mV.

3 Results

The firing rate of auditory nerve fibers covers only a very small range: high spontaneous rate fibers for example exhibit sustained firing rates ranging from 30 to about 300 spikes/s. Sound signals, on the other hand, can cover a huge dynamic range of more than 6 orders of magnitude! To overcome this discrepancy, dynamic compression is a key function performed by our inner ear. Our model provides large dynamic compression of more than 60 dB, the growth function follows approximately a fourth root law at medium signal levels, replicating recent measurements [5]. Fig. 1 shows the coding of a 1.5 kHz pure tone with stepwise increasing amplitude; sound pressure levels were 40, 60 and 80 dB (RMS). The top trace shows the signal. Note that signal amplitude at 40 dB (starting at $t = 0$ s) is a factor of 100 smaller than at 80 dB.The reaction of an ANF with a characteristic frequency of 1.4 kHz, slightly lower than the test tone, is displayed in the middle trace. Before a signal is applied ($t < 0$ s), the ANF fires with its spontaneous rate of approximately 30 spikes/s. When the signal is switched on, the ANF reacts with a sharp rise of its firing rate which decays again to a sustained rate of approximately 120 spikes/s. For a tenfold increase in stimulus amplitude (60 dB, $t = 50$ ms) another transient is generated and the spontaneous rate increases to 210 spikes/s. For yet another tenfold amplitude increment the sustained firing rate hardly increases any more (220 spikes/s); HSR fibers saturate about 40 dB above threshold. Still, the fiber is able to generate a transient onset response. In Fig. 1d the response for the step from 60 to 80 dB is shown with high temporal resolution. The stimulus changes to 80 dB at $t = 100$ ms, the change in ANF is delayed in the inner ear and occurs after 103 ms. The

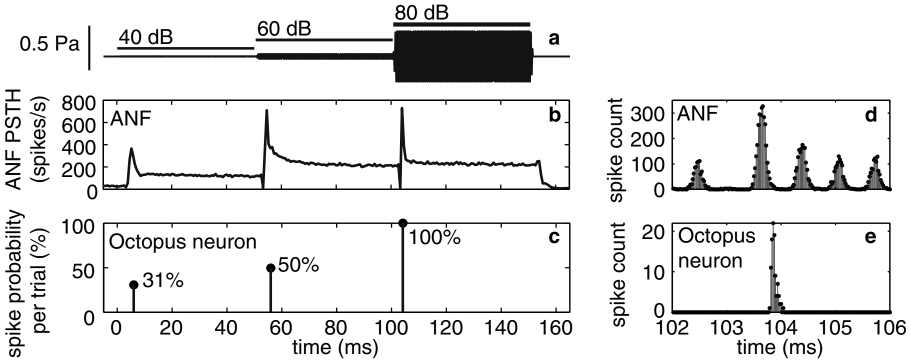


Fig. 1. Onset processing of octopus neurons. (a) sound stimulus is a 1.5 kHz pure tone with stepwise increasing amplitude (rise time: 1 ms). (b) Poststimulus-time histogram (PSTH) of a single ANF. (c) octopus neuron activity per trial. In the left column (b+c), averaged spike counts are collected in 0.66 ms time bins (1 cycle), in the right column (d+e), raw spike counts are displayed for each sampling time (21 μs). Spike counts are from 60 ANFs innervating a single octopus neuron summed over 100 trials.

responses are phase-locked, ANF activity is limited to the negative half of the stimulus cycle (rarefaction).

3.1 Temporal Processing of Onsets Neurons

For high-frequency sounds, octopus neurons fire only at signal onset, whereas for frequencies up to about 800 Hz, they can fire at every cycle. Fig. 1c shows the reaction of an octopus neuron to a tone with stepwise increasing amplitude. At the onset of the 40 dB tone, the neuron fires with 31% probability. A classical integrate-and-fire neuron would fire during the whole period of the tone burst - not so octopus neurons. Excitatory input currents are shunted during continuous signals which prevents further action potentials. This mechanism still works when the signal amplitude is increased 10-fold (even two times). When the signal level increased from 40 to 60 dB, the neuron fired with a probability of 50%, for the increase to 80 dB an action potential was always (100 trials) elicited. Thereby the temporal precision of the action potentials are remarkable: whereas ANFs fire for the whole half stimulus cycle, 90% of the action potentials of the octopus neuron were in an interval of less than 150 μs (Fig. 1e). This precision is reached by coincidence detection of at least four synchronously firing ANFs and by the fast membrane time constant of octopus neurons.

The spike-triggered reverse-correlation technique revealed the average temporal excitation pattern which most likely causes the octopus neuron to fire [8]. For its calculation, spontaneous activity of ANFs provided random input. The octopus neuron model responded to spontaneous ANF activity only extremely sparsely. To provide more synchronous activity, the traces of 20 ANFs were presented three times (to yield a total of 60 inputs), and, to mimic higher firing rate, the input conductance was increased by a factor of four (corresponding to a firing rate of a single ANF of 120 spikes/s). To calculate the spike-triggered reverse-correlation, ANF spikes were averaged (0.25 ms time bins) around time windows, where the octopus neuron fired (Fig. 2a). About 10 ms before and almost immediately after the octopus neuron elicited a spike, the reverse correlation relaxes to the spontaneous rate (1800 spikes/s for 60 input fibers). To trigger a spike, highly synchronous input activity was required, which was more than a factor of 4 higher than spontaneous activity. Moreover, depressed activity about 1–3 ms in

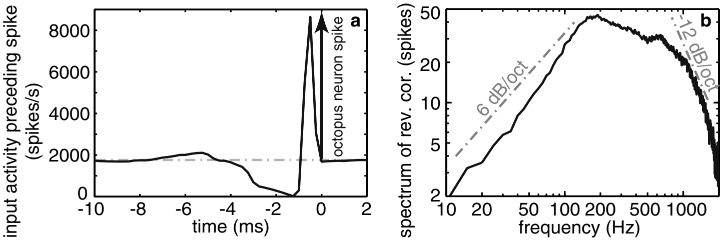


Fig. 2. Spike-triggered reverse-correlation and its frequency transformation

advance facilitates the generation of an action potential. The reverse correlation can be interpreted in a similar way as the impulse response of a linear filter. Its spectrum revealed that temporal processing of octopus neurons very much resembles a band-pass filter. The low-frequency slope was close to 6 dB/oct, the pass-band (-3 dB) reached from 110 Hz to about 650 Hz.[1]

This finding indicates that octopus neurons might also play a role in the processing of amplitude modulated (AM) sounds. We therefore tested the the reaction of ANFs and octopus neurons to AM tones (Fig. 3). The firing rate of ANFs follow the envelope of the AM tone. Octopus neurons decode the AM by a single action potential on a distinct phase of the AM signal. The firing probability plotted as a function of AM frequency (Fig. 3d) also exhibits a band-pass characteristic, similar to Fig. 2b.

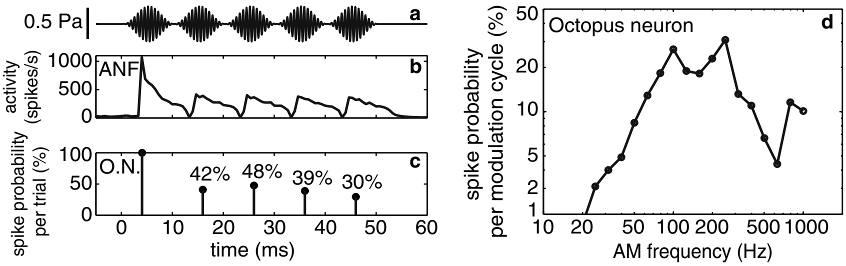


Fig. 3. Processing of amplitude modulated tones. (a) sound stimulus (1.5 kHz tone, amplitude modulated at 100 Hz, 100% modulation), (b) PSTH of a single ANF (0.66 ms bins) and (c) octopus neuron activity (O.N.; numbers indicate firing probabilities). (d) Octopus neuron spike probability per modulation cycle (the first cycle, where the octopus neuron always fired was omitted) as a function of amplitude modulation frequency. The plot displays data from 9 AM periods with 100 repetitions per frequency point.

4 Conclusion

Auditory nerve fibers of the auditory system code both spectral- and temporal properties of sound. Temporal precision is greatly enhanced by octopus neurons, which require synchronous firing of multiple ANFs within a brief (<1 ms) period. Octopus neurons play a major role in sound localization, they greatly enhance temporal precision and fire to onsets with a jitter of less than 150 μs. They suppress continuous activity over a wide amplitude range and still fire at signal onsets. The spectrum of the spike-triggered reverse correlation shows the band-pass behaviour of octopus neurons. Thereby, from a signal processing point of view, the 6 dB/oct low-frequency slope indicates that octopus neurons perform a

[1] The spike-triggered reverse-correlation of a leaky integrate-and-fire-neuron would be an exponential function, its frequency response a low-pass filter.

temporal differentiation of their input. The areas below- and above spontaneous activity in the reverse-correlation functions are almost equal, which is required for a large suppression of continuous input activity. Note that spontaneous activity of the input provides an offset, which is essential to perform high-pass filtering with its bimodal impulse response – as negative spike rates do not exist. The bandpass characteristics of the octopus neurons is also reflected in the processing of amplitude modulated sounds. Whereas the low-frequency slope of the spectrum of the reverse correlation function and of the response to AM modulated tones is very similar, its high frequency part is also influenced by other factors, for example the filtering properties of the inner ear. Notably, the high sensitivity of octopus neurons to AM-frequencies above 100 Hz coincides with the fundamental frequency of speech sounds – it is therefore likely that octopus neurons also play an important role processing voiced phonemes. Onset-type neurons are found at various levels of neuronal processing; we hypothesize that their suppression of steady-state input activity provides a key mechanism to extract speech signals in noise.

Acknowledgements

This work was funded by the German Federal Ministry of Education and Research (reference number 01GQ0443).

References

1. Ferragamo, M. J., Oertel, D.: Octopus cells of the mammalian ventral cochlear nucleus sense the rate of depolarization. J. Neurophysiol. **87** (2002) 2262–2270.
2. Holmberg, M., Hemmert, W.: An auditory model for coding speech into nerve-action potentials. in: Proceedings of the Joint Congress CFA/DAGA (2004) 773-4.
3. Meddis, R., O'Mard, L. P., Lopez-Poveda, E. A.: A computational algorithm for computing nonlinear auditory frequency selectivity. *J. Acoust. Soc. Am.* **109** (2001) 2852–2861.
4. Oxenham, A. J. Shera, C. A.: Estimates of human cochlear tuning at low levels using forward and simultaneous masking. *J. Assoc. Res. Otolaryngol.* **4** (2003) 541–54.
5. Lopez-Poveda, E. A., Plack, C. J., Meddis, R.: Cochlear nonlinearity between 500 and 8000 Hz in listeners with normal hearing. *J. Acoust. Soc. Am.* **113** (2003) 951–960.
6. Rothman, J. S., Manis, P. B.: The roles potassium currents play in regulating the electrical activity of ventral cochlear nucleus neurons. J. Neurophys. **89** (2003) 3097–3113.
7. Sumner, C. J., Lopez-Poveda, E. A., O'Mard, L. P., Meddis, R.: A revised model of the inner-hair cell and auditory-nerve complex. J. Acoust. Soc. Amer. **111** (2002) 2178–2188.
8. Svirskis, G., Kotak, V., Sanes, D. H., Rinzel, J.: Sodium Along With Low-Threshold Potassium Currents Enhance Coincidence Detection of Subthreshold Noisy Signals in MSO Neurons. J. Neurophysiol. **91** (2004) 2465–2473.

A SOM Based 2500 - Isolated - Farsi - Word Speech Recognizer

Jalil Shirazi[1] and M.B. Menhaj[2]

[1] Electrical Engineering Dep, Gonabad Azad University, Gonabad, Iran
jshshr2002@yahoo.com
[2] Electrical Engineering Dep, Amir Kabir University of Technology, Tehran, Iran
mbmenhaj@yahoo.com

Abstract. A modified 2-D Kohonen Self-Organizing (MSOM) neural network is used for recognizing Farsi isolated words. The network dimension is 10*15 cells with a hexagonal topology and it is trained using 300 Farsi words. As input vectors for learning, speech spectrum and energy of signal are used. The weight vectors of the cells are then fine tuned using supervised learning vector quantization 3 (LVQ3) technique. The cells are labeled to 28 out of 29 Farsi phonemes. At the word recognition stage, the quasi phonemes are obtained. Then the phonemes are determined. Using the phonetic rules of Farsi words and the connection rules of Farsi characters, the recognized word will appear on the monitor. To remedy the errors, a 2500 word dictionary is used. The determined sequence of phonemes is given to the dictionary, and the closest word to the sequence is shown on the monitor. The proposed recognizer is able to recognize all vowels with the accuracy of 100 percent, and it also recognize correctly 55 isolated words among 100 words.

1 Introduction

Automatic recognition of speech belongs to the broad category of pattern recognition tasks. Speech recognition is difficult, because human beings' recognition of speech consist of many tasks, ranging from detection of phonemes from speech waveforms to high level understanding of messages. What we believe we hear, we in fact reconstruct in our minds from pieces of received information. Speech elements are not unique at all. Distributions of the spectral samples of different phonemes overlap. A phoneme's acoustic spectrum varies in the context of different phonemes. The same phonemes spoken by different persons can be confused too, for example, /m/ of one speaker might sounds like /l/ of another. Even the same speaker is not able to produce exactly the same acoustic signal for the same utterance [1].

Recognition of the speech of arbitrary speakers is much more difficult than generally believed. The difficulties would be even greater if the vocabularies were unlimited. it seems that a speech recognizer with large vocabulary must recognize phonemes. Neural phonetic typewriter for unlimited continuous speech using the self-organizing map has been built for Finish and Japanese [1], [2]. In this paper, phoneme recognition is implemented for recognition a speaker dependent of large vocabulary Farsi words. In this method a two dimension Kohonen Self-Organizing neural

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 589 – 595, 2005.

network has been used for Farsi phonemes. After training the network with feature vectors of Farsi words, cells are labeled by Farsi phonemes. After implementing the network for recognizing six vowels: /a, @, e, o, u and i/, the phoneme set has been increased to 28 Farsi phonemes and finally a 2500 word dictionary has been used for error correction and orthography.

For recognition of one spoken word, first it's feature vectors is extracted, then it's quasi phonemes are obtained by the network. Phonemes are determined by using dynamic window lengths. Using Farsi rules, a few errors are corrected. The resulted final phonemes sequence is compared to a dictionary and the nearest word is obtained from the dictionary. In the second section of this paper, Kohonen Self-Organizing map is reviewed. In sections 3 and 4 speech processing and network learning are given. Section 5 presents word recognition, section 6 dictionary and finally section 7 conclude the paper.

2 The Sofm

The Self-Organizing feature map (SOFM) is a set of weight vectors that are logically associated with a lattice of display cells. The cells are tuned to input signals through an unsupervised learning process. During training, the weight vectors are not updated independently but as topologically related subsets. The selection of subset updating cells at each learning step requires the definition of a center (winner) cell and also a topological neighborhood around it. The update neighborhood radius is large initially and shrinks monotonically with time. SOFM attempts to find topological structure in the input data and displays it in one or two dimensions [3], [4], [5]. The SOFM can be used for vector quantization. Cells should be grouped into subsets which correspond to discrete classes. In this model the neighborhood N_c is defined for lateral interaction. At each learning step, all of the cells whit in N_c are updated, whereas cells outside N_c are left intact. This neighborhood is centered around the cell which the best match with input x is found:

$$\|x - m_c\| = min_i \{\|x - m_i\|\} \tag{1}$$

The updating process is:

$$\begin{aligned} m_i(t+1) &= m_i(t) + \alpha(t)[x(t) - m_i(t)] && if\ i \in N_c(t) \\ m_i(t+1) &= m_i(t) && if\ i \notin N_c(t) \end{aligned} \tag{2}$$

Where $\alpha(t)$ is the adaptation gain (0 < $\alpha(t)$ < 1) and should decrease with time. A biological lateral interaction often has the shape of a bell curve. Denoting the coordinates of cells c and i by the vectors r_c and r_i respectively, a proper form for $\alpha(t)$ might be:

$$\alpha(t) = \alpha_0(t) exp\left(-\|r_i - r_c\|^2 / \delta(t)^2\right) \tag{3}$$

with $\alpha_0(t)$ *and* $\delta(t)$ as suitable decreasing functions of time [2], [5], [6].

The process may be started by choosing arbitrary, even random, initial values for them $\boldsymbol{m}_i(0)$. The only restriction is that they should be different. In this map, several codebook vectors $\boldsymbol{m}_i$ may assigned to each class, and each of them is labeled with the corresponding class symbol. If the map is used as a classifier, after unsupervised learning, to improve classification accuracy, $\boldsymbol{m}_i$ should be updated. The map must be trained to function as a classifier using a training set of feature vectors whose classification is already known. Kohonen proposed learning scheme for fine tuning known as LVQ1, LVQ2 and LVQ3 algorithms [2]. Kohonen's LVQ1, LVQ2 and LVQ3 algorithms are supervised since their application requires a labeled data set.

3 Acoustic Preprocessing

To learn the map by feature vectors of speech signal, feature vectors with 16 components has been used. Acoustic preprocessor of our system consists of the following stages:

(1) 16-bit analog-to-digital converter with 12.8-KHZ sampling rate.
(2) 256 point FFT, which is computed every 20ms using a 256- point window.
(3) The logarithmic energy is computed for the whole frame.
(4) The logarithmic energy which is computed on 12 filters with equal bandwidth in 100 – 3000 HZ.
(5) The logarithmic energy which is computed on 3 filters with equal bandwidth in 3000 – 5000 HZ.
(6) The average is subtracted from all of the 15 components.
(7) 15 components are normalized to 100.

We have tested other speech features for the network like LPC coefficients, cepstral coefficients and filter bank coefficients [8], [9]. But FFT reflected its clustering properties better than others.

4 Network Learning

We use Kohonen map for phoneme recognition. The network dimension is 10*15 cells with hexagonal topology. We have tested rectangular topology, but hexagonal topology performance was better. We have trained the network by different $\alpha(t)$ functions, and got the best result in clustering and performance with:

$$\alpha(t) = \alpha_0(t) exp\left(-\|r_i - r_c\|^2 / \delta(t)^2\right)$$

$$\left.\begin{aligned} \alpha_0(t) &= 0.05\,(1 - t/1000) \\ \delta(t) &= 2\,(7 - 6t/1000)^2 \end{aligned}\right\} \qquad 0 \le t \le 1000$$

$$\tag{4}$$

$$\left.\begin{aligned} \alpha_0(t) &= 0.02\,(1 - t/(N - 1000)) \\ \delta(t) &= 2\,(3 - 2t/(N - 1000))^2 \end{aligned}\right\} \qquad 1000 \le t \le N$$

N is the total learning times. With using these functions, there is no neighborhood and all weight vectors are updated at each step. We have used a modified SOM known MSOM to minimize dead cells.

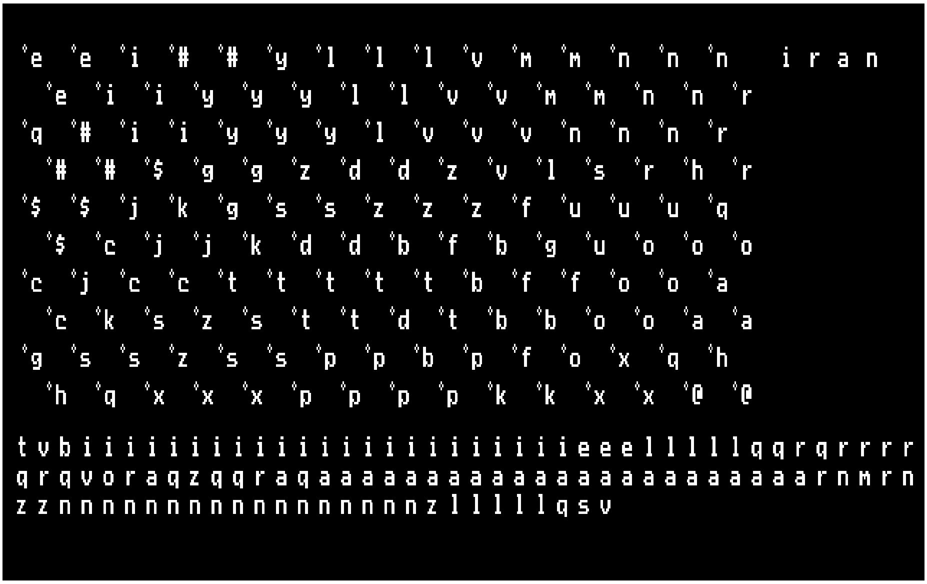
The MSOM consists of the following stages:

(1) Random selection of an input vector.
(2) Finding winner cell.
(3) Training the network.
(4) Repeating sequences 1, 2 and 3 iteratively.
(5) Determining lazy neurons.
(6) Random re initialization lazy neurons weight vectors.
(7) Returning to step1.

There are 23 consonant phonemes in Farsi. They are written in FARSDAT database: /b, p, t, d, s, z, c, g, k, j, q, ', r, $, x, #, v, f, h, m, n, l and y /. They are written in Farsi as: /ب, پ, ت, د, س, ز, چ, گ, ك, ج, غ, ع, ر, ش, خ, ژ, و, ف, ح, م, ن, ل, ي/. Also there are 6 vowel phonemes in Farsi: /i, a, e, u, o and @/. They are written in Farsi as: /اي, آ, اِ, او, اُ and اَ / [7].

We have trained network 200 to 300 words 100000 iterations. After training, network was labeled by different phonemes. It was done by giving 180 vectors for each phoneme with known classification, and assigning the cells to different phonemes by using majority voting, according to the frequency with which each $\boldsymbol{m}_i$ is closest to the calibration vectors of a particular class.

Recognition of discrete phonemes is a decision-making process in which the final accuracy depends on the rate of misclassification errors. It is therefore necessary to



```
◊e  ◊e  ◊i  ◊#  ◊#  ◊y  ◊l  ◊l  ◊l  ◊v  ◊m  ◊m  ◊n  ◊n  ◊n    i r a n
  ◊e  ◊i  ◊i  ◊y  ◊y  ◊y  ◊l  ◊l  ◊v  ◊v  ◊m  ◊m  ◊n  ◊n  ◊r
◊q  ◊#  ◊i  ◊i  ◊y  ◊y  ◊y  ◊l  ◊v  ◊v  ◊v  ◊n  ◊n  ◊n  ◊r
  ◊#  ◊#  ◊$  ◊g  ◊g  ◊z  ◊d  ◊d  ◊z  ◊v  ◊l  ◊s  ◊r  ◊h  ◊r
◊$  ◊$  ◊j  ◊k  ◊g  ◊s  ◊s  ◊z  ◊z  ◊z  ◊f  ◊u  ◊u  ◊u  ◊q
  ◊$  ◊c  ◊j  ◊j  ◊k  ◊d  ◊d  ◊b  ◊f  ◊b  ◊g  ◊u  ◊o  ◊o  ◊o
◊c  ◊j  ◊c  ◊c  ◊t  ◊t  ◊t  ◊t  ◊t  ◊b  ◊f  ◊f  ◊o  ◊o  ◊a
  ◊c  ◊k  ◊s  ◊z  ◊s  ◊t  ◊t  ◊d  ◊t  ◊b  ◊b  ◊o  ◊o  ◊a  ◊a
◊g  ◊s  ◊s  ◊z  ◊s  ◊s  ◊p  ◊p  ◊b  ◊p  ◊f  ◊o  ◊x  ◊q  ◊h
  ◊h  ◊q  ◊x  ◊x  ◊x  ◊p  ◊p  ◊p  ◊p  ◊k  ◊k  ◊x  ◊x  ◊@  ◊@

t v b i i i i i i i i i i i i i i i i i i i i i i i e e e l l l l l q q r q r r r r
q r q v o r a q z q q r a q a a a a a a a a a a a a a a a a a a a a a a a r n m r n
z z n n n n n n n n n n n n n n n n n z l l l l l q s v
```

Fig. 1. The labeled network, quasi phonemes and phonemes for word /iran/

try to minimize the error by using supervised learning scheme, using a training set of speech spectra with known classes. We have used supervised learning LVQ3 with 40000 iterations for fine tuning the weight vectors. We tested LVQ1, LVQ2 and K-means [3], [4], [6] for fine tuning, but the LVQ3 performance was the best.

Figure 1 is an example of phonemes map and shows the map when its neurons are labeled according to the majority voting to 28 out of 29 Farsi phonemes.

5 Word Recognition

Word recognition of one spoken word consists of following stages:

(1) Computing feature vectors.
(2) Finding nearest labeled cell to each vector by network (quasi phonemes).
(3) Determining phonemes from the quasi phonemes.
(4) Correcting some errors occurred in phonemes.
(5) Transferring phonemes to the dictionary to obtain nearest word.

Consider that the feature vectors are taken at regular intervals every 10 ms, and they are first labeled in accordance with nearest cells. These labeled are called quasi phonemes. In contrast, the duration of true phoneme is variable, from 40 to 400 ms. We have used heuristic rules for segmentation of quasi phoneme sequences into true phonemes. These rules are heuristic completely and relate to phonemes. We have used one slide time domain window, that covers dynamically period of 60 ms to 140 ms. If *m* out of *n* successive quasi phonemes are the same, they correspond to a single phoneme. The parameters *m* and *n* are phoneme and speaker dependent. Figure 1 shows quasi phonemes and phonemes for word /iran/ with phoneme sequences /i, r, a, n/.

6 Dictionary

Because of the coarticulation effects, transformation of the speech spectra due to neighboring phonemes, errors appear in phonemic transcription. So if the classification of speech spectra were without error, the phonemes would not be identifiable from them completely reliable. We have used many rules to correct phoneme map recognizer errors. For example, some Farsi phonemes cannot get connected to make a single word like /l/ and /r/. In such cases we drop one of them.

Phonetic typewriter was implemented for Finnish and Japanese. Both of these languages, like Latin are characterized by the fact that their orthography is almost identical to their phonemic transcription. Farsi orthography is not identical to phonemic transcription: for example, word with phoneme sequences /ل, اِ, م, آ, ك / (/ k, a, m, e ,l /) is written /كامل/. In addition some of Farsi phonemes have many orthographies: for example, phoneme /s/ has 3 orthographies: /س, ص and ث/ .

We used a 2500 word dictionary for solving the above problems and also correcting some error and typing Farsi orthography. The obtained phonemes sequences are given to the dictionary in order to find nearest word. Because the

dictionary is made accessible in memory using software hash coding method, the speed of searching became high. Figure 2 shows the labeled network for Farsi phonemes with Farsi orthography. In addition the word /کامل/ obtained from phoneme sequences:
/ل, اِ, م, آ, اُ, ک / (/k, o, a, m, e , l/) from dictionary.

For more elaboration on the system accuracy, 100 words were pronounced by one person. The designed system could recognize 55 words correctly. The accuracy of this system depends on the type test words and also quality pronunciation as well.

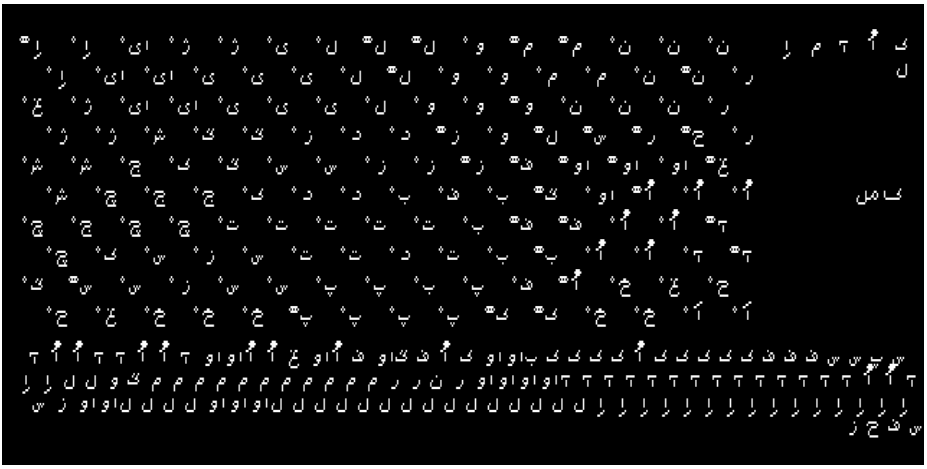


Fig. 2. The labeled network to Farsi phonemes, quasi phonemes, phonemes and obtained dictionary word with Farsi orthography

7 Conclusion

In this paper we tested different topologies for SOFM network, different LVQ for fine tuning and different type speech features. The experimented results show hexagonal for topology, LVQ3 for fine tuning and FFT for speech feature are the best. Heuristic rules are used for segmentation of quasi phonemes to phonemes. First the proposed network is implemented for only six vowels, the accuracy of network was very good and it could recognize all input vowels. By increasing number of phonemes, accuracy of the network was decreased. Generally, the spectral properties of consonants behave more dynamically than vowels. So network misclassification for words with many consonants is more than other words. To remedy coarticulation effects problem, many rules are used. Some of Farsi phonemes have many orthographies and further Farsi orthography is not identical to it's phonemic transcription. A 2500 word dictionary is used for correcting errors and type Farsi words with correct orthography. System accuracy in isolated word recognition became 55 percent. To increase the system accuracy, combination different speech features, automatic segmentation and use of HMM is proposed.

References

[1] T. Kohonen, "The neural phonetic typewriter,"computer, vol 21, no 3, pp. 11-22, March 1988.

[2] T. Kohonen, "The self-organizing map," Proceeding of the IEEE, vol. 78, no 9, pp. 1464-1480, September 1990.

[3] N. R. Pal, J. C. Bezdek, and E. C. K. Tsao, "Generelized clustering networks and Kohonen's self-organizing scheme," IEEE Trans. Neural Networks, vol. 4, pp. 49-558, 1993.

[4] N. B. Kaarayiannis, "Fuzzy Algorithms for Learning Vector Quantization," IEEE Trans. Neural Networks, vol. 7, no. 5, pp. 1196-1211, 1996.

[5] T. Kohonen, Self-Organization and Associative Memory (Third Edition), Berlin: Springer-verlag,1989.

[6] N. B. Karayiannis and M. M. Randolph-Gips, "Soft Learnung Vector Quantization and Clustering Algorithms Based on Non-Euclidean Norm: Multinorm Algorithms," IEEE Transactions on Neural Networks, vol. 14, no. 1, pp. 89-102, 2003.

[7] M. Bijankhan, M.J. Sheikhzadegan, "FARSDAT: The Speech Database of Farsi Spoken Language," Proceeding of Speech Science and Technology Conference, pp. 826-831, Dec 1994.

[8] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding," Proc. IEEE, vol. 73, pp. 1551-1588, 1985.

[9] P. E. Papamichalis. Practical Approaches to speech coding, Englewood Cliffs, NJ: Prentice-Hall, 1987.

Training HMM/ANN Hybrid Speech Recognizers by Probabilistic Sampling

László Tóth and A. Kocsor

Research Group on Artificial Intelligence,
H-6720 Szeged, Aradi vértanúk tere 1., Hungary
{tothl, kocsor}@inf.u-szeged.hu

Abstract. Most machine learning algorithms are sensitive to class imbalances of the training data and tend to behave inaccurately on classes represented by only a few examples. The case of neural nets applied to speech recognition is no exception, but this situation is unusual in the sense that the neural nets here act as posterior probability estimators and not as classifiers. Most remedies designed to handle the class imbalance problem in classification invalidate the proof that justifies the use of neural nets as posterior probability models. In this paper we examine one of these, the training scheme called probabilistic sampling, and show that it is fortunately still applicable. First, we argue that theoretically it makes the net estimate scaled class-conditionals instead of class posteriors, but for the hidden Markov model speech recognition framework it causes no problems, and in fact fits it even better. Second, we will carry out experiments to show the feasibility of this training scheme. In the experiments we create and examine a transition between the conventional and the class-based sampling, knowing that in practice the conditions of the mathematical proofs are unrealistic. The results show that the optimal performance can indeed be attained somewhere in between, and is slightly better than the scores obtained in the traditional way.

1 Introduction

Most machine learning algorithms are prone to inferior performance when the training data is imbalanced, that is when the number of training examples accessible from the various classes is significantly different. In such cases it is frequently observed that the classifier is biased towards predicting the more common classes, performing worse on the rarer classes. Although the exact explanation of this behavior may differ from algorithm to algorithm (see [9] for general reasons), in the hope of an improvement it is always possible to alter the effective class frequencies by presenting more examples from the rarer classes to the learning algorithm. These methods come under the general name of "resampling techniques" [9]. (See the material of the workshops [4] and [5] for more details on techniques proposed to handle class imbalance.)

The class imbalance problem is also present in speech recognition because the natural distribution of speech sounds (phones) is not uniform. However, the solutions proposed by the machine learning community are not necessarily applicable

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 597–603, 2005.

here. This is because most machine learning papers dealing with the topic focus on classification performance, while in speech recognizers the sub-unit models are used as probability estimators. In particular, the so-called "Hidden Markov Model/Artificial Neural Net (HMM/ANN) hybrid recognizers" [2] apply ANNs to estimate the posterior probabilities of the classes. This is made possible by a nice theoretical proof which shows that, under ideal conditions, ANNs estimate the class posteriors [1]. In practice, however, the class imbalance of the training set can lead to inaccurate estimates. A natural idea is to apply the resampling techniques, but these invalidate the proof, so their application is theoretically questionable. In this paper we examine one peculiar resampling method, the "probabilistic sampling" training technique recommended by Lawrence et al. [6], and argue that it is still usable in training ANNs for HMM/ANN hybrids. First, in Section 2 we point out that theoretically it forces the network to estimate scaled class-conditional probabilities instead of class posteriors and this poses no real problem as the recognizer can be easily modified to work with these. Then we show experimentally in Section 3 that when the recognizer is built on a net trained by probabilistic sampling, it yields the same good or slightly better performance than with the conventional training. The paper rounds off with some conclusions and remarks in Section 4.

2 HMM/ANN Hybrids

Several ways of applying ANNs to speech recognition have been proposed (see [7] or [3] for a review), but the most popular of these is the "hybrid HMM/ANN" paradigm of Bourlard et al. [2]. This approach exploits the fact that, under ideal conditions, ANN classifiers approximate the class posteriors. That is, denoting the space of the local feature vectors by X and the set of class labels by C, we can use them to estimate $P(C|X)$. In the hybrid framework the HMM states play the role of the classes of the ANN, and the states usually directly correspond to phone classes. The HMM framework requires the class-conditionals $P(X|C)$, which can be calculated from the posteriors by Bayes' rule as $P(X|C) = P(C|X) \cdot P(X)/P(C)$. From the HMM optimization point of view $P(X)$ is a constant scaling factor and can be ignored. So the HMM/ANN hybrids work with $P(C|X)/P(C)$, which thus gives an estimate of $P(X|C)$ to within a scaling factor. The $P(C|X)$ values are produced by an ANN, and the $P(C)$ values are obtained by a simple frequency counting of the class labels over the training corpus.

3 Probabilistic Sampling

Let us now examine why and when ANNs estimate the class posteriors, and what happens if training is performed by probabilistic sampling. Let us assume that the network has K outputs denoted by y_k ($k = 1, ..., K$), and that it is trained by minimizing the sum-of-squares error [1]. We will also assume that the training

[1] A similar proof exists for the minimum cross-entropy error criterion as well [1].

data is sampled in such a way that its distribution follows the real distribution $P(X)$ of the data points over X. Under these conditions it can be shown that if the size of the training data is allowed to go to infinity, the error function can be written as

$$E = \frac{1}{2}\sum_k \int [y_k(x) - < t_k|x >]^2 \, P(x)dx + B, \tag{1}$$

where B is a constant that is not important here, and $< t_k|x >$ is the conditional average of the target values t_k at x [1]. Obviously, Eq. (1) takes its minimum when $y_k =< t_k|x >$. Now, if the network structure and the labelling of the training data follow the 1-of-K coding scheme (that is t_k takes a value of 1 for the correct class output and 0 for the rest), it is easy to show that $< t_k|x >$ approximates $P(c_k|x)$ (again assuming a representative sampling and an infinite amount of sample data at point x).

Examining Eq. (1) more closely, we see that at any point x of the input space X it is $< t_k|x >$, the local ratio of positive and negative examples from class c_k, that determines the optimal value for y_k. The local errors of these estimates are in turn weighted by $P(x)$, which forces the network to give a closer approximation in those regions of the input space where the density of input data is high, and permits it to give a poorer approximation in regions where the data density is lower. If class labels correlate well with certain regions of the input space X (which we may assume, otherwise the learning task would be insoluble), then the data density will be lower in those regions where the sparsely represented classes lie. This provides the main reason why the network will perform worse on these classes.

This observation leads to the idea of altering the effective class frequencies by presenting more examples from the rarer classes to the learner. In practice, of course, we usually have no way of generating further samples from any class, so resampling is simulated by replicating some of the samples of the rarer classes. An extreme case of this is when the training data set is manipulated so that it contains the same amount of training examples from each class. When training an ANN with the backpropagation algorithm, there is of course no need to really replicate the samples: only the algorithm has to be modified slightly. Usually the training data items are presented to the algorithm in a random order, that is at each iteration a data item is randomly chosen from the full database. We will refer to this method as “full sampling”. A possible alternative is to first choose a class at random, and then randomly pick a training sample from the samples belonging to this class. We will call this general, two-step sampling scheme “probabilistic sampling” [6], and the special case when each class is chosen with uniform probability “uniform class sampling”. In general, however, the choice of the class can follow any distribution, not just a uniform one. For example, if class k is chosen with probability $P(c_k)$, that is its own prior probability, then the two-step sampling approach will be practically equal to the traditional one-step full sampling scheme. This will allow us to generate a continuum between full sampling and uniform class sampling by linearly interpolating the probability of class c_k between $P(c_k)$ and $\frac{1}{K}$.

Let us now discuss how the optimum of the error function of Eq (1) changes when using uniform class sampling instead of full sampling. We will see that manipulating the class frequencies influences both the global data distribution and the local conditional averages. First let us examine the data distribution, which was originally written as

$$P(X) = \sum_k P(X|c_k)P(c_k). \tag{2}$$

The manipulation of the class frequencies can be formalized by weighting the terms as

$$P'(X) = \sum_k P(X|c_k)P(c_k)W_k, \tag{3}$$

where W_k are class-dependent weights. From this we can see that modifying the class frequencies changes the focus of the error function, as it modifies $P(X)$. If class labels correlate well with certain regions of the input space, then giving more samples from the sparse classes indeed corresponds to giving more samples from the low data density regions, thus forcing the net to give a better approximation in these areas.

However, the local conditional probabilities are also influenced by this weighting. Clearly, the new $P'(c_k|X)$ values can be written as

$$P'(c_k|X) = \frac{P(X|c_k)P(c_k)W_k}{\sum_j P(X|c_j)P(c_j)W_j}. \tag{4}$$

We can think of the denominator as a normalizing factor required to make the local estimates add up to one. In the case of uniform class sampling W_k is inversely proportional to $P(c_k)$ and cancels it out, so overall the $P'(c_k|X)$ values will be proportional to $P(X|c_k)$. These will be the local targets of the network, so we can say that with uniform class sampling the neural network learns the class-conditionals $P(X|c_k)$ within a scaling factor. This causes no problem when integrating the network into the HMM framework, and in fact makes it even simpler: the division by the class priors $P(c_k)$ can be omitted, and the scaling factor will not affect the final maximization process.

4 Experimental Results

All the results presented in this paper were obtained using the MTBA Hungarian Telephone Speech Database [8]. This is the first Hungarian speech corpus that is publicly available and has a reasonably large size. The most important data block of the corpus contains the recordings of phonetically balanced sentences that were read out aloud by 500 speakers. Recordings were made via mobile and line phones with the speakers varying both in age and gender. All the sentences were manually segmented and labelled at the phone level, and these manually allocated phone labels served as target classes when training the neural net.

Altogether 58 different phonetic symbols occur in the database, but after fusing certain rare allophones we worked with only 52 phone classes in the experiments.

For training purposes 1367 sentences were selected from the corpus. The word recognition tests reported here were performed on another block of the database that contains city names. All the 500 city names (each pronounced by a different caller) were different. From the 500 recordings only 431 were employed in the tests, as the rest contained significant non-stationary noise or were misread by the caller. All words were assumed to have equal priors in the word recognition tests.

For acoustic preprocessing we applied the Hvite module of the well-known Hidden Markov Model Toolkit (HTK) [10]. We used the most popular preprocessor configuration, that is we extracted 13 MFCC coefficients along with the corresponding delta and delta-delta values, thus obtaining the usual 39-element feature vector [10]. For recognition we used our own HMM/ANN decoder implementation, which was earlier found to have a performance similar to that of the standard HTK recognizer.

The neural net used in the system contained 150 sigmoidal hidden neurons and a softmax output layer. Training was performed by conventional backpropagation. Besides comparing the full sampling and uniform class sampling methods, we decided to create a transition between them by making the algorithm select class c_k with a probability $(1-\lambda)P(c_k)+\lambda\frac{1}{K}$, and tested it with various λ values between 0 and 1. We did so for purely empirical reasons. It should not be forgotten that the whole investigation here originated from the observation that the mathematical proof regarding the estimation of the posteriors assumes ideal conditions, and that in practice problems with imbalanced classes were reported. Our argument of Section 3 regarding the estimation of scaled class-conditionals also assumes ideal conditions that do not hold in reality. So while full sampling tends to behave poorly on rarer classes, uniform class sampling may do just the opposite due to over-compensation. This is why it seemed practically justified to create a transition between the two extremes.

As regards division by the class priors, we argued that theoretically it is required when using full sampling and not when using uniform class sampling. However, it is not obvious whether we should use it when the training scheme is somewhere in between. Furthermore, there is evidence that under certain conditions even the conventional model may not require this division [2]. Owing to these uncertainties, we decided to always run the recognizer with the division factor and without it.

The stopping criterion is always a critical issue with every gradient-based algorithm. With our system we have the long-known observation that a certain fixed number of iterations (with a gradually decreased learning rate) produces a nearly optimal solution which cannot be significantly improved either by further iterations or subtle training criteria. However, because uniform class sampling changes the distribution of the data, we could not be sure that the usual amount of iterations were enough in this case. So in each case we allowed two further rounds of 10 iterations. The results reported are the averages of the three scores

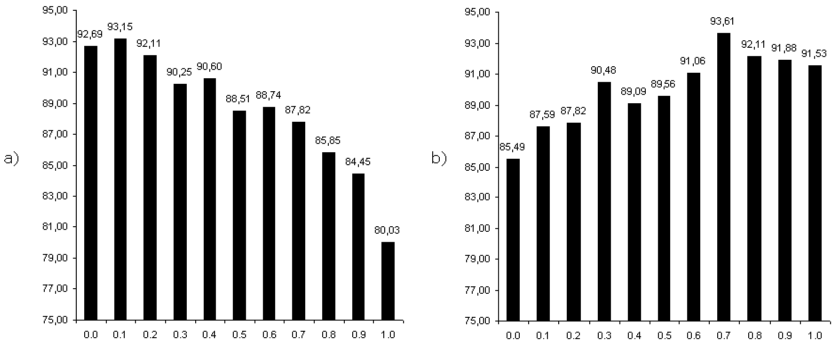


Fig. 1. Word recognition accuracies(%) as a function of λ, with and without division by the priors

obtained after the three iteration cycles. We should mention here that these never differed significantly, their deviation always being around 1-1.5%, which can be attributed to the random factors present in the whole training process.

Figure 1 shows the recognition results for different λ values, both with and without division by the priors. Clearly, a λ around 0.1 seems optimal when dividing by the priors, and a λ of 0.7 resulted in the best results when no division by the priors was applied. These are both better than the corresponding results at $\lambda = 0.0$ and $\lambda = 1.0$ which should have performed the best, according to the proofs discussed in Section 3. This justifies the point that in practice it is worth using the probabilistic sampling scheme for the training of ANNs of HMM/ANN hybrids as it can bring about a modest improvement over the conventional method ($\lambda = 0.0$, division by the priors).

5 Conclusions

This paper investigated the feasibility of the probabilistic sampling training scheme for the training of the ANN components of HMM/ANN hybrid speech recognizers. First we examined uniform class sampling, which is a special case of probabilistic sampling. We argued that although it invalidates the a posteriori probability proof of the conventional training scheme, it is still usable because it gives estimates of the class-conditional probabilities (within a scaling factor) and, in fact, the recognition system requires just these anyway. Second, we suspected that in practice it might be worth interpolating between the conventional full sampling and uniform class sampling, as the mathematical proofs made unrealistic assumptions. In the experiments we indeed found that the optima are somewhere in between – around $\lambda = 0.1$ and $\lambda = 0.7$ respectively, depending on whether we divide by the class priors or not. In both cases our results were slightly better that those obtained by the conventional approach ($\lambda = 0$, division by the priors). This justifies our use of the proposed training scheme in HMM/ANN hybrids.

References

1. Bishop C. M.: Neural Networks for Pattern Recognition. Clarendon Press (1995)
2. Bourlard, H. A., Morgan, N.: Connectionist Speech Recognition – A Hybrid Approach. Kluwer Academic (1994)
3. Bourlard, H. A., Morgan, N.: Hybrid HMM/ANN Systems for Speech Recognition: Overview and New Research Directions. In: Giles, C. L. and Gori, M. (eds.): Adaptive Processing, LNAI 1387, pp. 389-417, (1998)
4. Chawla, N. V., Japkowicz, N. and Kolcz, A. (eds.).: Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Data Sets. http://www.site.uottawa.ca/ nat/Workshop2003/workshop2003.html (2003)
5. Japkowicz, N. (ed.): Proceedings of the AAAI'2000 Workshop on Learning from Imbalanced Data Sets. AAAI Tech. Report WS-00-05 (2000)
6. Lawrence, S., Burns, I, Back, A, Tsoi, A. C., Giles, C. L.: Neural Network Classification and Prior Class Probabilities. In: Tricks of the Trade: Lecture Notes in Computer Science State-of-the-Art Surveys, ed. Orr, G., Müller, K. R. and Caruana, R., Springer, pp. 299-314. (1998)
7. Trentin, E., Bengio, Y., Furnlanello, C., De Mori, R.: Neural Networks for Speech Recognition. In: De Mori (ed.): Spoken Dialogues with Computers, Academic Pr., pp. 311-361. (1998)
8. Vicsi, K, Tóth, L., Kocsor, A., Csirik, J.: MTBA – A Hungarian Telephone Speech Database. Híradástechnika, Vol. LVII, No. 8 (2002) 35- 43 (in Hungarian)
9. Weiss, G. M., Provost, F.: The Effect of Class Distribution on Classifier Learning: An Empirical Study. Tech. Report ML-TR-44, Dep. Comp. Sci., Rutgers Univ. (2002)
10. Young, S. et al.: The HMM Toolkit (HTK) – software and manual. http://htk.eng.cam.ac.uk

Chord Classifications by Artificial Neural Networks Revisited: Internal Representations of Circles of Major Thirds and Minor Thirds

Vanessa Yaremchuk and Michael R.W. Dawson

Biological Computation Project,
Department of Psychology,
University of Alberta,
Edmonton, Alberta,
Canada T6G 2P9
mike@bcp.psych.ualberta.ca

Abstract. This paper describes an artificial neural network that can be viewed as an extension of a pioneering network described by Laden and Keefe. This network was trained to classify sets of four musical notes into four different chord classes, regardless of the musical key or the form (inversion) of the chord. This new network has a slightly modified training set; after successful training the internal structure was analyzed and was found to be unique. That is, rather than using the 12 musical notes of Western music, the network used only 4 musical notes based upon circles of major thirds and of minor thirds[1].

1 Introduction

Artificial Neural Networks (ANNs) have been used to study a variety of musical tasks such as perception of pitch, perception of tonal structure, perception of musical sequences, and composition. See the many examples in [2, 3].

In a pioneering study, Laden and Keefe [4] trained ANNs to classify sets of musical notes as being major, minor or diminished chords. They used a series of different networks, varying the number of hidden units and network connectivity. None of their simple networks were able to classify the chords with complete accuracy. One purpose of the current study is to describe a simple ANN that is capable of performing this task, as well as more complicated variants of it, completely correctly. A second purpose is to examine the representations developed by the ANN that provided this capability.

The current chord classification task is more complicated than Laden and Keefe's as follows: First, the network presented here classifies four types of chords. Dominant chords have been added to the previous types of major, minor and diminished. Second, it uses four notes (tetrachords) to represent each stimulus; Laden and Keefe used triads to represent the major and minor chords. Third, all inversions of each chord are included in the training set; Laden and Keefe only used one form of each chord. The current chord network differs from Laden and Keefe's in that its units use a Gaussian activation function instead of the traditional sigmoid-shaped activation function. Although it uses a Gaussian function, the ANN is not a radial basis function

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 605 – 610, 2005.

network. This is because its net input function is a dot product, not a distance function[5, 6]. As shown below, one result of these changes is that a simple network correctly classifies 100% of the chords, even with this more complex version of the task.

2 Method

The ANN used 24 input units to create a binary representation of chords, in which each input unit is associated with a particular musical note in a purely local sense over two octaves. This is illustrated in Figure 1, where each input unit is associated with a key on a keyboard that spans two octaves starting from a low A and ranging up to a high G#. A note is included in a given stimulus by giving its input unit an activation of 1, otherwise the input unit is turned off by giving it an activation value of zero.

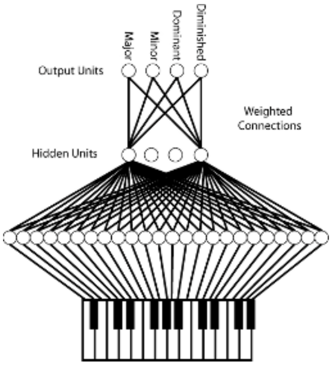


Fig. 1.

Four output units were used – one for each chord type. The network was trained to classify an input pattern as being one of four types of chord regardless of musical key or of the inversion of the chord. The correct response was simply represented by having the appropriate output unit turn on (value of 1) while the remaining three output units turn off (value of 0).

The network used four hidden value units, because initial tests indicated this was the smallest number of hidden units we could use to have the network successfully classify all the chords. Fewer than four hidden units continually resulted in networks that were not able to completely learn the task. That is, there were always some chords that were not correctly classified.

The network uses value units, specifically a Gaussian activation function of the form:

$$G(net_i) = \exp(-\pi(net_i - \mu_j)^2) \quad (1)$$

Where G() is the activation function, net_i is the net input for output unit i, and μ_j is the mean of the Gaussian for value unit j [6, 7]. The network was trained using a variation of the generalized delta rule for networks of value units [7]. Prior to training, all connection weights were randomly assigned values ranging from –0.10 to

+0.10 and all biases (thresholds) were set to 0.00. The network was trained with a learning rate of 0.0001 and zero momentum. Each of the 135 chords was presented to the network each epoch. The order of presentation was randomized prior to each epoch. Connection weights were updated using the learning rule after each stimulus presentation. Training stopped, after 82579 epochs, when the network generated a hit for every output unit on every pattern. A hit was defined as an activation of 0.90 or higher when the desired activation was 1.00, and as an activation of 0.10 or lower when the desired activation was 0.00.

3 Network Interpretation

How does this simple network correctly classify this complicated set of chords? Interpretation of the internal structure of the trained network reveals it isn't a case of mere memorization.

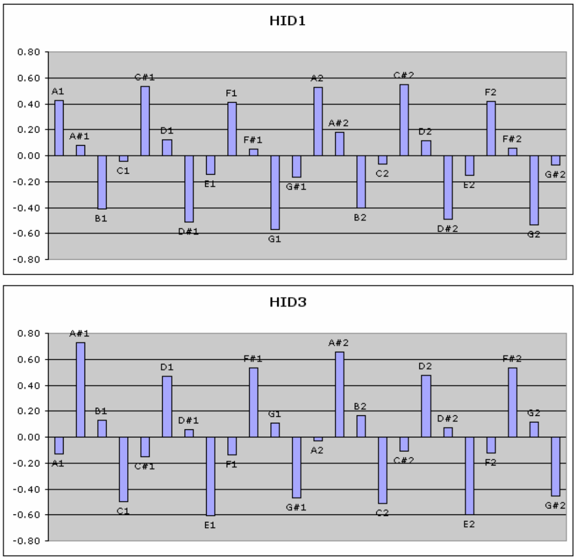


Fig. 2. Connection weights for hidden units 1 and 3

The first step in interpreting the network's structure was to examine the connection weights from the input units to the hidden units. The connection weights for hidden unit 1 and hidden unit 3 can be seen in Figure 2. They reveal a conversion of individual (and different) musical notes into equivalence classes of notes defined by circles of major thirds (see Figure 3). That is, the connection weights into each unit can be classified as belonging to one of four categories: strong negative, weak negative, strong positive, and weak positive. All of the notes belonging to one of the circles of major thirds from Figure 3 are assigned the same type of connection weight. Both hidden units 1 and 3 use the circles of major thirds representation but they are out of phase with each other. They do not assign the same connection weight to the same circles.

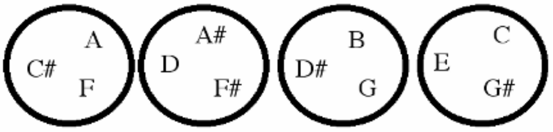


Fig. 3. Circles of major thirds

A similar analysis of the weights of connections between the inputs and Hidden units 2 and 4 reveals a different set of equivalence classes for musical notes: different notes are assigned (roughly) to circles of minor thirds, which are illustrated in Figure 5. As can be seen in Figure 4, it is not such a clean fit as with hidden units 1 and 3 to the major thirds. There is some effect of the octave range, producing variations in connection weight that modulates the circles of minor thirds organization. Nevertheless, circles of minor thirds provide a powerful method of interpreting the regularities in these graphs.

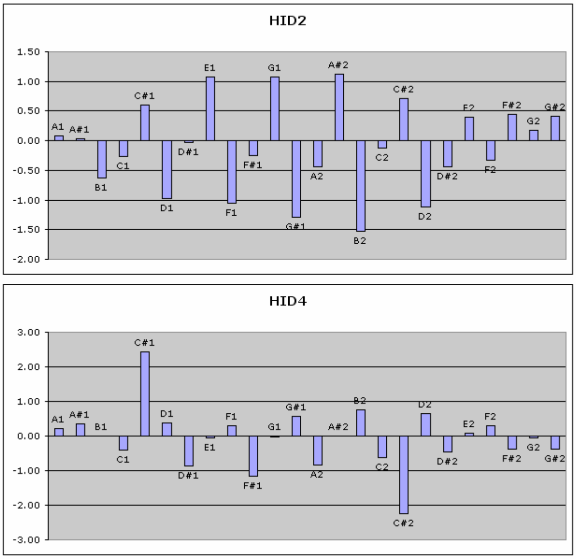


Fig. 4. Connection weights for hidden units 2 and 4

The second step in interpreting the network's internal structure was to examine hidden unit responses to the patterns, as is summarized in the table below.

For example, consider the responses of hidden units 2 and 4 in conjunction. When the activation levels of these hidden units in response to each training pattern are added together, minor chords produce a combined response of 0.61 or higher while all other chord types (major, dominant, diminished) generate a combined response of 0.20 or lower. (Activation values are all positive and range from 0.00 to 1.00.) All of the minor chords elicit a moderate to high response in hidden unit 2 or hidden unit 4 and a small number elicit a moderate to high response in both.

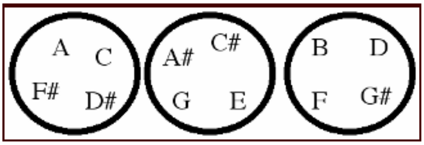
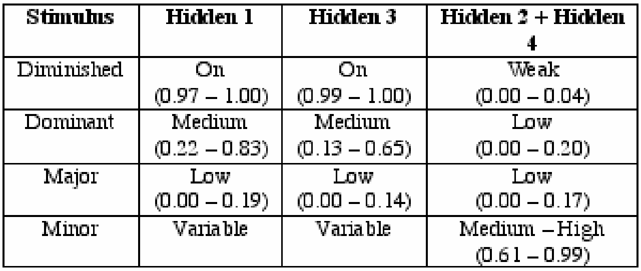


Fig. 5. Circles of minor thirds

Another look at hidden units 2 and 4 reveals an interesting division of labor in their responses to minor chords. Hidden unit 2 responds to the following minor chords: A, A#, C, C#, D#, E, F#, G. This set of keys makes up the octatonic collection, also known as the diminished scale in jazz [8] as it is made up of two diminished chords. The remaining keys (B, D, F, G#) also make up a diminished chord. Hidden unit 4 responds to these minor chords as well as a small number of others that overlap with the hidden unit 2 set.

Similar insights emerge from considering the responses of the other two hidden units. Hidden unit 1 generates high activity for all diminished chords (e.g., A C D# F#). Notice that diminished chords have exactly one note from each of the above circles of major thirds. It activates strongly enough to the diminished chords to distinguish them from the dominant and major chords. Hidden unit 3 demonstrates the same pattern to an even stronger degree. Hidden units 1 and 3 have low responses to all major chords. Major chords (e.g., C E G C) only ever have notes from two of the circles of major thirds (Fig 3) at a time. For example, the notes for the root position of C major all come from two of the circles in figure 3. Dominant chords generate a moderate response in hidden unit 1 (0.22 – 0.83) and a weak to moderate response in hidden unit 3 (0.13 – 0.65).



Stimulus	Hidden 1	Hidden 3	Hidden 2 + Hidden 4
Diminished	On (0.97 – 1.00)	On (0.99 – 1.00)	Weak (0.00 – 0.04)
Dominant	Medium (0.22 – 0.83)	Medium (0.13 – 0.65)	Low (0.00 – 0.20)
Major	Low (0.00 – 0.19)	Low (0.00 – 0.14)	Low (0.00 – 0.17)
Minor	Variable	Variable	Medium – High (0.61 – 0.99)

4 Discussion

The current network deviated from those studied by Laden and Keefe in several respects. All of the chords were represented as tetrachords, and different inversions of each tetrachord were used. This was possible because all of the stimuli were represented using a local representation that covered two octaves, rather than a single octave "pitch class" representation. Four chord types were created with this representation, rather than three. Finally, the processing units employed a Gaussian activation function rather than a logistic. These variations in method produced a relatively simple network that was able to learn to generate correct chord classifications for a relatively challenging stimulus set, representing an advance over the small networks described by Laden and Keefe.

Of greater interest, though, were the regularities that were uncovered by interpreting the internal structure of the network. The internal structure demonstrates that the network did not merely memorize the chords but has classified them. Formal western musical structures – which were used to define the training set – are based upon a set of 12 different note types, with each note class repeating every octave. The network used a much different set of note classes. Two of the hidden units assigned notes to only 4 different types, where notes that are differentiated in Western music (e.g., A, C#, F) were all treated as being the same note because they all belong to the same circle of major thirds (Figure 3). The other two hidden units also condenses the 12 notes of Western music into 4, but this time based these equivalence classes on circles of minor thirds (Figure 5). A question arises as to whether humans might use these same representations when they process music chords. We are currently investigating this issue by conducting experiments in which human listeners are trained to identify chords in a learning paradigm that is analogous to the one used to train the network that was reported above.

References

1. Franklin, J.A. Recurrent neural networks and pitch representations for music tasks. in Seventeenth International Florida Artificial Intelligence Research Symposium Conference. 2004. Miami Beach, FA: AAAI Press.
2. Todd, P.M. and D.G. Loy, Music and connectionism. 1991, Cambridge, Mass.: MIT Press. xi, 268.
3. Griffith, N. and P.M. Todd, Musical Networks: Parallel Distributed Perception And Performace. 1999, Cambridge, Mass.: MIT Press. xv, 385.
4. Laden, B. and B.H. Keefe, The representation of pitch in a neural net model of pitch classification. Computer Music Journal, 1989. 13: p. 12-26.
5. Moody, J. and C.J. Darken, Fast learning in networks of locally-tuned processing units. Neural computation, 1989. 1: p. 281-294.
6. Dawson, M.R.W., Minds And Machines: Connectionism And Psychological Modeling. 2004, Malden, MA: Blackwell Pub.
7. Dawson, M.R.W. and D.P. Schopflocher, Modifying the generalized delta rule to train networks of nonmonotonic processors for pattern classification. Connection Science, 1992. 4: p. 19-31.
8. Krumhansl, C.L., Cognitive Foundations Of Musical Pitch. Oxford psychology series; no. 17. 1990, New York: Oxford University Press. x, 307 p.

Biclustering Gene Expression Data in the Presence of Noise

Ahsan Abdullah[1,2] and Amir Hussain[2]

[1] National Univ. of Computer & Emerging Sciences, Islamabad, Pakistan
[2] Department of Computing Science & Mathematics, University of Stirling, UK
ahsan@nu.edu.pk, ahu@cs.stir.ac.uk

Abstract. Production of gene expression chip involves a large number of error-prone steps that lead to a high level of noise in the corresponding data. Given the variety of available biclustering algorithms, one of the problems faced by biologists is the selection of the algorithm most appropriate for a given gene expression data set. This paper compares two techniques for biclustering of gene expression data i.e. a recent technique based on crossing minimization paradigm and the other being Order Preserving Sub Matrix (OPSM) technique. The main parameter for evaluation being the quality of the results in the presence of noise in gene expression data. The evaluation is based on using simulated data as well as real data. Several limitations of OPSM were exposed during the analysis, the key being its susceptibility to noise.

1 Introduction

The developments in DNA arrays enable simultaneous measurements of the expression levels of thousands of genes. These methodologies have led to an explosion in the rate at which gene expression data is accumulated. One of the challenges in this regard is the inherent high level of noise present. Unfortunately, the noise often hides the patterns of interest – for example, the data often contains 'technical' and 'biological' noise [3]. Several potential sources of measurement of noise are discussed in [5].

Analyzing the DNA micro array data involves some form of 'grouping' that is biologically significant. Due to certain well-known limitations of traditional (one-way) clustering techniques, several (two-way) clustering or *biclustering* techniques have recently been proposed. The biclustering problem being NP-Complete in nature [2].

Several biclustering algorithms have recently been analyzed in [6], however, the important issue of noise immunity did not receive sufficient attention. In this paper we analyze two biclustering techniques i.e. a new technique based on crossing minimization paradigm [1] and an Order Preserving Sub Matrix (OPSM) technique [2] with specific reference to their comparative performance under noise.

OPSM [2] is a biclustering technique that incorporates a greedy algorithm for discovering a fixed pattern of rows in a data set. It defines its biclusters as a sub matrix, for which there exists a permutation of columns under which the sequence of values in every row is strictly increasing. Our work analyzes this method of

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 611 – 616, 2005.

biclustering gene expression data for its real world application. Biclustering by crossing minimization [1] is a graph theoretic technique, in which the data matrix is represented by a binary bipartite graph. Biclustering is achieved by crossing minimization, as vertices with high connectivity are "pulled" together, thus resulting in cluster formation.

1.1 Micro Array Technology

Micro array is a slide of glass or some other substrate on which thousands of DNA molecules are attached at fixed location (spots). Each spot correspond to a specific gene in a cell. RNA from the sample of interest is taken and hybridized with this array. The amount of hybridized RNA is measured and this gives the expression level of a particular gene. Micro array experiments typically involve five steps.

1.2 Noise in Gene Expression Data

Micro array technology is most commonly available in two forms i.e. GeneChip®, a trademark used by Affymetrix Inc, that relies on DNA oligonucleotide for chip manufacturing and Spotted Arrays developed at Stanford University, use glass slide on which DNA is immobilized using a robot arm.

These technologies enable measuring expression levels of thousands of genes in a cell simultaneously under one experimental condition, or of a particular tissue type and give a global view of cell. However, DNA micro array technology is not without shortcomings. Three forms of noise or limitations of micro arrays is highlighted and studied.

Noisy Nature of Data. The actual expression value changed, this is due to 'experimental' (technical) noise or by 'biological' noise [3]. Experimental noise is caused by different stages of micro array experiments listed above. Careful design and selection of the immobilized DNA and planning for hybridization of array with samples is required. Biological variations are due to real differences, impurities or misclassification between different cell types and tissues, which are being compared.

Missing Values. Values in data matrices are frequently missing due to methods used for their creation, dust particles or scratches on the slides and scanning of image for extraction of expression levels are some of the major contributors of this type of noise.

Irrelevant Genes. As each experiment yields the levels of expression of thousands of genes and majority of such genes are irrelevant to the class distinction. Therefore, the combined effect of large numbers of irrelevant genes can potentially obscure the contributions of the relevant ones. Irrelevant genes are regarded as noise, because the problem is how to identify the irrelevant genes from the relevant ones?

Study of these techniques is beyond the scope of this paper, but the point being made here is that noise in gene expression data is more of a rule than an exception.

1.3 Challenges of Gene Clustering

Due to the special characteristics of gene expression data, and the particular requirements from the biological domain, gene-based clustering presents several challenges and is still an open problem [4]. The main challenges being no a-prior knowledge about “true” number of clusters (or biclusters) and ensuring high noise immunity.

2 Background

Ben-Dor et al. [2] defined a bicluster as an order preserving sub matrix. According to their definition, a bicluster is a group of rows whose values induce a linear order across a subset of the columns. Given bicluster (Fig. 1) is an example of OPSM where columns are arranged in order c1 < c2< c3< c4 linear.

The data consist of n x m matrix D, where the rows correspond to genes and the columns to tissues (or, more generally, to conditions). OPSM finds an order-preserving submatrix for which there is a permutation of its columns, under which the sequence of values in every row is strictly increasing. For expression data a sub matrix is determined by a set of genes G and a set of tissues T such that, within the set of tissues T, the expression levels of all the genes G have the same direction.

c1	c2	c3	c4
70	13	19	10
9	40	49	35
40	20	27	15
90	15	20	12

(a) Input Data

c1	c2	c3	c4
10	13	19	70
35	40	49	69
15	20	27	40
12	15	20	90

(b) Output Data

Fig. 1. An example of OPSM bicluster

Biclustering by crossing minimization [1] in its simplified form is given as a four step procedure as follows:

1. Discretize the given data matrix S to obtain a binary data matrix S_B.
2. Corresponding to S_B create a binary bipartite graph G_B i.e. with edge weights 1 and 0.
3. Run a crossing minimization heuristic on G_B corresponding to S_B to get G^+_B
4. Extract clusters from $G^+_{B.}$

Discretization reduces the complexity of the problem. Observe that when data matrix is discretized (in step-1); values over a certain threshold are set to 1, while remaining values are set to 0. There are number of ways of data discretization [8].

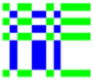


Fig. 2(a)

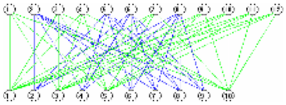


Fig. 2(b)

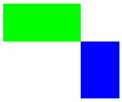


Fig. 2(c)

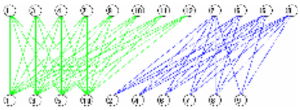


Fig. 2(d)

Fig. 2(a) shows S_B before using crossing minimization heuristic, while Fig. 2(c) shows the same matrix after using a crossing minimization heuristic. Fig. 2(d) shows the bipartite graph view of Fig. 2(c) i.e. G^+_B with clusters extracted and crossings reduced from 678 to 258.

3 Related Work

We will touch upon the related work done in the field of biclustering and the noise models proposed and used. Our work is focused around comparison of biclustering techniques in the presence of noise; we have not been able to find work which covers the idea of this paper.

In [6] several biclustering techniques have been discussed (including OPSM) for the type of biclusters they can find, their structure and method used to perform the search, but not with reference to noise. In [7] Gauhar Wadhera has compared different biclustering techniques for their time complexity, real data set used and the way biclusters are discovered.

4 Results Using Simulated Data

For quantifying the quality of biclusters two measures are used i.e. Purity and Entropy. These are with reference to the original biclusters that exits in the simulated data and are known in advance. The value of the Purity (P) and Entropy (E) will be between 0 and 1. The purest bicluster having Purity of 1 and Entropy of 0.

Purity of a bicluster is a measure of how pure the bicluster is. The Purity of a biclustering solution can be defined as the number of points (data elements) in that bicluster, which also belongs to the original bicluster. Entropy is a measure of the "disintegration" of the bicluster. More formally, Entropy of a bicluster is defined as the number of points (or data elements) of other biclusters that are included in bicluster being considered.

We corrupt G^+_B in two steps. In the first step, the vertices in each bipartition are randomly permuted. In the second step, the resulting G_B is "contaminated" by randomly adding edges (white noise) between the vertices in the bipartitions with probability $< ½$. Similarly, edges are also randomly removed (white noise) from the vertices in each of the bipartitions with probability $< ½$. The effect of these operations on S_B would be replacement of 1's by 0's for edges removed, and replacement of 0's by 1's for edges inserted.

We consider a simulated data set consisting of $K_{15,5} \cup K_{30,5} \cup K_{55,5}$ where $K_{m,n}$ represents a bipartite clique with *m* and *n* vertices in each bipartition. The simulated data set is intentionally kept small due to execution time limitations of OPSM. The graphs of P and E versus varying white noise are shown in Fig. 3. Fig. 4 shows the visualization of the same data set, where black dots represents noise, white spots (or dots) represent missing data or noise, while each cluster is represented by a different color. In Fig. 4(d) the red rectangles represent the clusters extracted.

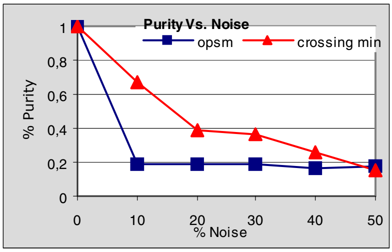


Fig. 3(a). Effect of noise on purity of biclusters

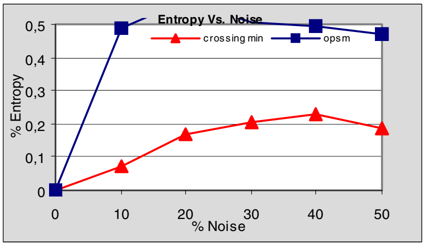


Fig. 3(b). Effect of noise on entropy of biclusters

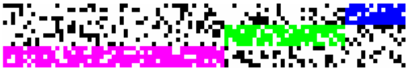


Fig. 4(a). Input with 25% white noise

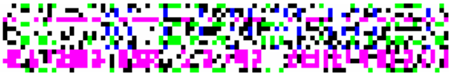


Fig. 4(b). Randomly permuted Fig. 2(a)

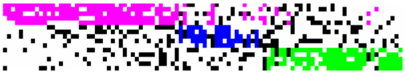


Fig. 4(c). Output using crossing minimization

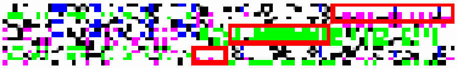


Fig. 4(d). Output using OPSM

Observe that in Fig. 4(c) the biclusters formed using crossing minimization are well defined, while in Fig. 4(d) corresponding to biclusters formed using OPSM have disintegrated.

5 Results Using Real Data

Experiments were performed using real gene expression data set gppca (59x11) downloaded from http://ep.ebi.ac.uk/EP/EPCLUST/. Because of the inherent limitations of OPSM [2], a small data set was used i.e. consisting of 59 rows and 11 columns. However, our crossing minimization based biclustering technique can bicluster hundreds of rows and hundreds of columns simultaneously, in less than a second. OPSM took hundreds of seconds for more than six columns.

We assume for simplicity, that the real data set is without any noise, and we then add white noise. Using OPSM two clusters were extracted, with 24 and 21 rows, respectively. Note that the number of clusters and the number of columns (i.e. 4) within a cluster remained same with 0% noise as well as 10% noise. The pool size for partial models maintained was 45% [2]. For the example being discussed, median value based discretization was used.

For biclustering by crossing minimization the numbers of columns are not required to be specified prior to extraction, hence resulting in more "natural" biclustering. Using this technique three major clusters were extracted as shown in Fig. 5(c).

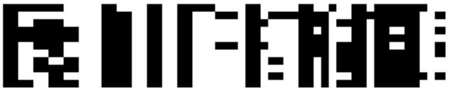


Fig. 5(a). Input with 0% noise

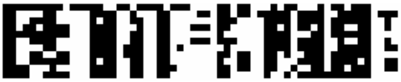


Fig. 5(b). Input with 10% noise

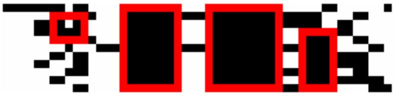


Fig. 5(c). Output with 0% noise

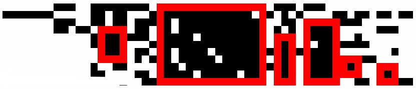


Fig. 5(d). Output with 10% noise

As biclustering results in grouping of similar values, therefore, one measure of the quality of the results could be the standard deviation (Stdev) of each column. It was found that even with only four columns, the Stdev using OPSM was about twice as compared to biclustering using crossing minimization.

6 Conclusions

We see that OPSM has low noise immunity as compared to biclustering by crossing minimization. For this study the cluster "heights" in simulated data were deliberately kept same due to limitations of OPSM; biclustering by crossing minimization does not have such a limitation.

References

1. A. Abdullah & A. Hussain, "A New Biclustering Technique Based On Crossing Minimization", In proceedings *Brain Inspired Cognitive Systems Conference (BICS'04),* Univ. of Stirling, Scotland, UK, Aug/Sep. 2004
2. Ben-Dor, B. Chor, R. Karp, & Z. Yakhini, "Discovering local structure in gene expression data: The order-preserving submatrix problem", In *Proceedings of the 6th International Conference on Computational Biology (RECOMB'02)*, pages 49–57, 2002
3. Keller, M. Schummer, L Hood & W. Ruzzo, "Bayesian Classification of DNA Array Expression Data" Technical Report UW-CSE-2000-08-01, August, 2000
4. Jiang, C. Tang, & A. Zhang, "A *Cluster Analysis for Gene Expression Data: A Survey"*. Technical Report 2002-06, State University of New Your at Buffalo, 2002
5. R. Dror, "Noise models in gene array analysis" Report in fulfillment of the area exam requirement in the MIT Department of Electrical Engineering and Computer Science, 2001
6. S. C. Madeira & A. L. Oliveira, "Biclustering algorithms for biological data analysis: a survey", *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1), pp. 24-45, Jan. 2004
7. G. Wadhera, "Research Literature Commentary: Subspace clustering methods for gene expression data analysis", www.stanford.edu, Aug. 2004
8. Y. Yang and G. I. Webb, "*A Comparative Study of Discretization Methods for Naive-Bayesian Classifiers*". In proc. of the Pacific Rim Knowledge Acquisition Workshop, National Center of Sciences, Tokyo, Japan, 2002

Gene Extraction for Cancer Diagnosis by Support Vector Machines

An Improvement and Comparison with Nearest Shrunken Centroid Method

Te-Ming Huang and Vojislav Kecman

School of Engineering, The University of Auckland, New Zealand
v.kecman@auckland.ac.nz, huangjh@win.co.nz

Abstract. A cancer diagnosis by using the DNA microarray data faces many challenges the most serious one being the presence of thousands of genes and only several dozens (at the best) of patient's samples. Thus, making any kind of classification in high-dimensional spaces from a limited number of data is both an extremely difficult and a prone to an error procedure. The improved Recursive Feature Elimination with Support Vector Machines (RFE-SVMs) is introduced and used here for an elimination of less relevant genes and just for a reduction of the overall number of genes used in a medical diagnostic. The paper shows why and how the, usually neglected, penalty parameter C influence classification results and the gene selection of RFE-SVMs. With an appropriate parameter C chosen, the reduction in a diagnosis error is as high as 37% on the colon cancer data set. The results suggest that with a properly chosen parameter C, the extracted genes and the constructed classifier will ensure less over-fitting of the training data leading to an increase accuracy in selecting relevant genes.

1 Introduction

Recently, huge advances in DNA microarrays have allowed the scientist to test thousands of genes in normal or tumor tissues on a single array and check whether those genes are active, hyperactive or silent. Therefore, there is an increasing interest in changing the criterion of tumor classification from morphologic to molecular [1]. In this perspective, the problem can be regarded as a classification problem in machine learning, in which the class of a tumor tissue with a feature vector $\mathbf{x}$ is determined by a classifier. Each dimension, or a feature, in $\mathbf{x}$ holds the expression value of a particular gene which is obtained from DNA microarray experiment. The classifier is constructed by inputting l feature vectors of known tumor tissues into machine learning algorithms. To construct an accurate and reliable classifier with every gene included is not a straightforward task due to the fact that in the practice a number of tissue samples available for training is much less (a few dozens) than the number of features (a few thousands). In such a case, the classification space is nearly empty and it is difficult to construct a classifier that generalizes well. Therefore, there is a need

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 617–624, 2005.

to select a handful of most decisive genes in order to shrink the classification space and to improve the performance.

Support vector machines (SVMs) are one of the latest developments in statistical learning theory and they have been shown to perform very well in many areas of biological analysis including evaluating microarray expression, detecting remote protein homologies, and recognizing translation initiation sites. More recently, SVMs-based feature selection algorithms dubbed, Recursive Feature Elimination with Support Vector Machines (RFE-SVMs) have been introduced and applied to a gene selection for a cancer classification. In this work, we present the simulation results of the improved RFE-SVMs by tuning the C parameters on the popular colon cancer data set [2] and make comparison with the well-known nearest shrunken centroid method [3,4]. The C parameter plays an important role for SVMs in preventing an over-fitting but its effects on the performance of RFE-SVMs are still unexplored.

The paper is organized as follows: In section 2, we review SVM-RFE and some prior work in this area. The results on the influence of the C parameter on a correct selection of relevant features are presented in section 3. Section 4 shows the comparison between the improved RFE-SVMs and the nearest shrunken centroid on colon data set [2].

2 Prior Work

2.1 Support Vector Machines

The support vector machine classifier is based on the idea of margin maximization and it can be found by solving the following optimization problem [5]:

$$\text{Min } \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i^2 \tag{1a}$$

$$\text{s.t } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, l \tag{1b}$$

The decision function for linear SVMs is given as $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$. In this formulation, we have the training data set (x_i, y_i) $i = 1, \ldots, l$ where $x_i \in \Re^n$ are the training data points or the tissue sample vectors, y_i are the class labels, l is the number of samples and n is the number of genes measured in each sample. By solving the optimization problem (1), i.e., by finding the parameters $\mathbf{w}$ and b for a given training set, we are effectively designing a decision hyperplane over an n dimensional input space that produces the maximal margin in the space. Generally, the optimization problem (1) is solved by changing it into the dual problem below,

$$\text{Max } L_d(\alpha) = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T\mathbf{x}_j \tag{2a}$$

$$\text{s.t } 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, l \tag{2b}$$

$$\text{and} \quad \sum_{i=1}^{l}\alpha_i y_i = 0 \tag{2c}$$

In this setting, one needs to maximize the dual objective function $L_d(\alpha)$ with respect to the dual variables α_i only. The equality constraint (2c) can be eliminated by adding a constant of 1 to all the entries of the kernel matrix as suggested in [6,7]. Hence, the dual objective becomes

$$\text{Max } L_d(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j (\mathbf{x}_i^T \mathbf{x}_j + \mathbf{1}) \tag{3}$$

subject only to the box constraints $0 \leq \alpha_i \leq C$. The optimization problem can be solved by various established techniques for solving general quadratic programming problems with inequality constraints.

2.2 Recursive Feature Elimination with Support Vector Machines

The idea of using the maximal margin for gene selection was first proposed in [8] and it was achieved by coupling recursive features elimination with linear SVMs to find a subset of genes that maximizes the performance of the classifiers. In a linear SVM, the decision function is given as $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$ or $f(\mathbf{x}) = \sum_{k=1}^{n} w_k x_k + b$. For a given feature x_k, the size of the absolute value of its weight w_k shows how significantly does x_k contribute to the margin of the linear SVMs and to the output of a linear classifier. Hence, it is used as a feature ranking coefficient in RFE-SVMs. In the original RFE-SVMs, the algorithm first starts constructing a linear SVMs classifier from the microarray data with n number of genes, then the gene with the smallest w_k^2 is removed and another classifier is trained on the remaining $n-1$ genes. This process is repeated until there is only one gene left. A gene ranking is produced at the end from the order of each gene being removed and the most relevant gene will be the one that is left at the end. However, for computational reasons, the algorithm is often implemented in such a way that several features are reduced at a time. In such a case, the method produces a feature subset ranking, as opposed to a feature ranking. Therefore, each feature in a subset may not be very relevant individually, and it is the feature subset that is optimal in some sense [8].

2.3 Selection Bias and How to Avoid It

As shown in [8], the leave-one-out error rate of RFE-SVMs can reach as low as zero percent with only 16 genes on the well-known colon cancer data set from [2]. However, as it was later pointed out in [1], the simulation results in [8] did not take selection bias into account. The leave-one-out error presented in [8] was measured using the classifier constructed from the subset of genes that were selected by RFE-SVMs using the complete data set. It gives too optimistic an assessment of the true prediction error, because the error is calculated internally. To take the selection bias into account, one needs to apply the gene selection and the learning algorithm on a training set to develop a classifier, and only then to perform an external cross-validation on a test set that had not been seen during the selection stage on a training data set. As shown in [1], the selection bias can be quite significant and the test error that is based on 50% training and 50% test can be as high as 17.5% for the colon cancer data set. Another important observation from [1] is that there are no significant improvements when the number of genes used for constructing the classifier is reduced: the prediction errors are relatively constant until approximately 64 or so genes. These observations indicate that the performance and the usefulness of RFE-SVMs may be in question. However, the influence of the parameter C was neglected in [1] which restricts the results obtained. As a major part of this work, we further investigate the problem by changing (reducing) the parameter C in RFE-SVMs, in order to explore and to show the full potentials of RFE-SVMs.

3 Influence of the Parameter C in RFE-SVMs

The formulation in (1) is often referred to as the 'soft' margin SVMs, because the margin is softened and the softness of the margin is controlled by the C parameter. If C is infinitely large, or larger than the biggest α calculated, the

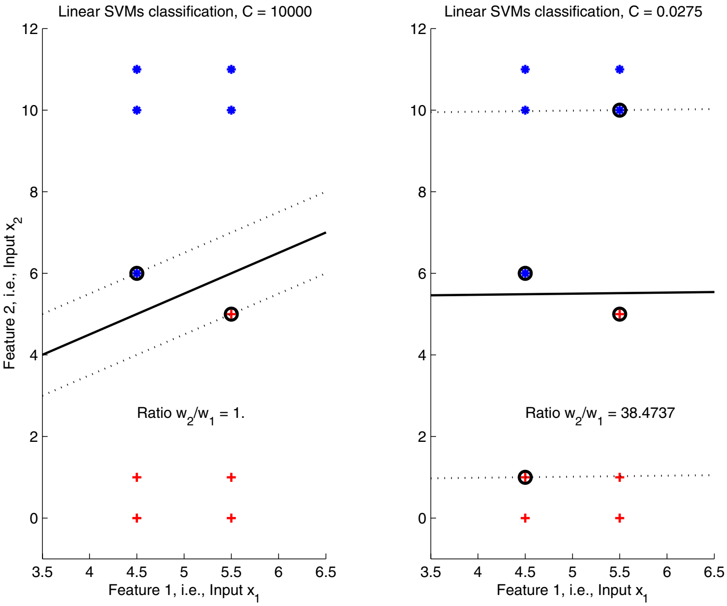


Fig. 1. A toy example shows how C may be influential in a feature selection. With C equal to 10000, both features seem to be equally important according to the feature ranking coefficients (namely, $w_1 = w_2$). With $C = 0.025$, a request for both a maximal and a 'hard' margin is relaxed and the feature 2 becomes more relevant than feature 1, because w_2 is larger than w_1 ($w_2/w_1 = 73$). While the former choice $C = 10000$ enforces the largest margin and all data to be outside it, the later one ($C = 0.025$) enforces the feature 'relevance' and gives better separation boundary because the two classes can be perfectly separated in a feature 2 direction only.

margin is basically 'hard', i.e., no points in the training data can be within or on the wrong side of the margin.

If C is smaller than the biggest original α_i, the margin is 'soft' one. As seen from all the $\alpha_j > C$ will be constrained to $\alpha_j = C$ and corresponding data points will be inside, or on the wrong side of, the margin. In the most of the work related to RFE-SVMs such as [8,9], the C parameter is set to a number that is sufficiently larger than the maximal α_i i.e., a hard margin SVM is implemented within such an RFE-SVMs model. Consequently, it has been reported that the performance of RFE-SVMs is insensitive to the parameter C. However, Fig.1 shows how C may influence the selection of more relevant features in a toy example where the two classes (stars * and pluses +) can be perfectly separated in a feature 2 direction only. In other words, the feature 1 is irrelevant for a perfect classification here. Note in the right hand side plot that a decrease in C i.e., a constraining of the dual variables $\alpha_i = C$, leads to a moving of some data within the margin. However, at the same time this helps in detecting the more relevant feature which is an input 2 here.

4 Gene Selection for the Colon Cancer and Comparison with the Nearest Shrunken Centroid

In this section, we present the selection of relevant genes for the colon data set which is well known in the gene microarray literature. The colon data set was analyzed initially in [2] and it is composed of 62 samples (22 normal and 40 cancerous) with 2000 genes' expressions in each sample. The training and the test sets are obtained by splitting the dataset into two equal groups of 31 elements, while ensuring each group has 11 normal and 20 cancerous tissues. The RFE-SVM is only applied on the training set to select relevant genes and to develop classifiers, and then the classifiers are used on the test set to estimate the error rate of the algorithms. 50 trails were carried out with random split for estimating the test error rate. A simple preprocessing step is performed on the colon data set to make sure each sample is treated equally and to reduce the array effects. Standardization is achieved by normalizing each sample to the one with zero mean and with a standard deviation of one. To speed up the gene selection process, 25% of the genes are removed at each step until less than 100 genes remained still to be ranked. Then the genes are removed one at a time. The simulation results for the colon data set are shown in Fig.2.

The Ambroise and McLachlan's curve in Fig.2 is directly taken from [1] and it is unclear what C value is used in this paper. By comparing the error rates for various C parameters, it is clear that changing the parameter C has significant influence on the performance of RFE-SVMs in this data set. The error rate is reduced from previously 17.5% as reported in [1] to 11.16% (a reduction of 35%) when C is equal to 0.005. For $C = 0.01$, the gene selection procedure improves the performance of the classifier: this trend can be observed by looking at the error rate reduction from initially around 15% at 2000 genes to 11.9% with 26 genes. Similar trend can be observed when $C = 0.005$, but the error rate reduction is

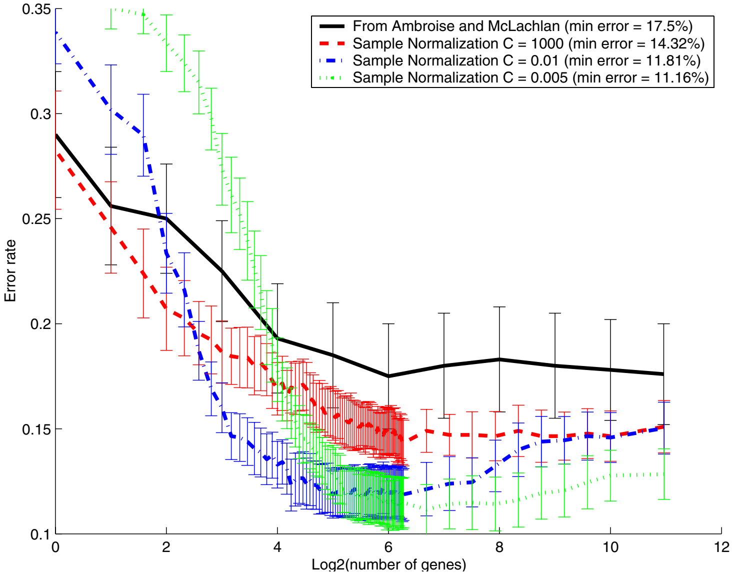


Fig. 2. Simulation result on the colon cancer data set with various C parameters. The error bar represents the 95% confidence interval.

not as significant as in the previous case. This is due to the fact that the error rate of the linear SVMs with $C = 0.005$ is already low, when all the genes are used. This also demonstrates that tuning the C parameter can reduce the amount of over-fitting on the training data even in such a high dimensional space with small number of samples. A preliminary comparison on the lowest leave-one-out error rate between RFE-SVMs and the well-known nearest shrunken centroid from [3] shows RFE-SVMs (8.0% at $C = 0.005$) is slightly better than nearest shrunken centroids (9.67%). The leave-one-out error rates presented here from both algorithms coincides with the suggestion in [1] that there are some wrongly labeled data in the training data set.

In order to further test the performance between the improved REF-SVMs and nearest shrunken centroid, we use again 50% of colon data for training and another 50% for testing. To make the comparison statistically more significant, we perform the experiment 100 times instead of 50 times as in Fig.2. Figure 3 shows the test errors and the corresponding 95% confidence interval of RFE-SVMs and the nearest shrunken centroid with various number of genes. As shown in the Fig.3, the performance of RFE-SVMs is superior to the nearest shrunken centroid in this test setting. It is interesting to point out that the error rate between the two algorithms is more significant in this more difficult setting (less training data) than in the leave-one-out setting. This may indicate that RFE-SVMs has more superior performance when the number of samples is low. The same, better, performance is observed in selecting the genes for CF (Cystic Fibrosis) diagnosis. (Due to the proprietary character of the CF data sets we

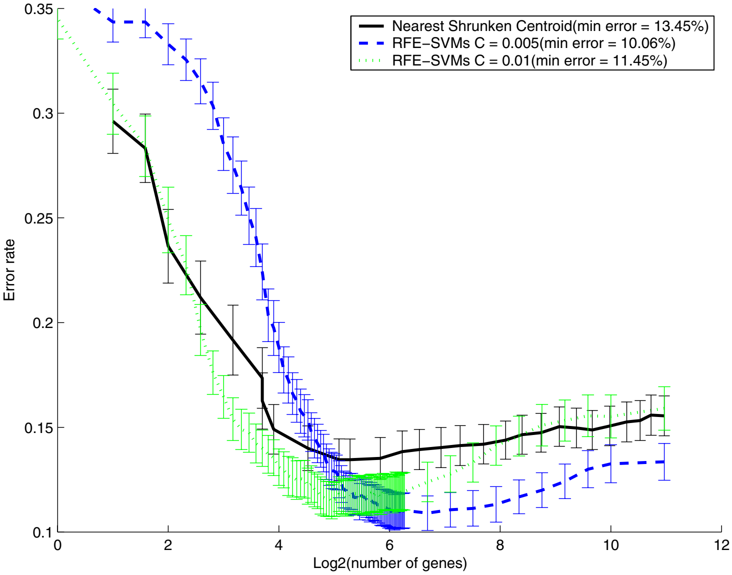


Fig. 3. Test errors on the colon cancer data set for two different methods

can't show the comparative results here). However, for a CF data set (with only 18 samples and approximately 12000 features), we would like just to mention that the RFE-SVM (having the error rate of 5%) performs again much better than the shrunken centroid (where the error rate is 33%).

5 Conclusions

We presented the performance of improved RFE-SVMs algorithm for genes extraction of DNA microarray data for diagnosing colon cancer. Why and how is this improvement achieved by using different values for the C parameter is discussed in details. With a properly chosen parameter C, the extracted genes and the constructed classifier will ensure less over-fitting of the training data leading to an increased accuracy in selecting relevant genes. The simulation results suggest that the classifier performs better in the reduced gene spaces selected by RFE-SVMs than in the complete 2000 dimensional gene space. This is a good indication that RFE-SVMs can select relevant genes, which can help in the diagnosis and in the biological analysis of both the genes' relevancy and their function. The comparison between the improved RFE-SVMs and nearest shrunken centroid on the colon data set suggested that the improved RFE-SVMs performs better when the number of data used for training is reduced. This phenomenal is also observed in the CF data set. Finally, the results in this work are developed from a more machine learning and data mining perspective, meaning unrelated to any valuable insight from a biology and medicine. Thus, there is a need for a tighter cooperation between the biologists and/or medical experts and data miners in all the future investigations.

References

1. Ambriose, C., McLachlan, G.: Selection bias in gene extraction on the basis of microarray gene-expression data. In: PNAS. Volume 99. (2002) 6562–6566
2. Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon cancer tissues probed by oligonucleotide arrays. In: Natl. Acad. Sci. USA, USA (1999) 6745–6750
3. Tibshirani, R., Hastie, T., Narasimhan, B., Chu, G.: Diagnosis of multiple cancer types by shrunken centroids of gene expression. In: National Academy of Sciences of the United States of America. Volume 99., USA (2002) 6567–6572
4. Black, M.: Statistical analysis of gene expression microarray data. Lecture note for Advanced Bioinformatics 2 (BIOSCI 744), Auckland, The University of Auckland (2004)
5. Kecman, V.: Learning and soft computing : Support vector machines, neural networks, and fuzzy logic models. Complex adaptive systems. MIT Press, Cambridge, Mass. (2001)
6. Huang, T., Kecman, V.: Bias b in svms again. In: 12th European Symposium on Artificial Neural Networks, Bruges, Belgium (2004)
7. Kecman, V., Vogt, M., Huang., T.: On the equality of kernel adatron and sequential minimal optimization in classification and regression tasks and alike algorithms for kernel machines. In: 11th European Symposium on Artificial Neural Networks, Bruges, Belgium (2003) 215–222
8. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning **46** (2002) 389–422
9. Rakotomamonjy, A.: Variable selection using svm-based criteria. Journal of Machine Learning (2003) 1357–1370

High-Throughput Multi-dimensional Scaling (HiT-MDS) for cDNA-Array Expression Data

M. Strickert[1], S. Teichmann[2], N. Sreenivasulu[1], and U. Seiffert[1]

[1] Pattern Recognition Group, Gene Expression Group,
Institute of Plant Genetics and Crop Plant Research Gatersleben
{stricker, srinivas, seiffert}@ipk-gatersleben.de
[2] University of Osnabrück
steichma@uos.de

Abstract. Multidimensional Scaling (MDS) is a powerful dimension reduction technique for embedding high-dimensional data into a low-dimensional target space. Thereby, the distance relationships in the source are reconstructed in the target space as best as possible according to a given embedding criterion. Here, a new stress function with intuitive properties and a very good convergence behavior is presented. Optimization is combined with an efficient implementation for calculating dynamic distance matrix correlations, and the implementation can be transferred to other related algorithms. The suitability of the proposed MDS for high-throughput data (HiT-MDS) is studied in applications to macroarray analysis for up to 12,000 genes.

Keywords: Multi-dimensional scaling, clustering, gene expression analysis.

1 Introduction

Data analysis is a multi-stage process that usually starts at the raw data inspection from which iteratively more sophisticated data models are formulated. Thus, for unknown data, such as for gene expression levels from the first series of screening experiments, many future assumptions are based upon the groundwork of the initial results. Especially visualization techniques can help to gain basic knowledge about the data. For a decent number of continuous-valued attributes, scatter plots provide a very direct access to the data. For high-dimensional input, linear mapping techniques like the principal component analysis (PCA) or the projection pursuit (PP) can help to focus on data projections which are 'interesting' from the mathematical point of view, e.g. the directions of largest variance or views on the central data mass [3,14]. Self-organizing maps (SOM) are well-established non-linear neural tools for clustering data on a rectangular or a hexagonal lattice of specialized neurons [9]. The SOM, however, compels a data model that is structurally different from linear mappings. Since data are represented by prototypes, the one-to-one correspondence between input data and their low-dimensional counterparts vanishes: in case of Ultsch's emergent SOM there are more prototypes than input points, but usually input samples

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 625–633, 2005.

are mapped to a smaller number of specialized neurons. As a result, prototype-based representations give a more abstract view on the data, which is beneficial for model generalization and simplification but not for revealing details of the data distribution or of the data granularity. A nonlinear method that assigns each data point a related copy in the target space is multi-dimensional scaling (MDS) [10]. Its most common objective for visualization is arranging the low-dimensional copies in such a way that their mutual distances match best the original distances. This optimization with respect to good surrogate locations is realized by stress function minimization for which several alternatives exist [2]. Here, MDS is realized by a stochastic gradient descent on a stress function with superior convergence properties. The proposed high-throughput HiT-MDS method is suitable for embedding very large data sets in a distance-preserving way into Euclidean space, usually with 2 or 3 dimensions for visualization. Using MDS for large data sets is closely related to the suggested speed-up of Sammon's mapping by Pekalska et al. [12] and to Naud and Duch who combine MDS with the data prototypes from a learning vector quantizer [11]. The topic presented here, MDS for gene expression analysis, coincides perfectly with a recent publication of Taguchi and Oono introducing a non-metric MDS variant called nMDS [6]. Like other MDS approaches, the proposed HiT-MDS can be used for non-metric data too, for example, by considering the rank order of mutual data items; also non-symmetric dissimilarity measures and sparse distance matrices can be realized for getting global reconstructions of incomplete local distance information.

2 Improved Multi-dimensional Scaling

The challenge of multi-dimensional scaling lies in the optimization of the free parameters $\hat{\mathbf{x}}^i \in \hat{\mathbf{X}}_{n \times d}$ which are the locations of points $\hat{\mathbf{x}}^i = (\hat{x}^i_1, \ldots, \hat{x}^i_d)$ in the d-dimensional target space corresponding to $i = 1 \ldots n$ input points $\mathbf{x}^i \in \mathbf{X}_{n \times q}$. The most canonic approach to forcing the mutual distances $\hat{\mathrm{d}}_{ij} = \mathrm{d}(\hat{\mathbf{x}}^i, \hat{\mathbf{x}}^j)$ of all data pairs indexed by (i, j) to the original distances $\mathrm{d}_{ij} = \mathrm{d}(\mathbf{x}^i, \mathbf{x}^j)$ is defined by minimizing the raw stress function of classical scaling [5]:

$$\mathrm{s} = \sum_{i<j}^{n} (\mathrm{d}_{ij} - \hat{\mathrm{d}}_{ij})^2 \overset{!}{=} \min \quad \text{with distances} \quad \mathrm{d}_{ij}(\mathbf{x}^i, \mathbf{x}^j) = \sum_{k=1}^{d} (x^i_k - x^j_k)^2 .$$

For computational convenience, the squared Euclidean distance is considered, and symmetry $\mathrm{d}_{ij} = \mathrm{d}_{ji}$ can be assumed for the distance matrix. However, as pointed out by Basalaj [1], the straight-forward minimization of the stress by gradient descent is prone to getting stuck in local optima. For Euclidean distances though, eigenvalue methods can be used to find a solution, making classical scaling equivalent to PCA [5]. The freely available `XGvis` package by Buja et al. [2] implements different types of MDS for which eigenmethods fail. `XGvis` comes with a very good documentation of the technical details. Their stress function is of the general form $\mathrm{s} = (1 - \cos^2_{\varphi})^{\frac{1}{2}}$ which is based on the squared cosine between the original distance matrix $\mathbf{D}$ and the reconstructed distance matrix $\hat{\mathbf{D}}$; thereby, the variables in $\boldsymbol{\varphi}$ are parameters of the used Minkowski metric, for data weighting, and for the specification of the order of the moments of the original distances.

Here, an approach is taken which less versatile than the one of Buja et al., but which yields very faithful overall data representations: such most 'honest' data displays are aimed at by maximizing the Pearson correlation of distances:

$$\mathsf{r}(\mathbf{D},\hat{\mathbf{D}}) = \frac{\sum_{i\neq j}^{n} (\mathrm{d}_{ij} - \mu_{\mathbf{D}}) \cdot (\hat{\mathrm{d}}_{ij} - \mu_{\hat{\mathbf{D}}})}{\sqrt{\sum_{i\neq j}^{n} (\mathrm{d}_{ij} - \mu_{\mathbf{D}})^2} \cdot \sqrt{\sum_{i\neq j}^{n} (\hat{\mathrm{d}}_{ij} - \mu_{\hat{\mathbf{D}}})^2}} =: \frac{\mathscr{B}(\hat{\mathrm{d}})}{\sqrt{\mathscr{C}} \cdot \sqrt{\mathscr{D}(\hat{\mathrm{d}})}} \quad \in [-1;1]$$

Matrix $\mathbf{D} = (\mathrm{d}_{ij})_{i,j=1\ldots n}$ contains pattern distances and $\hat{\mathbf{D}} = (\hat{\mathrm{d}}_{ij})_{i,j=1\ldots n}$ those of the reconstructions. Maximizing r yields a balanced consideration of all distance matrix entries; up to the author's knowledge, this maximization cannot be solved with eigenmethods, therefore inducing the proposed heuristic solution. The right fraction is a convenient one-to-one correspondence to the left term used in the following: $\mathscr{B}(\hat{\mathrm{d}})$ is related to the mixed summation of both original and reconstructed distances, $\mathscr{D}(\hat{\mathrm{d}})$ refers to the dissimilarities dependent on the choices of the reconstructions $\hat{\mathbf{X}}$, and $\mathscr{C}$ denotes the connection to the initially calculated and thus constant input pattern distances.

In order to formulate an efficient correlation-based stress function for minimization, the tight range of the Pearson correlation $[-1;1]$, -1 denoting anti-correlation, 0 uncorrelatedness, and 1 perfect correlation, is swapped and widened by inverse power transform:

$$\mathrm{s} = \big(\mathsf{r}(\mathbf{D},\hat{\mathbf{D}})\big)^{-K} \text{ for Euclidean distances } \hat{\mathbf{D}} = \Big(\sqrt{\textstyle\sum_{l=1}^{d} (\hat{x}_l^i - \hat{x}_l^j)^2}\,\Big)_{\substack{j\neq i\,,\\ i,j=1\ldots n}} .$$

Integer $K > 0$ are assumed. For even K, best 'inverse' point configurations with anti-correlated distance relationships might be found. This theoretical situation, however, has never been encountered experimentally – maybe the applied random projection initialization of the embedded points prevents such a defect. Thus, no matter which relationships are coded by $\mathbf{D}$, the inverse power r^{-K} can be minimized by optimally arranging the reconstructions $\hat{\mathbf{X}}$ in the Euclidean target space. This is achieved by a gradient descent on the stress function s, which requires finding zeros of the derivatives of s w.r.t. the free parameters $\hat{x}_k^i$:

$$\mathrm{s} = \mathsf{r}^{-K} \circ \hat{\mathbf{D}} \circ \hat{\mathbf{X}} \overset{!}{=} \min \;\Rightarrow\; \frac{\partial \mathrm{s}}{\partial \hat{x}_k^i} = \sum_{j=1\ldots n}^{j\neq i} \frac{\partial \mathsf{r}^{-K}}{\partial \hat{\mathrm{d}}_{ij}} \cdot \frac{\partial \hat{\mathrm{d}}_{ij}}{\partial \hat{x}_k^i} \overset{!}{=} 0,\ i = 1\ldots n \qquad (1)$$

Solutions are found by iterative updates $\Delta\hat{x}_k^i = -\gamma \cdot \frac{\partial \mathrm{s}}{\partial \hat{x}_k^i}$ of step size γ into the direction of the steepest gradient of s. The missing derivatives in Eqn. 1 are

$$\frac{\partial \mathsf{r}^{-K}}{\partial \hat{\mathrm{d}}_{ij}} = \frac{\partial \frac{(\mathscr{C}\cdot\mathscr{D}(\hat{\mathrm{d}}))^{\frac{K}{2}}}{\mathscr{B}(\hat{\mathrm{d}})^K}}{\partial \hat{\mathrm{d}}_{ij}} = K \cdot \mathsf{r}^{-K} \cdot \frac{(\hat{\mathrm{d}}_{ij} - \mu_{\hat{\mathbf{D}}}) \cdot \mathscr{B}_{ij}^{-}(\hat{\mathrm{d}}) - (\mathrm{d}_{ij} - \mu_{\mathbf{D}}) \cdot \mathscr{D}_{ij}^{-}(\hat{\mathrm{d}})}{\mathscr{B}(\hat{\mathrm{d}}) \cdot \mathscr{D}(\hat{\mathrm{d}})}$$

$$\mathscr{B}_{ij}^{-}(\hat{\mathrm{d}}) = \mathscr{B}(\hat{\mathrm{d}}) - (\mathrm{d}_{ij} - \mu_{\mathbf{D}}) \cdot (\hat{\mathrm{d}}_{ij} - \mu_{\hat{\mathbf{D}}})\,, \quad \mathscr{D}_{ij}^{-}(\hat{\mathrm{d}}) = \mathscr{D}(\hat{\mathrm{d}}) - (\hat{\mathrm{d}}_{ij} - \mu_{\hat{\mathbf{D}}})^2$$

$$\frac{\partial \hat{\mathrm{d}}_{ij}}{\partial \hat{x}_k^i} = 2 \cdot (\hat{x}_k^i - \hat{x}_k^j) \Big/ \sqrt{\textstyle\sum_{l=1}^{d} (\hat{x}_l^i - \hat{x}_l^j)^2} = 2 \cdot (\hat{x}_k^i - \hat{x}_k^j) \,/\, \hat{\mathrm{d}}_{ij} \ .$$

The embedding procedure is outlined in Algorithm 1. High-dimensional data items are assumed to be embedded. They are used for the initialization of their low-dimensional counterparts by random linear mapping. Random mappings provide a first-shot for loosely distance-preserving first target point configurations [8]; here, the matrix elements of R (Alg. 1, l. 2) are randomly taken from $[-1;1]$. Input data are also used for calculating the distance matrix $\mathbf{D}$ which, in an alternative initialization setup, could be directly taken from file. For computational convenience the mean is subtracted from $\mathbf{D}$, which does not affect the Pearson correlation computations $\mathsf{r}(\mathbf{D},\hat{\mathbf{D}})$. Target point adaptation is done iteratively in a neural network training manner: in each step a random point is taken its distance relations to the other points is improved by changing its coordinates. Convergence takes place when for all points the stress function cannot be further minimized, i.e. when the correlation $\mathsf{r}(\mathbf{D},\hat{\mathbf{D}})$ is maximum.

Algorithm 1 HiT-MDS

```
 1: Read input data $\mathsf{X}$.
 2: Initialize $\hat{\mathsf{X}}$ by random projection $\hat{\mathsf{X}}_{n\times d} = \mathsf{X}_{n\times q}\cdot \boldsymbol{R}_{q\times d}$.
 3: Calculate distance matrix $\mathbf{D}$ and subtract its mean $\Rightarrow$ constant $\mathscr{C}$.
 4: Calculate $\hat{\mathbf{D}} \Rightarrow$ initial $\mathscr{B}(\hat{\mathrm{d}}), \mathscr{D}(\hat{\mathrm{d}})$.
 5: repeat
 6:    Draw a pattern index $1 \le i \le n$ from randomly shuffled list.
 7:    for all $j \neq i$ do
 8:       $\Delta\hat{\mathsf{x}}^i \leftarrow \Delta\hat{\mathsf{x}}^i - \frac{\partial \mathsf{r}^{-K}}{\partial \hat{\mathrm{d}}_{ij}} \cdot \frac{\partial \hat{\mathrm{d}}_{ij}}{\partial \hat{\mathsf{x}}^i}$ { accumulate derivatives corresponding to Eqn. 1 }
 9:    end for
10:    $\hat{\mathsf{x}}^i \leftarrow \hat{\mathsf{x}}^i + \gamma \cdot \Delta\hat{\mathsf{x}}^i$ { adapt location of target point }
11:    Recalculate distances $\hat{\mathrm{d}}(\hat{\mathsf{x}}^i, \hat{\mathsf{x}}^j)$ influenced by new position of point $\hat{\mathsf{x}}^i$;
12:    thereby, keep track of changes in $\mathscr{B}(\hat{\mathrm{d}})$ and $\mathscr{D}(\hat{\mathrm{d}})$.
13: until convergence criterion is met.
14: Postprocess: center and normalize $\hat{\mathsf{X}}$ by largest dimension variance.
```

Substantial Speedup is obtained if, for each update step, the distance matrix correlation is not completely recalculated but incrementally adjusted. An unoptimized implementation would take an order of $\mathcal{O}(n^2)$ matrix elements into consideration, but since only $n-1$ distances $\hat{\mathrm{d}}_{ij}$ change during each iteration on pattern i, much computing power is saved by tracking only changes $\Delta\hat{\mathrm{d}}_{ij}$ in the constituents $\mathscr{B}(\hat{\mathrm{d}})$ and $\mathscr{C}(\hat{\mathrm{d}})$ of r. Generally, new values are expressed by means of existing old values plus adding the specific changes caused by the adaptation of the i-th target point. Reformulating the average target distance yields:

$$\mu(\hat{\mathbf{D}}^{new}) = \mu(\hat{\mathbf{D}}^{old} + \Delta\hat{\mathbf{D}}^{i}) = \mu(\hat{\mathbf{D}}^{old}) + \mu(\Delta\hat{\mathbf{D}}^{i}), \quad \Delta\hat{\mathbf{D}}^{i} = (\Delta\hat{\mathrm{d}}_{kj})_{k,j=1\ldots n}$$

with zero elements of $(\Delta\hat{\mathrm{d}}_{kj})$ for $k \neq i$. In the same manner, incremental update formulas can be given for the constituents of the correlation r and thus for the terms in the stress function s that depend on the target distances (Alg. 1, l. 12):

$$\mathscr{B}_{new}(\hat{\mathrm{d}}) = \mathscr{B}_{old}(\hat{\mathrm{d}}) + \sum_{j=1\ldots n}^{j\neq i} \Delta\hat{\mathrm{d}}_{ij} \cdot \mathrm{d}_{ij} \ , \quad \mathscr{D}_{new}(\hat{\mathrm{d}}) = \mathscr{D}_{old}(\hat{\mathrm{d}}) + (n^2 - n) \cdot \mathscr{T}$$

$$\text{with } \mathscr{T} := \mu(\Delta\hat{\mathbf{D}}^{i})^2 \cdot \sum_{j=1\ldots n}^{j\neq i} \Delta\hat{\mathrm{d}}_{ij} \cdot \Big(\Delta\hat{\mathrm{d}}_{ij} + 2 \cdot \big(\hat{\mathrm{d}}_{ij}^{old} - \big(\mu(\hat{\mathbf{D}}^{old}) + \mu(\Delta\hat{\mathbf{D}}^{i})\big)\big)\Big)$$

MDS with the new stress function in combination with the above speedup formulas is designed for high-throughput operation and it is therefore referred to as HiT-MDS.[1] The free parameters of HiT-MDS are the exponent K of the stress function and the step size γ which can be called a learning rate. Empirical studies on different data sets have showed that the proposed algorithm is very robust with respect to the choices of γ and K. A two phase learning, first a rough ordering with $K = 4$ and increasing $\gamma = 10^{-3}, 10^{-2}, 10^{-1}$ followed by a fine tuning with $K = 1$ and $\gamma = 5, 25, 100, 250$ is likely to meet the requirements even of unknown data sets; usually, a number between 10 and 500 training epochs will suffice to obtain good convergence for the current parameter set.

3 Application to cDNA Array Expression Data

Two studies are presented that show the suitability of HiT-MDS for the analysis of high-throughput expression data. In the experimental design, several thousand gene expression patterns were analyzed which correspond to barley seed development in the time span 0–26 days after flowering, in three distinct tissues: pericarp (p), endosperm (end), and embryo (e). The conducted experiments resulted in high-throughput expression data with a relatively small number of experiments and a large number of gene expression levels. The two major questions of interest are: 1. How are the experiments, representing the tissues at a particular developmental stage, characterized with respect to their transcriptome similarity of specifically expressed genes? 2. Can inter-experimental clusters of coexpressed genes be identified for which, afterwards, regulatory functions can be assigned? In order to address these key questions it is necessary to bring the high-dimensional data for visualization into lower dimension, thereby keeping their original distances. HiT-MDS is applied to obtain such visualizations, first for grouping the experiments with similar transcriptome, secondly for identifying the hot-spots of coexpressed genes from unfiltered expression data.

1. Experiment Clustering. The first HiT-MDS application aims at the visualization of the inter-experiment relationships: a two-dimensional embedding is wanted that reconstructs the distances of the original space of gene expression intensities. The common way to do this is principal component analysis (PCA). More recently, independent component analysis (ICA) has been used which is a linear decomposition technique [7], or locally linear embeddings LLE are computed [13]. However, none of these methods is explicitly formulated like MDS for the most natural approach to the original data, which is the best available

[1] A C-language command line version is available at `http://hitmds.webhop.net/`.

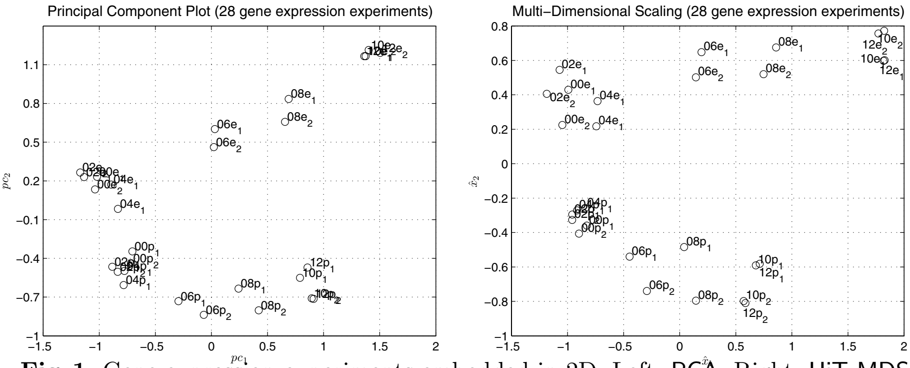


Fig. 1. Gene expression experiments embedded in 2D. Left: PCA. Right: HiT-MDS.

distance reconstruction. The analyzed data set comprises 28 experiments, composing 7 developmental points on from endosperm/embryo (e) and pericarp (p) tissue which are reproduced by two independent experiments (1,2); each experiment is represented by 1,421 log-transformed gene expression values. Fig. 1 compares the experiment visualization of PCA (left) and HiT-MDS (right). Both embeddings show a temporal ordering from 00 to 12 days, but tissues e vs. p are more strictly separated by HiT-MDS, and the $00-04\mathsf{e}_{1/2}$ cluster as well as the $08\mathsf{p}_{1/2}$ pair are more faithfully spread. Distance correlations characterize the reconstruction quality; they are $\mathsf{r}(\mathbf{X}, \hat{\mathbf{X}}_{\mathsf{PCA}}) = 0.973$ and $\mathsf{r}(\mathbf{X}, \hat{\mathbf{X}}_{\mathsf{HiT\text{-}MDS}}) = 0.981$, which supports higher confidence for the HiT-MDS embedding. For these obtained groupings, the identification of regulating key genes will be of interest in future investigations.

2. Gene Clustering. A much more demanding experiment is conducted with data from 12k-arrays (11786 genes) available for 31 developmental stages of three tissue types. The left panel of Fig. 2 shows a density plot of the embedded 31-dimensional genes zoomed to the region of highest interest. Dark patches represent areas of high densities, i.e. many points with small Euclidean distance, leading to an overall correlation of $\mathsf{r}(\mathbf{X}, \hat{\mathbf{X}}_{\mathsf{HiT\text{-}MDS}}) = 0.979$. In contrast, PCA yields a correlation of only $\mathsf{r}(\mathbf{X}, \hat{\mathbf{X}}_{\mathsf{PCA}}) = 0.937$, which would result in misleading interpretations of a PCA-based density plot (omitted). HiT-MDS becomes even more useful for coexpression analysis, if the Euclidean distance measure for the 31-dimensional gene vectors is replaced by (1-correlation) between them; then, the maximum dissimilarity is 2 and the best correlation is 0. Thus, a 2D-embedding of such a dissimilarity matrix will cluster gene profiles of high correlation opposed to anti-correlated groups of genes. This is illustrated in the right panel of Fig. 2 for a preselected subset of 2000 embedded genes. By focusing on these fewer points, i.e. by loosening the global embedding constraints, more details become visible. The gene profiles associated with the points inside the exemplarily picked encircled high-density areas are plotted in Fig. 3. A good spatial separation and specificity is observed for the embedded genes and their corresponding relevance profiles, making the proposed approach a valuable tool for gene expression analysis: it is demonstrated that by HiT-MDS hot-spots of

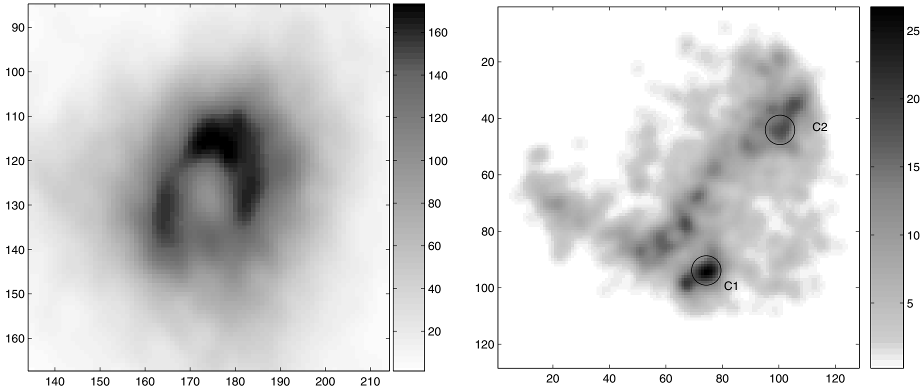


Fig. 2. Genes embedded in 2D. Left: Euclidean inter-gene embedding of 12k-genes. Right: correlation-based similarity on 2000 gene subset with finer clustering details.

coexpressed genes can be identified despite the presence of noise in the expression data by simultaneously processing all available data. Such coexpressed genes will be further examined for their functional role.

4 Conclusions

The proposed improvement of multi-dimensional scaling, HiT-MDS, allows to embed very large data sets into a low-dimensional Euclidean space, making the method suitable for visual data inspection. As a nonlinear method, HiT-MDS is able to capture even minor differences in the data by faithfully reconstructing given distance relations. The corresponding low-dimensional surrogates are obtained by a neural network type of incremental training that minimizes a fast converging stress function. This function is based on the correlations between input and output distances; thereby, functional terms contributing to the correlation calculation are also updated in an incremental way. In contrast to SOM learning, HiT-MDS is no mapping technique, which restricts its applicability to static data sets. However, no topological parameters have to be chosen, consequently HiT-MDS is not subject to topological restrictions of an underlying

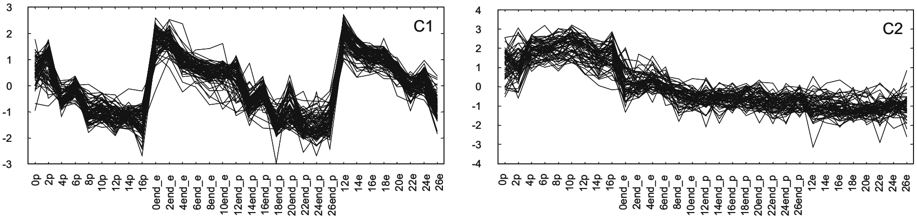


Fig. 3. Gene profiles corresponding to cluster C1 (left) and C2 (right) of Fig. 2.

neuron grid, and the success the data model can be well-evaluated by the correlation between input distances and those of the embedded points. Obviously, HiT-MDS can be applied to many problems beyond bioinformatics, e.g. for the visualization of high-dimensional items, for graph layout [4], and also for space conversion like the illustrated one, from correlation space to Euclidean space. Future attention is put on further improvements of the stress function – first preliminary results show that even faster initial convergence and more tolerant parameter choices are achieved by replacing the presented exponential correlations by Fisher's Z transform. Finally, parallel implementations on an SMP machine are under consideration for making interactive zooming into large subsets of analyzed data easily possible.

Acknowledgement. Thanks to Dr. Volodymyr Radchuk for helping probe preparation for the macroarray experiments. Thanks also to the anonymous reviewers for helping to improve the this contribution by their valuable comments. The work is supported by BMBF grant FKZ 0313115, GABI-SEED-II.

References

1. W. Basalaj. *Proximity Visualization of Abstract Data.* PhD thesis, Computer Lab, Univ. of Cambridge, URL: http://www.pavis.org/essay/pavis.pdf, 2001.
2. A. Buja, D. Swayne, M. Littman, N. Dean, and H. Hofmann. Interactive Data Visualization with Multidimensional Scaling. Report, University of Pennsylvania, URL: http://www-stat.wharton.upenn.edu/~buja/, 2004.
3. D. Cook, A. Buja, and J. Cabrera. Grand tour and projection pursuit. *Journal of Computational and Graphical Statistics*, 4(3):155–172, 1995.
4. E. Gansner, Y. Koren, and S. North. Graph drawing by stress majorization. In *Lecture Notes in Computer Science, Proc. of 12th Int. Symp. Graph Drawing (GD'04).* Springer, 2004.
5. J. Gower. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53:325–338, 1966.
6. Y. h. Taguchi and Y. Oono. Relational patterns of gene expression via non-metric multidimensional scaling analysis. *Bioinformatics*, 21(6):730–740, 2005.
7. A. Hyvärinen, J. Karhunen, and E. Oja. *Independent component analysis.* John Wiley & Sons, 1st edition, 2001.
8. S. Kaski. Dimensionality reduction by random mapping: Fast similarity computation for clustering. In *Proc. of the International Joint Conf. on Neural Networks (IJCNN'98)*, volume 1, pages 413–418. IEEE Service Center, Piscataway, NJ, 1998.
9. T. Kohonen. *Self-Organizing Maps.* Springer-Verlag, Berlin, 3rd edition, 2001.
10. J. Kruskal and M. Wish. *Multidimensional Scaling.* Sage Publications, New Park, California, 1978.
11. A. Naud and W. Duch. Visualization of large data sets using MDS combined with LVQ. In L. Rutkowski and J. Kacprzyk, editors, *Advances in Soft Computing*, pages 632–637. Physica Verlag, 2002.
12. E. Pekalska, D. deRidder, R. Duin, and M. Kraaijveld. A new method of generalizing Sammon mapping with application to algorithm speed-up. In M. Boasson, J. Kaandorp, J. Tonino, and M. Vosselman, editors, *Proc. 5th Annual Conf. of the Advanced School for Computing and Imaging*, pages 221–228. ASCI, 1999.

13. S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.
14. M. Strickert, S. Teichmann, N. Sreenivasulu, and U. Seiffert. 'DiPPP' Online Self-Improving Linear Map for Distance-Preserving Macro-Array Data Analysis. To appear at WSOM 2005.

Comparing Neural Network Architecture for Pattern Recognize System on Artificial Noses

Aida A. Ferreira[1], Teresa B. Ludermir[2], and Ronaldo R.B. de Aquino[2]

[1] Federal Center of Technological Education of Pernambuco, Av. Professor Luiz Freire,
500 Cidade Universitária, Recife-PE, Brazil. Cep:50.740-540
aaf@cin.ufpe.br
[2] Federal University of Pernambuco – Recife – PE - Brazil
tbl@cin.ufpe.br and rrba@ufpe.br

Abstract. Neural networks (NNs) have been used to classify odor patterns and are showing promising results. In this paper we present four different models of NNs to implement pattern recognition system in artificial noses (ANs). These models were analyzed comparing the classification error in the test set and using a hypothesis test. The models investigated are probabilistic neural network and multi-layer perceptrons with and without temporal processing. We used a complex data base with 9 different classes for evaluation the classifiers.

1 Introduction

Since the 80s, researches to create ANs, which will detect and classify odors automatically, have advanced significantly. AN can be used for monitoring the environment in order to control the quality of the air, in the health field to help diagnosing diseases and in the food, drink and cosmetics industry to control quality and monitor the production process.

An AN is a modular system, which consists of two main parts: a sensor system, made up of elements that detect odor, and a pattern recognition system that classifies the detected odors. NNs have been used as pattern recognition system and have showed good results.

Different NNs architectures have been used on ANs pattern recognition systems. The Multi-layer Perceptron (MLP), Radial Base Function (RBF) and Time Delay Neural Network (TDNN) architectures have been used to recognize different harvests of the same red wine in different works [1-3,9] and they have achieved promising results. NNs have also been used successfully to classify petrol-derived odors [3].

The main objective of this paper is to presents an experimental comparison for different NN architectures. The probabilistic neural network (PNN) and MPLs with and without temporal processing were chosen to create pattern recognition systems on ANs. The MLP was selected because it has been previously used in other works with success. The PNN architecture was chosen because it is indicated to classifying problems and for the fact that it has never been used before to create pattern recognition systems on ANs. We decided to investigate the MLP and PNN with temporal processing because the temporal processing has been previously used in other works with success.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 635 – 640, 2005.

2 Data Base

The problem dealt with in this work consists of classifying 9 different samples of turpentine English Candle (reference sample, croqueo naphtha with 100, 500 e 1000ppm of contamination, craqueo naphta with 100, 500 and 1000ppm of contamination, diesel oil and TBQ46) that were made available by a petrol refinery. The data was obtained through the use of an AN prototype [3]. This prototype is composed of eight polymer conductors sensors prepared with different dopant.

For each type of turpentine were done up to 8 acquisitions. On each acquisition the sensor resistance values were recorded every 20 seconds. As each acquisition lasted about 10 minutes, each sensor obtained an average of 30 values for each turpentine type on each acquisition. With the objective of using balanced information of each turpentine type (class), the acquisitions 5 and 6 of sample contaminated in croqueo with contamination level of 500ppm and 1000ppm were replicated until they had the same acquisition number as the other classes. For the same reason, the values of both the acquisition 2 contaminated in the coqueo with level of contamination 1000ppm and the acquisition 2 of TBQ46 turpentine were replicated until there were 30 values.

3 Models and Methodology of the Experiments

To create MLP and PNN without temporal processing, the set formed by the eight sensors values in the same time interval was considered a data base pattern. So, the database has 2160 patterns, which correspond to 30 patterns in eight acquisitions for each one of the nine turpentine types. To create MLP and PNN with temporal processing, a new database was created from the same original data, in a way that each pattern started to represent the eight sensors resistance in a time period (t) and the eight sensors resistance to the same substance in a time period (t+1).

In order to obtain an error classification estimate closer to the true error, the 10-fold cross-validation with stratification method was chosen to create the training sets and the test sets. This method has become a standard method in practical terms [5]. The patterns were divided in 10 independent portions and with stratification (same quantity of each class patterns in each portion); each portion has 10% of the data. In each experiment a portion was used to test the network and the nine portions left were used to train the networks.

The whole experiment was done in the Matlab tool. All the networks created without temporal processing consist of eight units in the input layer, one for each sensor. The networks output is represented by the codification 1-of-m, a unit for each turpentine type, being so, and the networks have nine units in the output layer.

All the networks created with temporal processing consist of sixteen units in the input layer, eight to represent each of the eight sensors in the time *t* and eight to represent the same eight sensors in the time *t+1*. The networks output is also represented by the codification 1-of-m, and then networks have nine units in the output layer. The number of nodes in the intermediary layer in PNN were not choose because the training algorithmic creates one node for each pattern in training set. The number of nodes in the hidden layer in MLP was between 8 and 26 because. The training algorithm is slow and networks with more than 26 nodes in hidden layer had become very difficult to train because they take a long time for processing. The hidden and the output layers nodes of the MLPs have logistic sigmoid activation function.

4 Experiments Results

4.1 PNN and MLP Without Temporal Processing

The PNNs implement the Bayesian decision strategy to classify input vectors [4]. The PNNs are divided in four layers: input unit layer, pattern unit layer, sum unit layer and output unit layer. The pattern layer units represent each training set pattern and use a radial base activation function. The pattern layer does a non-linear transformation of the input space to the hidden space; on most applications, the hidden space is of high dimensionality, so transforming the non-linear separable pattern set in linear separable output sets. The sum layer inputs have origin in correspondent pattern layer units from a determined class. The output layer units produce a binary output; which is equal to 1 in only one of the units 0 in the others.

With the objective of reducing the classification error of the PNNs, different radial base function width values were investigated (Spread parameter). Table 1 illustrates the results obtained from each Spread value investigated.

Table 1. PNNs Results

Spread	Node quantity		% Classif. Error Training		% Classif. Error Test	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
0.0001	1944	0.00	0.09	0.02	24.86	2.94
0.0005	1944	0.00	0.09	0.02	1.16	0.59
0.0050	1944	0.00	1.54	0.17	2.69	1.21
0.0500	1944	0.00	31.83	1.37	35.83	3.86
0.1000	1944	0.00	44.69	2.42	48.84	2.84

The MLPs were created with only one hidden layer. The MLPs were trained with the Leavenberg-Marquardt algorithm described in [7]. Initially, ten experiments were done with random weight initialization using 8, 10, 12, 14, 16, 18, 20, 22, 24 e 26 nodes in the hidden layer to select the best architecture to the problem. The architecture with 16 nodes in the hidden layer was chosen because it presented the smallest classification error in the validation set. The maximum number of iterations defined for all the trainings was 2500. The training stops if the Early Stop criteria [6] implemented by Matlab occurs 5 consecutive times, or if the maximum number of iterations is reached or, still, if the error in the training set is equal to zero.

For each training, validation and test set, formed from the portions created by the cross-validation method; 10 networks were created. Table 2 illustrates the mean results obtained from the best MLP architecture (16 nodes). The best network was the number 28 because it presented the smallest MSE in test set, equal *1.1451*.

Table 2. MLPs Results

Experiment	%Classif. Error Training		%Classif. Error Validation		%Classif. Error Test	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
1	46.84	33.85	47.50	32.64	47.82	33.37
2	13.29	19.83	14.81	19.25	13.94	19.80
3	32.69	31.50	34.01	30.66	33.38	31.10
4	25.85	27.80	27.05	27.37	27.59	26.91
5	28.99	28.35	31.22	27.56	30.93	27.51
6	42.62	34.77	43.60	33.98	43.47	34.20
7	29.52	25.95	31.37	24.93	31.02	25.87
8	45.44	34.58	47.10	33.58	47.18	33.85
9	18.20	22.52	19.58	21.51	20.00	21.56
10	28.39	33.51	29.35	32.82	30.19	32.50
Mean	31.18		32.56		32.55	
Std. Dev.	11.13		10.98		11.09	

4.2 PNN and MLP with Temporal Processing

The TDNNs with PNN were created using the same training and the same parameters for PNN without temporal processing. The PNNs created in this phase have 16 nodes in the input layer, 1863 nodes in the hidden layer and 9 nodes in the output layer. Table 3 describes results obtained for these networks.

Table 3. TDNNs with PNN Results

Spread	Node Quantity		%Classif. Error Training		% Classif. Error Test	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
0.05	1863	0.00	0.05	0.02	50.43	2.58
0.08	1863	0.00	0.05	0.02	4.30	1.26
0.10	1863	0.00	0.48	0.09	1.74	0.92
0.50	1863	0.00	28.52	1.29	30.05	3.57
1.00	1863	0.00	37.27	1.17	40.77	3.93

The TDNNs with MLP were created using the same training and the same parameters for MLP without temporal processing. The 26 nodes architecture in the hidden layer was chosen because it presents the smallest classification error in the validation set. Table 4 presents the results. The best network was the number 59 because it presented the smallest MSE in test set, equal *0.6725*.

A comparative study based in each model classification performance was made to show the advantages and the disadvantages of each model. Table 5 identifies the topology and the mean classification error obtained in the training, validation (when there is) and test sets and Table 6 describes the results from hypotheses test.

The PNNs presented a mean classification error of 1.16% in the test set. This error was the smallest error obtained among all created systems and this results were

Table 4. TDNN with MLP Results

Experiment	%Classif. Error Train		%Classif. Error Valid.		%Classif. Error Test	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
1	31.49	40.13	32.91	39.18	32.56	39.63
2	41.63	40.14	42.58	39.01	43.19	39.59
3	33.27	33.39	34.64	32.88	34.54	32.79
4	32.82	40.11	34.12	39.27	33.91	39.44
5	27.10	32.13	28.10	31.46	29.52	31.41
6	31.05	36.52	32.40	35.69	32.17	35.81
7	15.25	17.18	17.71	17.50	17.83	17.77
8	30.95	36.43	32.13	35.90	32.90	35.72
9	13.32	14.74	14.72	14.66	13.91	13.83
10	31.23	41.25	32.30	40.38	32.90	39.81
Mean	28.81		30.16		30.34	
Std. Dev.	8.50		8.22		8.46	

Table 5. Results from the best topologies

Topology	%Classif. Error Train	%Classif. Error Valid	%Classif. Error Test
PNN	0.09	-	1.16
MLP	31.18	32.56	32.55
TDNN with PNN	0.48	-	1.74
TDNN with MLP	28.81	30.16	30.34

Table 6. Results from hypotheses test

Hypothesis 0: Classify 1 = Classify 2 Test Significance: 5%		Hypothesis 1: Classify 1 < Classify 2		
Classify 1	Classify 2	Decision	p-value	Confidence interval
PNN	MLP	Reject H0	0.000000	-25.31
PNN	TDNN com PNN	Reject H0	0.000001	-2.38
PNN	TDNN com MLP	Reject H0	0.000000	-24.54
MLP	TDNN com PNN	Accept H0	1.000000	34.37
MLP	TDNN com MLP	Accept H0	0.688660	9.86

confirmed with a hypothesis tests in a significance level of 5%[8]. Besides having presented the smallest error on the test set, the PNNs presented the advantages of having the smallest training time and reduced quantity of parameters that were investigated. On the other hand, the PNN presented a disadvantage that is the necessity of creating a node in the hidden layer for each input pattern. This disadvantage can become a problem, if the created system has to be used in equipment with less computational resources.

The MLPs (with and without temporal processing) performance was not superior to any other network. The principal advantage of using MLPs is that the created systems presented the smallest number of nodes in the hidden layer, but the quantity of parameters that can be investigated and the training time are disadvantages.

The temporal information inclusion did not improve the TDNN with PNN networks performance in relation to the PNN networks. A probable explanation for the performance of these networks not being superior to the PNN networks performance is because as the input data dimension doubled more training patterns would be necessary, or it would be necessary to apply some techniques to reduce input pattern dimensionality.

The MLP with TDNN error was inferior to the MLP mean error; in other words, the temporal processing helped to improve the odor pattern classification performance with MLP. The hypothesis tests showed that this improvement was not enough to affirm that the TDNN with MLP classifiers are better than the MLP classifiers with a 5% significance level.

5 Conclusion

We decided to analyze the PNN because PNN was not yet applied to the odor pattern recognition problem and also considering the training simplicity and quickness of these networks.

A new and more complex data base use was important to prove the applicability of the MLP and TDNN architecture in odor patterns recognition problems.

From the results obtained we can conclude that the creation of odor pattern recognition systems for an AN, with PNN networks would allow a fast equipment update to new data bases, once the training from these networks is almost immediate and few adjustable parameters need to be investigated. However, if the equipment has few computational resources, it would be more indicated to use the MLP architecture. MLP allows networks creation with fewer nodes in the hidden layer, though it is necessary much more time to train these networks.

References

[1] M.S. Santos et al, “Artificial Nose and Data Analysis Using Multi Layer Perceptron” In Data Mining, WIT Press, Computational Mechanics Publication, 1998.

[2] A. Yamazaki, T.B. Ludermir and M.C.P. de Souto, "Classification of vintages of wine by an artificial nose using time delay NNs", IEE Electronics Letters, Vol. 37 (24), pp. 1466-1467.

[3] M.S. Santos, Construction of an artificial nose using NNs. (In Portuguese) Ph.D. Thesis, Centre of Informatics, Federal University of Pernambuco, Brazil, 2000.

[4] D.F. Specht, Probabilistic NNs and the Polynomial Adaline as Complementary Techniques for Classification, IEE Transactions on Neural Networks, 1990, Vol. 1, pp. 111-121.

[5] I.W. Witten, E. Frank, Editor D.D. Cerra, Data Mining, Practical Machine Learning Tools and Techniques with Java Implementations, 2000, Morgan Kaufmann Publishers, pp. 126.

[6] L. Prechelt, Proben1 – A Set of NN Benchmark Problems and Benchmarking Rules. Technical Report 21/94, Fakultät für Informatik, Universität Karlsruhe, Germany, 1994.

[7] R. Fletcher, Practical Methods of Optimization, Wiley, 1987.

[8] Johson R. A., Wichern D. W.: Applied Multivariate Statistical Analysis. Prentice Hall, (1999).

[9] A. Yamazaki and T. B. Ludermir. “Neural Network Training with Global Optimization Technique”. International Journal of Intelligent Systems, Vol.13 (2), p.77 - 86, 2003.

Medical Document Categorization Using *a Priori Knowledge*

Lukasz Itert[1,2], Włodzisław Duch[2,3], and John Pestian[1]

[1] Department of Biomedical Informatics, 3333 Burnet Avenue, Children's Hospital Research Foundation, Cincinnati, OH 45229, USA
[2] Department of Informatics, Nicolaus Copernicus University, Toruń, Poland
[3] School of Computer Engineering, Nanyang Technological University, Singapore

Abstract. A significant part of medical data remains stored as unstructured texts. Semantic search requires introduction of markup tags. Experts use their background knowledge to categorize new documents, and knowing category of these documents disambiguate words and acronyms. A model of document similarity that includes *a priori* knowledge and captures intuition of an expert, is introduced. It has only a few parameters that may be evaluated using linear programming techniques. This approach applied to categorization of medical discharge summaries provided simpler and much more accurate model than alternative text categorization approaches.

1 Introduction

The dream of semantic Internet populated with documents annotated with XML tags remains a difficult challenge. Automatic tools that convert unstructured textual data into semantically-tagged documents are still elusive. In the medical domain the need to create these tools is acute because errors may be costly, medical vocabularies are abbreviations and acronyms are rampant. Critical differences between General English and Medical English have been analyzed in a numbers of publications [1]. The "Discovery System" (DS) data repository [2] at the Cincinnati Childrens Hospital Medical Center (CCHMC), a large pediatric academic medical center with over 700,000 pediatric patient encounters per year, contains terabytes of medical data, mostly in form of raw texts, stored in a complex, relational database integrating many electronic hospital services.

The long-term goal of our research is to create tools that automatically annotate unstructured medical texts, adding full information about all medical concepts, ambiguous terms, expanding acronyms and abbreviations, using a variety of statistical and computational intelligence algorithms to achieve this goal. The first step towards full semantic annotation and disambiguation of medical text requires discovery of the document topic, for example the main disease that has been treated. It is clear that medical expert reading a given text quickly forms a hypothesis about the particular sub-domain the text belongs to and interprets the text in the light of the background knowledge derived from medical studies, textbooks and individual experience. This is especially true if relatively short

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 641–646, 2005.

texts, such as patient's discharge summaries, containing brief medical history, current symptoms, diagnosis, treatment, medications, therapeutic response and outcome of hospitalization, are analyzed. Many medical concepts appear very rarely in such short documents, therefore document categorization algorithms that ignore background medical knowledge make many errors.

In the next section a model trying to capture expert intuition in document categorization is introduced and a simple way to take the *a priori* knowledge into account proposed. Estimation of parameters of this model is done using linear programming techniques. Numerical experiments with over 4500 discharge summaries were made to compare this approach with standard document categorization methods.

2 Model of Similarity

Documents D_j of length l_j are composed of terms (words, collocations or concepts). Term frequencies tf_{ij} for term $i = 1 \ldots n$ in document j are calculated for all documents, and transformed to obtain features that help to reflect document similarity. Weights of features that appear with high frequency, or are derived from longer documents, should be reduced using logarithmic or square root functions. Uniqueness of each feature is inversely proportional to the number of documents this feature appears in; if the term i appears in df_i out of N documents weighting for non-zero term frequencies may be calculated as [3]:

$$s_{ij} = (1 + \log tf_{ij}) \log N/df_i \tag{1}$$

In the tf × idf weighting scheme additional scaling is used, for example [3]:

$$s_{ij} = \text{round}\left(10 \times \frac{1 + \log tf_{ij}}{1 + \log l_j} \log \frac{N}{df_i}\right) \tag{2}$$

In document categorization distribution of a given term among different categories is important, therefore the logarithm of ratio $\log(K/cf_i)$ of the number of classes K to the number of classes cf_i in which term i appears, should be used in the above equation. To avoid favoring long documents all vectors $(s_{1j}, \ldots s_{nj})$ may be divided by their length to obtain final feature vectors $\mathbf{x}_{ij}$, for example:

$$\mathbf{x}_{ij} = (1 + \log tf_{ij}) \log N/df_i; \qquad \mathbf{x}_j = \mathbf{s}_j/||s_j|| \tag{3}$$

This normalization tends to favor shorter documents. More sophisticated normalization methods have been introduced to counter this effect, but unbiased normalizations are hard to find.

Such *ad hoc* term weights do not take into account *a priori* knowledge. Before the document is examined the probability that it belongs to category C_k should be equal to the prior probability $p(C_k)$. The background knowledge about reference documents from class C_k may be represented using weighted frequencies $R_{ik} = R_k(tf_i)$ for the term i. These frequencies are collected in the reference vector R_k (more than one vector per class may be needed). The following algorithm seems to capture human intuitions of the document categorization process:

1. Initial distance between document D and the reference vectors R_k should be proportional to $d_{0k} = ||D - R_k|| \propto 1/p(C_k) - 1$.
2. If a term i appears in R_k with frequency $R_{ik} > 0$ but does not appear in D the distance $d(D, R_k)$ should increase by $\varDelta_{ik} = a_1 R_{ik}$.
3. If a term i does not appear in R_k but it has non-zero frequency D_i the distance $d(D, R_k)$ should increase by $\varDelta_{ik} = a_2 D_i$.
4. If a term i appears with frequency $R_{ik} > D_i > 0$ in both vectors the distance $d(D, R_k)$ should decrease by $\varDelta_{ik} = -a_3 D_i$.
5. If a term i appears with frequency $0 < R_{ik} \leq D_i$ in both vectors the distance $d(D, R_k)$ should decrease by $\varDelta_{ik} = -a_4 R_{ik}$.

Coefficients $a_1, \ldots a_4 > 0$ are adaptive constants. If a term appears in both D and R than the distance is decreased by a constant times the smaller of the two frequencies, because for small term frequencies this situation may happen by pure chance. A term that appears only in documents from the C_k class should be more important for this class than terms appearing in all classes, therefore term specificity is given by the class-conditional probability $p(i|C_k) = p(tf_i > 0|C_k)$. Given the document D, and reference vector R_k, probability that the class is C_k should be proportional to:

$$S(C_k|D; R_k) = 1 - \sigma\left(\beta\left[d_{0k} + \sum_i p(i|C_k)\varDelta_{ik}\right]\right) \tag{4}$$

Here $\varDelta_{ik}$ depends on adaptive parameters $a_1, \ldots a_4$ that may be specific for each class, and the distance depends on the d_{0k} which may also be treated as an adaptive parameter; the slope β is an additional parameter, giving 6 adaptive parameters per class. Weighted distance contributions may sum to a negative number therefore a logistic function $\sigma(\cdot)$ is used. Probabilities are estimated after softmax normalization $p(C_k|D; R_i) = S(C_i|D; R_i)/\sum_k S(C_k|D; R_k)$.

This approach seems to capture some human intuitions when texts are analyzed using background knowledge. Parameters $a_1, \ldots a_4$ may be estimated using neural networks with RBF-like architecture and $S(C_i|D; R_i)$ functions (4) in each hidden node i, and a soft-max function for the output node. An alternative is to use linear programming techniques for parameter optimization, solvable in polynomial time using interior point based methods. PCx algorithm has been used here [6]. Condition

$$d_{0k} + \sum_i p(i|C_k)\varDelta_{ik} = \min \tag{5}$$

maximizes similarity between documents and reference vectors, Eq. 4, and should be used with the following constraints:

$$\sum_i p(i|C_j)\varDelta_{ij} - \sum_i p(i|C_k)\varDelta_{ik} \geq d_{0k} - d_{0j}; \quad k \neq j = 1 \ldots K \tag{6}$$

where k indicates the correct class. For all N training vectors (documents) $K - 1$ constraints are created. Two cases have been considered: a common set of $a_1, \ldots a_4$ parameters for all classes, and a separate set for each class. Satisfying all $K-1$ inequalities for one document D guarantees that its similarity measure (4) is maximal for the correct class and provides correct classification.

3 Numerical Experiments

Customized SQL queries were created to retrieve discharge summaries from the database. Overall 4534 patients discharge summary records were used. All documents are short, less than 3000 characters, with the average length below 2000 characters. They are labeled by 10 distinct disease names, with "asthma" being the majority class that covers 19.1%, followed by Epilepsy (14.1%), Pneumonia (13.4%), Gastroenteritis (12.9%), Anemia (12.0%), Otitis media (10.8%), Urinary tract infection (UTI) (6.6%), Cystic fibrosis (6.2%), Cerebral palsy (3.9%), and the Juvenile Rheumatoid Arthritis (JRA) with 0.9%. Except for the last class that contained only 41 documents all the other classes were among the most common in the database containing discharge records.

The name of the disease used as the category label plays a dual role: it is one of the features used to describe the document, and it is also the class label. For example, documents from the "asthma" class frequently contain the name "asthma" as a part of some concept (such as "allergic asthma"), but they may also contain the names of other diseases. The frequency of appearance of each of the 10 disease names in the documents may be taken as an indicator of the class, giving a more informed base rate distribution. Using this approach leaves 55.3% of documents unclassified (including ties with several identical highest frequencies), 34.6% correctly classified and 10.1% errors.

To define the feature space each record has to be subject to several text processing techniques: exhaustive sets of parsing rules are used to handle punctuation issues and stop-word list of common English words to remove words that do not contribute to document categorization. MetaMap Transfer (MMTx) program package [5] has been used to discover UMLS Metathesaurus concepts [4] in these texts. To prevent any false-positive mapping a very restrictive MMTx settings has been used during string matching. Concepts are assigned to 135 semantic types, but only 26 types representing specific, medical concepts were found useful for document categorization. They include anatomical structures, body parts, functions, biological organisms, drugs and pharmacological substances, clinical procedures, disease and syndromes, symptoms, and test results. Using the UMLS ontology as a base all common words may be filtered out, and all unnecessary medical terms excluded. The final number of features included in the "native" space based on concepts discovered in medical records was 7220.

The reference texts were taken from MedicineNet [7], Children's Hospital Boston Child Health A to Z [8], and MedlinePlus: Medical Encyclopedia [9]. Documents describing each of the 10 selected diseases have been processed and 1097 unique UMLS concepts have been identified. In the discharge summaries only 807 of these concepts appeared and these concepts have been used as the feature space. Background knowledge contained in features that appear only in the reference space, but not in the limited selection of medical records taken for analysis, could be useful in future for categorization of new texts. Discharge summaries contain many more UMLS concepts than reference texts, but in most cases there is little or no correlation between names of these additional concepts and diseases. Thus *a priori* knowledge helps in feature selection and definition of the feature space.

All calculations presented below were done using 10-fold crossvalidation. Features were based on term frequencies (M0), binary present/absent values, and 4 popular weighting schemes [3]. Poor results of the SSV decision tree [10] (similar results are obtained with C4.5 tree) show that the similarity-based approach may be more appropriate here, and that the reference vectors containing *a priori* knowledge may help. To check how the nearest neighbor classifier performs using a single reference vector per class kNN with a cosine distance function has been used [3]. Direct application of Euclidean distance has no sense because reference vectors have different norms, and the shortest one will almost always be the closest (accuracies are between 6-15%). Best accuracy is obtained with unweighted term frequencies (60.1%), worst accuracy with binary vectors (43.8%) and 56-59% accuracy with M2-M5 tf weightings. A very large neural network is needed (300 neurons and $\sim$ 250 thousand parameters) to reach 71-72% accuracy on this data. SVM has never given such good results, with Gaussian kernel results at the level of 40% only and linear kernels in the range of 60%. Standard deviation was between 1.5-2.5%.

The approach described in Sec. 2 has been used to calculate coefficients $a_1, \cdots a_4$ in each crossvalidation using linear programming techniques. For each test vector these coefficients were used to compute similarity to all 10 reference vectors, selecting the highest similarity as class indicator. In the first case the same coefficients were used for each class. Parameter $\beta = 0.01$ was used, making the logistic transformation almost linear; higher values of β lead to sharp increase in the number of ties. For each crossvalidation (CV) step on average around 95% of all constraints were satisfied, however the number of vectors for which all constraints were fulfilled was only 61%, leading to the classification accuracy of 61.1%. Optimizing coefficients $a_1, \cdots a_4$ separately for each class decreased the percentage of all satisfied constraints to 92%, but increased the number of vectors for which all constraints were fulfilled by approximately 10%. The final CV accuracy was then 71.6±2.1% with tf frequencies and similar for various scalings. This is quite remarkable for a system with 4 parameters per class, considering the improvement over standard feature weighting techniques, and the size of the MLP network needed to reach similar results. Prototypes generated

Table 1. 10-fold crossvalidation accuracies in % for different feature weightings. M0: tf frequencies; M1: binary data; M2: $\sqrt{tf}$, M3: $1 + \log(tf)$, M4: Eq. (1); M5: Eq. (2).

	M0	M1	M2	M3	M4	M5
kNN	48.9	50.2	51.0	51.4	49.5	49.5
SSV	39.5	40.6	31.0	39.5	39.5	42.3
MLP (300 neurons)	66.0	56.5	60.7	63.2	72.3	71.0
SVM (C opt)	59.3 (1.0)	60.4 (0.1)	60.9 (0.1)	60.5 (0.1)	59.8 (0.01)	60.0 (0.01)
10 Ref. vectors	71.6	–	71.4	71.3	70.7	70.1

using LVQ method from all training data (one prototype/class) gave 66.3±1.6% using the same method, showing importance of the *a priori* knowledge.

It is also worth noting that the whole calculations for linear programming with prototypes on a 3.6 GHz PC took about 1.5 hour, while SVM with Gaussian kernel (with optimized C=10 and dispersion=0.1) or MLP takes more than 10 times longer. Linear SVM takes twice as much time and is much less accurate (calculations were done using the GhostMiner package [10]).

4 Conclusions

Categorization of documents should be treated as the first step towards full annotation, facilitating subsequent disambiguation of terms and concepts . Medical texts are very specific, containing very large number of unique concepts. Standard approach to the document classification, based on vector representation using the tf×idf weighting scheme [3] leads to quite poor results using the nearest neighbor and decision trees approaches. Knowledge contained in medical records, such as the discharge summaries analyzed here, is by itself not sufficient to categorize them. Therefore reference texts have been introduced, systematically describing each disease documents can be classified to. New approach to the term weighting and evaluation of similarity of documents that refers to the background knowledge and that seems to capture human intuitions has been presented and tested on medical records.

Even the simplest implementation of a prototype-based classifier with linear programming for optimization of parameters reported here gave substantial improvement in accuracy. Background knowledge should obviously be stored in more than one prototype. Finding the simplest decomposition of medical records into classes using either sets of logical rules or minimum number of prototypes, is an interesting challenge. The approach presented here seems to be a step in right direction.

References

1. D. Campbell, S.B. Johnson, Comparing syntactic complexity in medical and non-medical corpora. Proc. of the AMIA Annual Symposium, 2001: 90-5.
2. J. Pestian, B. Aronow, K. Davis, *Design and Data Collection in the Discovery System.* Proc. Int. Conf. on Math. and Eng. Techniques in Medicine and Biological Sciences, CSREA Press, Providence, USA 2002.
3. C.D. Manning and H. Schütze, Foundations of Statistical Natural Language Processing MIT Press, Cambridge, MA 1999.
4. UMLS: http://www.nlm.nih.gov/research/umls
5. MetaMap: http://mmtx.nlm.nih.gov
6. J. Czyzyk, S. Mehrotra, M. Wagner and S. J. Wright: PCx: An Interior-Point Code for Linear Programming, Optim. Method. Softw. 12 (1999) 397-430.
7. MedNet: http://www.medicinenet.com
8. C.H. Boston: http://web1.tch.harvard.edu/cfapps/A2Z.cfm
9. Medline Plus: http://www.nlm.nih.gov/medlineplus/encyclopedia.html
10. GhostMiner: http://www.fqspl.com.pl/ghostminer/

A Neurofuzzy Methodology for the Diagnosis of Wireless-Capsule Endoscopic Images

Vassilis Kodogiannis[1], and H.S. Chowdrey[2]

[1] Mechatronics Group, School of Computer Science, University of Westminster,
London, HA1 3TP, UK
kodogiv@wmin.ac.uk
[2] Biomedical Sciences Dept, School of Biosciences, University of Westminster,
London, W1W 6UW, UK
h.s.chowdrey@wmin.ac.uk

Abstract. In this paper, a detection system to support medical diagnosis and detection of abnormal lesions by processing endoscopic images is presented. The endoscopic images possess rich information expressed by texture. Schemes have been developed to extract new texture features from the texture spectra in the chromatic and achromatic domains for a selected region of interest from each colour component histogram of images acquired by the new M2A Swallowable Capsule. The implementation of an advanced fuzzy inference neural network which combines fuzzy systems and clustering schemes and the concept of fusion of multiple classifiers dedicated to specific feature parameters have been also adopted in this paper. The preliminary test results support the feasibility of the proposed method.

1 Introduction

Medical diagnosis is based on information obtained from various sources, such as results of clinical examinations and histological findings, patients' history and other data that physician considers in order to reach a final diagnostic decision. Imaging techniques have been extensively used, in the last decades, as a valuable tool in the hands of an expert for a more accurate judgment of patients' condition. Recently a new wireless endoscopy system has been developed by Israeli-based Given Imaging Limited and produces high-quality images of the small bowel without pain or discomfort to the patient [1]. The system consists of a small swallowable capsule containing a battery, a camera on a chip, a light source, and a transmitter. The camera-capsule has a length of three centimetres so it can be swallowed with some effort. In 24 hours, the capsule is crossing the patient's alimentary canal. For the purpose of this research work, endoscopic images have been obtained using this innovative endoscopic device. They have spatial resolution of 171x151 pixels, a brightness resolution of 256 levels per colour plane (8bits), and consisted of three colour planes (red, green and blue) for a total of 24 bits per pixel. The proposed methodology in this paper is considered in two phases. The first implements the extraction of image features while in the second phase one neurofuzzy scheme is implemented / employed to perform the diagnostic task. In this research, a new approach of obtaining statistical

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 647 – 652, 2005.

features/parameters from the texture spectra is proposed both in the chromatic and achromatic domains of the image. The definition of texture spectrum employs the determination of the texture unit (TU) and texture unit number (N_{TU}) values. Texture units characterise the local texture information for a given pixel and its neighbourhood, and the statistics of the entire texture unit over the whole image reveal the global texture aspects.

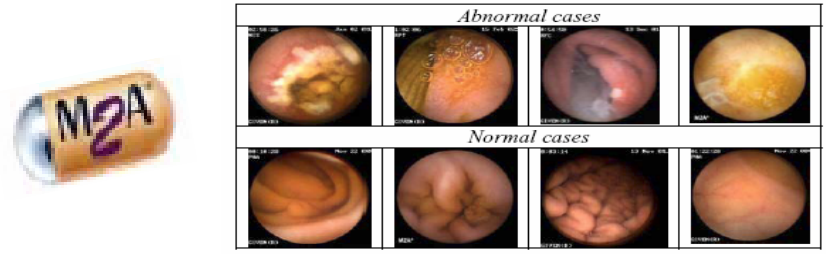


Fig. 1. Selected endoscopic images of normal and abnormal cases

For the diagnostic part, the concept of multiple-classifier scheme has been adopted, where the fusion of the individual outputs was realised using fuzzy integral. The neurofuzzy classifier-scheme adopted in this study utilises a two stage process. Initially a clustering algorithm is applied for the sample data in order to organise feature vectors into clusters such that points within a cluster are closer to each other than vectors belonging to different clusters. The fuzzy rule base then is created, using results obtained from this algorithm.

2 Image Features Extraction

Texture is broadly defined as the rate and direction of change of the chromatic properties of the image, and could be subjectively described as fine, coarse, smooth, random, rippled, and irregular, etc. For this reason, we focused our attention on nine statistical measures (standard deviation, variance, skew, kurtosis, entropy, energy, inverse difference moment, contrast, and covariance) [2]. All texture descriptors are estimated for all planes in both RGB {R (Red), G (Green), B (Blue)} and HSV {H (Hue), S (Saturation), V (Intensity)} spaces, creating a feature vector for each descriptor $D_i=(R_i,G_i,B_i,H_i,S_i,V_i)$. Thus, a total of 54 features (9 statistical measures x 6 image planes) are then estimated. For our experiments, we have used 70 endoscopic images related to abnormal cases and 70 images related to normal ones. Fig. 1 shows samples of selected images acquired using the M2A capsule of normal and abnormal cases. Generally, the statistical measures are estimated on histograms of the original image (1st order statistics) [3]. However, the histogram of the original image carries no information regarding relative position of the pixels in the texture. Obviously this can fail to distinguish between textures with similar distributions of grey levels. An alternative scheme is proposed in this study to extract new texture features from the texture spectra in the chromatic and achromatic domains, for a selected region of interest from each colour component histogram of the endoscopic images.

2.1 N_{TU} Transformation

The definition of texture spectrum employs the determination of the texture unit (TU) and texture unit number (N_{TU}) values. Texture units characterise the local texture information for a given pixel and its neighbourhood, and the statistics of all the texture units over the whole image reveal the global texture aspects. Given a neighbourhood of $\delta\times\delta$ pixels, which are denoted by a set containing $\delta\times\delta$ elements $P=\{P_0,P_1,....,P_{(\delta\times\delta)-1}\}$, where P_0 represents the chromatic or achromatic (i.e. intensity) value of the central pixel and P_i $\{i=1,2,...,(\delta\times\delta)-1\}$ is the chromatic or achromatic value of the neighbouring pixel i, the $TU=\{E_0,E_1,....,E_{(\delta\times\delta)-1}\}$, where $E_i\{i=1,2,...,(\delta\times\delta)-1\}$ is determined as follows:

$$E_i = \begin{cases} 0, & if \quad P_i < P_0 \\ 1, & if \quad P_i = P_0 \\ 2, & if \quad P_i > P_0 \end{cases} \tag{1}$$

The element E_i occupies the same position as the i^{th} pixel. Each element of the TU has one of three possible values; therefore the combination of all the eight elements results in 6561 possible TU's in total. The texture unit number (N_{TU}) is the label of the texture unit and is defined using the following equation:

$$N_{TU} = \sum_{i=1}^{(\delta\times\delta)-1} E_i \times \delta^{i-1} \tag{2}$$

Where, in our case, $\delta = 3$ .The texture spectrum histogram $(Hist(i))$ is obtained as the frequency distribution of all the texture units, with the abscissa showing the N_{TU} and the ordinate representing its occurrence frequency. The texture spectra of various image components {I (Intensity), R (Red), G (Green), B (Blue), H (Hue), S (Saturation)} are obtained from their texture unit numbers. The statistical features are then estimated on the histograms of the N_{TU} transformations of the chromatic and achromatic planes of the image (R,G,B,H,S,V).

3 Image Features Extraction

Recently, the concept of combining multiple classifiers has been actively exploited for developing highly reliable "diagnostic" systems [5]. One of the key issues of this approach is how to combine the results of the various systems to give the best estimate of the optimal result.

In this study, six subsystems have been developed, and each of them was associated with the six planes specified in the feature extraction process (*i.e.* R, G, B,

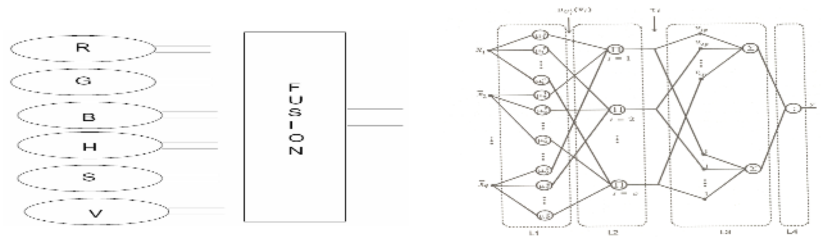


Fig. 2. Proposed fusion scheme and the neurofuzzy classifier

H, S, & V). For each subsystem, 9 statistical features have been associated with, resulting thus a total 54 features space. Each subsystem was modelled with the proposed neurofuzzy learning scheme. This provides a degree of certainty for each classification based on the statistics for each plane. The outputs of each of these networks must then be combined to produce a total output for the system as a whole as can be seen in Fig. 2. The aim in this study is to incorporate information from each plane/space so that decisions are based on the whole input space. The adopted in this paper methodology was to use the fuzzy integral (FI) concept which claims to resolve such issues by combining evidence of a classification with the systems expectation of the importance of that evidence [4]. The classification scheme utilised here is a neurofuzzy system that incorporates a two-stages clustering algorithm for finding the initial parameters of rules.

3.1 Clustering Algorithm

The clustering algorithm we apply in this paper consists of two stages. In the first stage the method similar to LVQ algorithm generates crisp *c*-partitions of the data set. The number of clusters *c* and the cluster centres $v_i,\ i = 1,...,c,$ obtained from this stage are used by FCM (Fuzzy *c*-means) algorithm in the second stage. The first stage clustering algorithm determines the number of clusters by dividing the learning data into these crisp clusters and calculates the cluster centres which are the initial values of the fuzzy cluster centres derived the second stage algorithm. Let $Z = [z_1,..., z_n] \in \mathrm{R}^{\mathrm{np}}$ be a learning data. The first cluster is created starting with the first data vector from **Z** and the initial value of the cluster centre is taking as a value of this data vector. Cluster centres v_i are modified for each cluster (i.e., $i = 1,...,c$) according to the following equation

$$v_i(t+1) = v_i(t) + a_t(z_k - v_i(t)) \qquad (3)$$

where $t = 0,1,2,...$ denotes the number of iterations, $a_t \in [0,1]$ is the learning rate and it is decreasing during performance of the algorithm (depending on the number of elements in the cluster). In the second stage the fuzzy c-means algorithm has been used. FCM is a constrained optimisation procedure which minimises the weighted within-groups sum of squared errors objective functions J_m with respect to fuzzy membership's u_{ik} cluster centres v_i, given training data z_k, $i=1,...,c$; $k=1,..,n$

$$\min_{(U,V)}\{J_m(U,V;Z)=\sum_{k=1}^{n}\sum_{i=1}^{c}(u_{ik})^m\left\|z_k-v_i\right\|^2\} \tag{4}$$

The number of clusters *c* and the initial values of cluster centres v_i come from the first stage clustering algorithm.

3.2 Fuzzy Inference Neural Networks

The two-stages clustering algorithm provides the fuzzy *c*-partition of the sample data. The number of rules in the proposed fuzzy inference neural network (FINN) equals to the number of clusters *c* obtained from the clustering algorithm. The proposed FINN scheme is a MIMO adaptive fuzzy logic system with centre average as defuzzification concept. The schematic of the FINN scheme which is shown in Fig. 2 consists of four layers. The first two layers LI and L2 correspond to IF part of fuzzy rules whereas layers L3 and L4 contain information about THEN part of these rules, and perform the defuzzification task. There are $c\times q$ elements in layer Ll. They realise the membership functions which are defined by

$$\mu_j^i=\exp\left[-\left(\frac{x_j-v_{ij}}{\sigma_{ij}}\right)^2\right] \tag{5}$$

for $j=1,...,q$ and $i=1,...,c$. The values v_{ij} in Eq. (5) denote the centres of the Gaussian membership functions and are equal to the values of the vectors v_i which have been derived from the second stage clustering algorithm. The second layer L2 has *c* elements which realise multiplication operation because of using Larsen rule in fuzzy reasoning procedure. Each element in this layer is associated with one fuzzy rule. Outputs of this layer represent the fire strength of the rules, expressed by

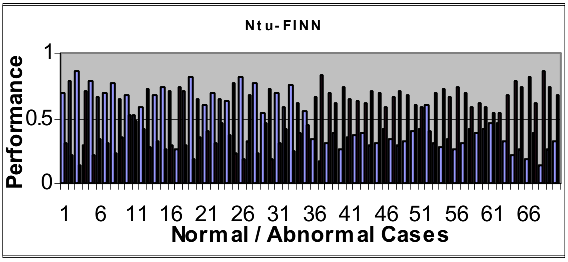
$$\tau_i=\prod_{j=1}^{q}\mu_j^i(\overline{x}_j) \tag{6}$$

The multi-layer connectionist structures of FINN scheme allow us to apply, learning procedures similar to the back-propagation method which is commonly used as learning algorithm for feed-forward multi-layer artificial neural networks.

4 Results

The $\mathrm{N_{TU}}$ transformation of the original histogram has produced a satisfactory diagnostic performance of the multi-classifier scheme. The soft combination of FINN-based classifiers using the FI fusion concept resulted in 94.28% accuracy over the testing dataset (4 mistakes out of 70 testing patterns), demonstrating in this way the efficiency of this scheme in terms of accuracy. More specifically, 3 normal cases as abnormal and one abnormal as normal one provide us a good indication of a "healthy" diagnostic performance. However the level of confidence/certainty was 0.52 as shown in Table 1.

Table 1. N_{TU}-based Performance

Modules	**FINN Accuracy (70 testing patterns)**	
R	94.28% (4 mistakes)	
G	97.14% (2 mistakes)	
B	92.85% (5 mistakes)	
H	94.28% (4 mistakes)	
S	95.71% (3 mistakes)	
V	91.42% (6 mistakes)	
Overall	94.28% (4 mistakes)	

5 Conclusions

An approach on extracting statistical features from endoscopic images using the M2A Given Imaging capsule have been developed by obtaining those quantitative parameters from the texture spectra from the calculation the texture unit numbers (N_{TU}) over the histogram spectrum. In this study, an intelligent decision support system has been developed for endoscopic diagnosis based on a multiple-classifier scheme.

References

1. Idden, G., Meran, G., Glukhovsky A., Swain P.: Wireless capsule endoscopy, Nature, (2000) 405-417.
2. Haralick, R.M.: Statistical and structural approaches to texture, IEEE Proc., **67** (1979) 786- 804.
3. Boulougoura, M., Wadge, E., Kodogiannis, V., Chowdrey, H.S.: Intelligent systems for computer-assisted clinical endoscopic image analysis, 2nd IASTED Int. Conf. on Biomedical Engineering, Innsbruck, Austria, (2004) 405-408.
4. Kuncheva, L.I.: Fuzzy Classifier Design, Physica-Verlag, (2000).
5. Wadge, E., Kodogiannis, V.S.,: Intelligent diagnosis of UTI in vivo using gas sensor arrays, Int. Conf. on Neural Networks and Expert Systems in Medicine and HealthCare, NNESMED 2003, (2003) 93-98.

Neural Network Use for the Identification of Factors Related to Common Mental Disorders

T.B. Ludermir[2], C.R.S. Lopes[1,2], A.B. Ludermir[2], and M.C.P. de Souto[2]

[1] State University of Bahia (UESB) – 45.216-510 – Jequié – BA – Brasil
[2] Federal University of Pernambuco C.P 7851 – 50.732-970 – Recife – PE - Brazil
tbl@cin.ufpe.br

Abstract. This paper shows that MLP trained with the optimizing *Simulated Annealing* algorithm, may be used for identification of the factors related to Common Mental Disorders (CMDs). The average percentage of correct classification of individuals with positive diagnostic for the CMDs was of 90.6% in the experiments related in the paper.

1 Introduction

CMDs, and among them the anxiety and depression have been pointed out as the common causes of morbidity in developed countries as much as in the developing ones, as the example of Brazil. These mental disorders represent a high social and economic charge because they are disabled, they constitute important cause of lost of workdays and they take a substantial use of health care services [3].

The use of techniques that may lead to an identification of the factors that present the larger possibility of being related to these CMDs it is relevant to assist within the decision taking process around the planning and intervention of public health care.

Artificial Neural Networks (ANNs) have been largely used in the health care field and they are known because they generally obtain a good precision result [2,4,6].

With this research we intend, mainly, to experimentally show that a MLP trained with Simulated Annealing (SA) algorithm is able to identify the factors related to the CMDs. The results obtained with MLP were compared with the ones presented by Ludermir [3]. She applied the logistics regression method, using the same data basis to analyze the independence of each variable association with the CMDs. On the statistic analysis for the identification of the factors related to the CMDs, it was estimated the simple and adjusted *odds-ratios,* whose statistic significance was evaluated by the χ^2 test, considering the 95% confidence interval and values of p (≤ 0.05).

2 Simulated Annealing

The SA algorithm [1] consists of a sequence of iterations. Each iteration consists of randomly changing the current solution to create a new solution in the neighborhood of the current solution. The neighborhood is defined by the choice of the generation mechanism. Once a new solution is created, the corresponding change in the cost function is computed to decide whether the new solution can be accepted as the cur-

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 653 – 658, 2005.

rent solution. If the change in the cost function is negative, the new solution is directly taken as the current solution. Otherwise, it is accepted according to Metropolis´s criterion [1,7]: if the difference between the cost function values of the current and new solutions is equal to or larger than zero, a random number in [0,1] is generated from an uniform distribution. If the random number is equal to or less than $exp(-\Delta E / T)$, where ΔE is the change in the cost function and T is the current temperature, then the new solution is accepted as the current solution. If not, the current solution is unchanged [1,7].

Given a set S of solutions and a real-valued cost function $f : S \to R$, the algorithm searches for the global solution *s*, such that $f(s) \leq f(s'), \forall s' \in S$. The search stops after I epochs, and a cooling schedule updates the temperature T_i of epoch *i*. The structure of the algorithm is shown in Figure 1.

```
$s_0 \leftarrow$ initial solution in $S$
For $i = 0$ to $I - 1$
Generate neighbor solution $s'$
If      $f(s') \leq f(s_i)$
        $s_{i+1} \leftarrow s'$
else
 $s_{i+1} \leftarrow s'$ with probability $e^{-\left[f(s')-f(s_i))/T_{i+1}\right]}$
otherwise $s_{i+1} \leftarrow s_i$
Return $s_i$
```

Fig. 1. Basic structure of the Simulated Annealing algorithm

3 Data Basis Description

Data collection was community-based through interviews and assessment of mental health status from a research made by Ludermir [3]. Ludermir determined the prevalence of the CMDs in the area studied, and analyzed the association with living and work conditions. The study was developed with 621 adults of an aleatory domicile sample and using a statistic model of logistic regression for the analysis of the data.

The data set has the following variables: literacy, education, house ownership, housing conditions and possession of household appliances (living conditions), insertion in the productive process and household per capita monthly income (work conditions). The total prevalence of the CMDs in the data set was 35%, 216 cases.

The following assumptions were made with ANN experiments: 1) A balance of positive/negative diagnose cases. It was randomly selected 216 patterns of negative diagnose for the composition of the basis. 2) The data was divided in 50% patterns for the training set; 25% for the validation set and 25% for the tests set [5]. 3) All variables in the date set are ordinal/categorical and they were codified with discrete numbers between 0 and 1. 4) The network output was defined with two nodes, 10 to represent the positive and 01 for the negative diagnoses. 5) Two folds, fold 1 and fold 2, were randomly created from the original data set to perform the experiments with ANNs.

4 Methodology

In this work, two different architectures were taken as initial topologies. The architectures have one-hidden-layer MLP networks with 4 hidden nodes, having all possible feedforward connections between adjacent layers, and two output nodes. The difference in the architecture is in the number of input nodes. For the initial experiments there are 11 input nodes and for experiments with main variables there are 7 input nodes. For each initial topology, 10 distinct random weight initializations were used, and the initial weights were taken from a uniform distribution between –1.0 and +1.0. For each weight initialization, 30 runs of SA were performed.

The MLP was trained with the SA algorithm for the simultaneous optimizing of the architecture and the weights of the network (nodes and connections). The identification of the CMD factors was possible by the optimizing the architecture of the network. It was observed the variables that were mostly used for the results obtaining on every execution of the algorithm, and with that, to identify those, which presented greater possibility of being related with the studied problem. This technique was adapted from Yamazaki [7]. The obtained results were compared with those presented by Ludermir [3] applying the statistic model of logistic regression.

The experiments were executed in two distinct stages: 1) in the beginning the data set was composed with all the data basis variables, in a total of 11; 2) from the obtained results in first experiments, it was performed new experiments with the number of resultants input variables, in a total of 7. These 7 variables were chosen based on the number of times the algorithm had chosen such variables. That is the most used variables (in terms of percentage) were chosen.

In order to implement this algorithm for a problem, the following aspects must be defined: (1) the representation of solutions; (2) the cost function; (3) the generation mechanism for the new solutions; and (4) the cooling schedule and stopping criteria. (1) Each MLP is specified by an array of connections, and each connection is specified by two parameters: (a) the connectivity bit, which is equal to one if the connection exists, and zero otherwise; and (b) the connection weight, which is a real number. If the connectivity bit is equal to zero, its associated weight is not considered, for the connection does not exist in the network. (2) The cost of each solution is the mean of two important parameters: (a) the classification error for the training set (percentage of incorrectly classified training patterns); and (b) the percentage of connections used by the network. Therefore, the algorithms try to minimize both network performance and complexity. (3) The generation mechanism acts as follows: first the connectivity bits for the current solution are changed according to a given probability, which in this work is set at 20%. This operation deletes some network connections and creates new ones. Then, a random number taken from a uniform distribution in [-1.0, +1.0] is added to each connection weight. These two steps can change both topology and connection weights to produce a new neighbor solution. (4) The cooling strategy chosen is the geometric cooling rule. The initial temperature is set to 1, and the temperature factor is set to 0.9. The temperature is decreased at each 10 iterations, and the maximum number of iterations allowed is 5000. The algorithm stops if: (a) the GL5 criterion [5] is met (based on the classification error for the validation set) after 300 iterations; or (b) the maximum number of 1000 iterations is achieved.

For each experiment the classification error in the test set, the percentage of correct classification of individuals with positive and negative diagnosis for the CMDs and the average of variables used by the SA algorithm in the obtaining of results were analyzed. All results where the classification error is greater than 28.13% (fold 1) and is greater than 29.75% (fold 2) was excluded of the analysis. For the definition of the variable amount that presented most relation with the CMDs, it was observed the average of variables used for the diagnostic classification (input nodes) in the analyzed executions of the SA algorithm. More details may be seen in Lopes [3].

5 Results

Six different experiments are showed in this paper. The experiments were done in two stages: 1) 11 input variables; 2) 7 input variables. The analysis of the results was developed in two distinct ways: 1) observing individually each used fold; 2) observing the obtained average between the folds.

Table 1 presents the experimental results in the following way: EXP1 11 inputs and fold 1, EXP2 11 inputs and fold 2, EXP3 11 inputs and average of the two folds, EXP 4 7 inputs and fold 1, EXP 5 7 inputs and fold 2, and EXP6 7 inputs and average of the two folds. The table contains the use percentage of each variable, an average classification error, the percentages of the correct classification of the cases with positive/negative diagnose for the CMDs. The bold face values are for the input variables which were most used in the experiment. The Students' t-test with a significance level of 5% [8] was used to perform the statistical analysis in the results.

With experiments done with 11 inputs we may observe in Table 1 that, in the experiments performed with both folds, among the used variables to measure the living conditions the one that mostly stood out was the education, and as for the work conditions it was the insertion in the productive process variable.

In the experiments with only the most used input variables, the average number of used variables was equal in both folds, four. Comparing the results of 7 inputs with 11 inputs, in relation to the classification error, in the fold 1 there was a reduction in 1.71% and in the fold 2 in 0.85%. The correct classification percentage in the cases with positive diagnose was increased in both fold, in 1.51% (fold 1) and 5.28% (fold 2). As for the correct classification percentage of the cases with negative diagnose for the CMDs presented, however, there was an increase in 1.92% in the fold 1 and a decrease in 3.58% fold 2. Analyzing the results with the average of both folds, the classification error was decreased in 1.28%, the correct classification percentage of the positive cases improved in 3.40% and the correct classification percentage of the cases with negative diagnose was decreased in 0.83%.

We may observe that, the variables that mostly stood out analyzing individually and simultaneously both folds with experiments with 7 input variables were: education, insertion in the productive process, gender and income. Among the used variables to measure the living conditions it stood out the education and as for the work conditions, the insertion in the productive process variable. The huge reduction in the use percentage of some variables between the two different architectures it is because of the reduction of the number of variables used.

In general, in the performed experiments with only the 7 variables mostly used by the SA algorithm, the network classification error as well as the correct classification percentage of the cases with positive diagnose for the CMDs were improved. Those results suggest that the exclusion of the variables that did not present relationship with CMDs in the data basis used, it contributed to the improvement of the results obtained in the experiments. Therefore, the process of variable and feature selection improved the performance of the system, provided faster and more cost-effective systems and provided a better understanding of the underlying process that generated the data.

Table 1. Experimental Results

	EXP1	EXP2	EXP3	EXP4	EXP5	EXP6
Age	**62.07**	57.45	**59.76**	52.00		26.00
Literacy	58.62	57.45	58.04			
Migration	51.72	59.57	55.65			
Education	**89.65**	**74.47**	**82.06**	72.00	60.00	66.00
House ownership	**65.52**	**61.70**	**63.61**	20.00	36.67	28.34
Insertion Productive Proc.	**65.52**	**87.23**	**76.38**	72.00	86.67	79.34
Housing conditions	51.72	**61.70**	56.71		20.00	10.00
Gender	**65.52**	**74.47**	**70.00**	60.00	73.33	66.67
Status marital	**79.31**	53.19	**66.25**	40.00		20.00
Income	**62.07**	**74.47**	**68.27**	72.00	70.00	71.00
Possession household app.	48.27	**68.08**	58.18		53.33	26.67
Classification Error	23.08	24.21	23.64	21.37	23.36	22.36
Positive Diagnose	89.08	88.73	88.90	90.59	94.01	92.30
Negative Diagnose	64.75	62.84	63.79	66.66	59.25	62.96

6 Final Considerations

Even though the logistic model is the methodology normally used when the purpose is to identify the factors of risk that have association with the variable answer, where the coefficients of regression may be interpreted by the *odds-ratios* [3], it was possible to observe good results in the experiments with MLP with relation to the prediction of the positive cases for the studied problem. The obtained average in our experiments around the correct classification of the individuals with positive diagnose for the CMDs was of 90.6%.

Our results are similar to those obtained with the statistic technique of logistic regression applied by Ludermir [3], when it is compared with the analysis made with simple *odds-ratios,* where the variables education, income, gender and insertion in the productive process were statistically significant, with the values of $p \leq 0.0001$. After the adjustment of the odds-ratios in the results obtained by Ludermir [3], education and income presented relationship with CMDs. It is important to remind that those variables, in all the experiments with ANNs, they were present in the obtained results, standing out among the variables that presented larger percentage of use for the algorithm, in other words, larger possibility of they being related with CMDs.

With the optimizing network architecture it was possible to establish which variables presented greater probability of being related with the studied problem. That is, the applied methodology in the experiment is presented as an interesting alternative for problems application when the purpose is the identification of factors related to the variable response (network output).

From the results presented here, we may conclude that a trained MLP network with the SA algorithm, with simultaneous optimizing of its architecture and weights, may be an interesting alternative to the statistic model of logistic regression, to the analysis of the factors related with the CMDs, because the network is able to detect all the possible interactions among the many explaining variables.

As future work possibility we pointed out: 1) to measure the performance of new experiments, applying the same methodology, with the use of others data set; 2) the creation of others data bases folds, because, the use of more folds will lead to much more trustable results, once that, the order of the examples and the chosen examples may influence in the obtained results in the ANN; 3) the use of other optimizing techniques, as for example, *Tabu Search*.

Acknowledgment

The authors are thankful to Brazilian research councils CAPES, CNPq and Finep for financial support.

References

1. Kirkpatrick S., Gellat Jr., C.D and Vecchi, M.P (1983). “Optimization by SA”, Science, 220: 671-680.
2. Lopes, C. R. S. (2003). ANN for the Identification of Factors Related to Common Mental Disorders (In Portuguese). Master Thesis. Federal University of Pernambuco.
3. Ludermir, A. B. (1998). “Socioeconomic status employment, migration and common mental disorders in Olinda, Northeast Brazil”. PhD Thesis, London School of Hygiene and Tropical Medicine.
4. Ottenbacher, K. J., et al (2001). “Comparison of logistic regression and neural networks to predict rehospitalization in patients with stroke”. Journal of Clinical Epidemiology, Vol. 54 (11), pp. 1159-1165.
5. Prechelt, L. (1994). Proben 1 – A set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, Fakultat fur Informatik, Universitat Karlsruhe.
6. Tu, J. V. (1996). Advantages and disadvantages of using Artificial Neural Networks versus Logistic Regression for Predicting Medical Outcomes. Journal Clin. Epidemiol, Vol. 49 (11), pp. 1225-1231.
7. A. Yamazaki and T.B. Ludermir (2003). Neural Network Training with Global Optimization Techniques, International Journal of Neural Systems, Vol. 13 (2), pp. 77-86.
8. Johson R. A., Wichern D. W.: Applied Multivariate Statistical Analysis. Prentice Hall, (1999).

Development and Realization of the Artificial Neural Network for Diagnostics of Stroke Type

O.Yu. Rebrova and O.A. Ishanov

Institute of Neurology, Moscow

Abstract. Methods of artificial neural networks are applied to the development of the decision support system for differential diagnostics of three types of stroke. Diagnostic sensitivity and positive predictive value were used as the basic criteria for estimation of efficiency of the developed algorithm. Their values appeared to be 97% and 99% respectively, and these results significantly exceed both the existing level of physicians' diagnostics, and the efficiencies of statistical algorithms developed earlier. C-code was generated, and web-based application was realized. Research of algorithm's efficiency continues.

Keywords: artificial neural network, medical diagnostics, web-based application, perceptron, stroke.

1 Introduction

Cerebral stroke is one of the central problems of clinical medicine nowadays since it takes the third place among the reasons of death rate from noninfectious pathology. 80% of patients who survived after stroke are handicapped [1]. Three types of stroke (ischemic, haemorrhagic, subarachnoid hemorrhage) are marked out and tactics of treatment in these three cases should be different. Efficiency of medical actions depends on correctness of diagnostics of stroke type, and, hence, the prognosis for each patient too. In practice the rate of physicians' errors in stroke type diagnostics even by experienced doctors is about 20-45% [2]. Computer tomography is not available for about 80% of stroke patients in Russia. That is why free intellectual computer system to support decision-making by emergency department personnel (neurologist or general practitioner) could promote decrease of medical errors.

The algorithms of differential diagnostics of stroke types developed earlier were not effective because of low accuracy (71% on the average) and of some essential restrictions. So we have set the task to develop and realize effective system to support medical decisions.

There are two basic ways of computer intellectual systems development: design of mathematical algorithms using data mining and representation of experts' knowledge. Development of formal rules for diagnostics and classification can be carried out using statistical and logical methods (logit regression, discriminant analysis, etc.) and cybernetic methods of optimization (neural networks (NN), etc.). Earlier we solved our problem by statistical methods (logit-regression models), also we have developed expert system on the basis of network hierarchical threshold model. Our

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 659 – 663, 2005.

results were successful enough and significantly exceeding an existing level of routine medical diagnostics, however they have not allowed to reach the high values of efficiency, so we decided to try NN approach. This approach was already successfully used to solve diagnostic problems in neurology [3-9].

2 Materials and Methods

Data on 298 patients with acute (1-7 days) stroke were analyzed: 211 cases of ischemic stroke (IS), 73 cases of haemorrhagic stroke (HS), 14 cases of subarachnoid haemorrhage (SAH). The relative frequencies of stroke types correspond to their population prevalences. The true type of stroke (the final diagnosis) was verified by the results of clinical, instrumental and laboratory investigation, including brain computed tomography.

Initial sample has been divided to training (n=268) and control (n=30) samples using computer randomization. To test algorithms additional sample of 25 cases (16 cases of IS, 8 cases of HS, 1 case of SAH) was collected.

Each case initially was described by 239 variables, including 8 quantitative variables, the rest were qualitative or binary ones. Qualitative nominal variables have been preliminary transformed to binary variables using the rule "1-of-N". The variables describe patient's clinical features (both somatic and neurological status), disease and life history.

The statistical analysis of data was carried out using software package «Statistica 5.5» (StatSoft, Inc., USA).

To develop artificial NN «Statistica Neural Networks 4.0» and «Statistica 6.0» (StatSoft, Inc., USA) software were used.

According to the concept of evidence-based medicine sensitivity (Se), specificity (Sp), diagnostic accuracy, positive (PPV) and negative (NPV) predictive values, and also agreement index of independent diagnostic conclusions K (Kappa) served as the criteria to estimate the efficiency of the developed diagnostic algorithms. Point and interval (95% confidence intervals) estimations have been computed for the listed parameters.

3 Results

Primary to construction of mathematical algorithms for stroke types diagnostics we estimated efficiency of routine physicians' diagnostics. With this purpose we cross-tabulated preliminary (formulated at admission) and final (formulated at discharge) diagnoses. Two approaches were applied. At the first we made calculations only for those cases (n=119), in which the stroke type has been specified in preliminary diagnosis. Average Se thus has appeared to be 86% [79%; 91%], accuracy of diagnostics - 84% [76%; 90%]. At the second approach we believed that in 155 cases the stroke type has not been specified in preliminary diagnosis owing to uncertainty of doctors, and considered these cases as cases of refusal to specify stroke type. Then average Se medical diagnostics has appeared to be 53% [47%; 59%], accuracy of diagnostics - 36% [31%; 42%]. These results we further used for comparison with developed NN algorithms.

It was necessary to decrease the dimension of attribute space. We used following approaches for this purpose:

1. The expert estimation of attributes importance;
2. The statistical analysis of data - methods of the analysis of tables (the Chi-square test, Fisher exact test) and comparisons of groups (Mann-Whitney U-test, Kruskal-Wallis ANOVA);
3. Genetic algorithm.

Combining results of these three approaches, we have step by step selected 34 perspective attributes from 239 initially available attributes.

To solve classification task we had to search best architecture and parameters of a network, so we used "Intelligent Problem Solver" module. For training NN we used training sample, for controlling – the control one, for testing – the test one.

NN of four architectures (topologies) were developed: linear NN, probability NN, radial basic functions, multi-layer (three- and four-layer) perceptrons.

The best network appeared to be the four-layer perceptron with two hidden layers of neurons (29-12-14-3 neurons in corresponding layers). All input attributes are qualitative, and 24 of them are binary. Six inputs describe anamnesis vitae, 9 – anamnesis morbi, 1 – somatic status and 13 – neurological deficit. The output layer consists of three neurons according to number of differentiated stroke types. The decision in favor of one of diagnoses is based on the most probable condition.

Classification matrices were considered as the basic result of NN functioning. The results of classification are presented in Table 1, and parameters of efficiency - in Table 2. Average Se appeared to be 97% [94%; 98%], PPV - 99% [97%; 100%] on training sample. Diagnostic accuracy of algorithm is 98% [96%; 99%]. The agreement of the NN conclusions and final diagnoses is 0,949 [0,908; 0,989].

Table 1. Results of classification by four-layer perceptron (absolute frequencies)

The conclusions of NN	Final diagnoses					
	Training sample			Control sample		
	IS (n=187)	HS (n=69)	SAH (n=12)	IS (n=24)	HS (n=4)	SAH (n=2)
IS (n=216)	187	6	0	23	0	0
HS (n=68)	0	63	0	1	4	0
SAH (n=14)	0	0	12	0	0	2

The Se of logit-regression model developed by us earlier was 80% [74%; 85%], the Se of routine physicians' diagnostics was estimated as 86% [79%; 91%]. So NN algorithm produces significantly better results.

Table 2. Parameters of efficiency of four-layer perceptron

Parameter	Training sample			Control sample		
	IS	HS	SAH	IS	HS	SAH
Se	100%	91%	100%	96%	100%	100%
Sp	93%	100%	100%	100%	96%	100%
PPV	97%	100%	100%	100%	80%	100%
NPV	100%	97%	100%	86%	100%	100%

We also have carried out additional testing of the developed algorithm on 25 cases (16 cases of IS, 8 cases of HS, 1 case of SAH) which have not been included earlier neither in training, nor in control samples. Results of testing have appeared to be the following: 16 of 16 cases of IS and 6 of 8 cases of HS were classified correctly, 1 case of SAH was recognized correctly too. One case of HS was incorrectly carried to IS, the second case - to SAH. Last error can be considered as rather non-gross as the pathophysiological mechanism (haemorrhage) has been defined correctly and hence the specific treatment cannot bring harm to the patient.

Further we have set the task to develop web-based application realizing developed NN algorithm. We used the C-code generator of «Statistica 6.0» software. Then we transformed C-code to Perl-code, and using CGI technology we have organized the functioning of the NN at the Internet-site of the Russian National Stroke Research Center (http://www.stroke-center.ru). The web-form for a choice of necessary values of attributes is given to the user. The codes (values) of attributes entered by the user are sent to the server and are processed by a CGI-script, the results are sent back to the user. Entered data including the values of variables entered by the user, the IP-addresses, e-mail addresses, results of data processing are added to database file and can be visualized through web-interface or kept in a XML-format on a local disk for the further processing by data analysis programs. The further analysis of collected data will allow to repeatedly estimate the NN efficiency.

4 Conclusion

Artificial NN for differential diagnostics of three types of stroke was developed. Its efficiency significantly exceeds the efficiency of routine medical diagnostics, and earlier developed logit-regression algorithm. Web-realization of artificial NN can be free used by doctors as a support system for decision-making. Research of efficiency of algorithm continues.

Acknowledgement

The authors wish to acknowledge Marina Yu. Maximova, MD, DSci, for medical advising.

References

1. Vereshchagin, N.V., Varakin, Yu.Ya.: The Epidemiology of Stroke in Russia. J. Neurology Psychiatry. Stroke Suppl. **1** (2001) 34-40 (in Russian).
2. Vereshchagin, N., Piradov, M., Suslina, Z. (eds.): Guidelines for stroke care – 2002. Intermedica, Moscow (2002) (in Russian).
3. Edwards, D.F., Hollingsworth, H., Zazulia, A.R., Diringer, M.N.: Artificial neural networks improve the prediction of mortality in intracerebral hemorrhage. Neurology. **53** (1999) 351-7.
4. Kemeny, V., Droste, D.W., Hermes, S., Nabavy, D.G., Schulte-Altedorneburg, G., Siebler, M., Ringelstein, E.B.: Automatic embolus detection by neural networks. Stroke. **30** (1999) 807-10.
5. Li, Y.C., Liu, L., Chiu, W.T., Jian, W.S.: Neural network modeling for surgical decisions on traumatic brain injury patients. Int. J. Med. Informatics. **57** (2000) 1-9.
6. Swiercz, M., Mariak, Z., Krejza, J., Lewko, J., Szydlik, P.: Intracranial pressure processing with artificial neural networks: prediction of ICP trends. Acta Neurochir. (Wien). **142** (2000) 401-6.
7. Hsu, M.H., Li, Y.C.: Predicting cranial computed tomography results of head injury patients using an artificial neural network. AMIA Annu. Symp. Proc. (2003) 868.
8. Di Bona, S., Niemann, H., Pieri, G., Salvetti, O.: Brain volumes characterization using hierarchical neural networks. Artif. Intell. Med. **28** (2003) 307-22.
9. de Tomasso, M., De Carlo, F., Difruscolo, O., Massafra, R., Sciruicchio, V., Belotti, R.: Detection of subclinical brain electrical activity changes in Huntington's disease using artificial neural networks. Clin. Neurophysiol. **114** (2003) 1237-45.

A First Attempt at Constructing Genetic Programming Expressions for EEG Classification

César Estébanez, José M. Valls, Ricardo Aler, and Inés M. Galván

Universidad Carlos III de Madrid,
Avda. de la Universidad, 30, 28911, Leganés (Madrid), Spain
{cesteban, jvalls, aler, igalvan}@inf.uc3m.es

Abstract. In BCI (Brain Computer Interface) research, the classification of EEG signals is a domain where raw data has to undergo some preprocessing, so that the right attributes for classification are obtained. Several transformational techniques have been used for this purpose: Principal Component Analysis, the Adaptive Autoregressive Model, FFT or Wavelet Transforms, etc. However, it would be useful to automatically build significant attributes appropriate for each particular problem. In this paper, we use Genetic Programming to evolve projections that translate EEG data into a new vectorial space (coordinates of this space being the new attributes), where projected data can be more easily classified. Although our method is applied here in a straightforward way to check for feasibility, it has achieved reasonable classification results that are comparable to those obtained by other state of the art algorithms. In the future, we expect that by choosing carefully primitive functions, Genetic Programming will be able to give original results that cannot be matched by other machine learning classification algorithms.

1 Introduction

Within the Machine Learning field, there are many domains where the main difficulty is not to determine the proper algorithm to be applied, or even selecting the most relevant attributes. In these cases, the attributes available in raw data are not the most significant for classification, and new attributes have to be constructed. This is usually called feature induction or constructive induction [1, 2, 3]. The brain computer interface (BCI) is a domain where raw data has to undergo some preprocessing, so that the right attributes for classification are obtained. In BCI research, several transformational techniques have been used to obtain high classification accuracy (over 90%): Principal Component Analysis [4], the Adaptive Autoregressive Model [5], FFT or Wavelet Transforms [6], etc. Intuitions, empirical results, and knowledge about the domain is what lead researchers towards using these methods. However, it would be useful to automatically build significant attributes appropriate for every kind of signal and classification task.

Genetic Programming (GP) is an evolutionary technique for evolving symbolic programs [7]. Most research has focused in evolving functional expressions,

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 665–670, 2005.

but using loops and recursion have also been considered [8]. Evolving circuits are also among the successes of GP [9]. In this paper, we intend to use GP to evolve expressions that project original data, which is expressed in coordinates of a space with N dimensions, to a new space with M dimensions (where $M << N$). Here, we consider two-class classification problems. We expect that by finding the right projection, it will be possible to separate data linearly (approximately) in the projected space.. The secondary goal is to project data into a smaller space: This way, the dimensionality of the problem is reduced and a new set of attributes is obtained. This new emergent-attributes represent the relevant information needed for classification. They also can reveal non-observable relationships between the original attributes, improving the understanding of the problem. Fitness of each projection is determined by computing the degree of linear separation of data in the projected space. This has been implemented as a linear perceptron. Our motivation for using GP is that the set of primitives GP uses for building hypothesis is a parameter of the algorithm. Other machine learning algorithms work with pre-defined primitives like node-comparisons in ID3 or neurons in ANN. GP allows to include the most relevant primitives for the domain, although this selection requires sometimes a long trial-and-error process.

The final aim of our research is to obtain results that cannot be obtained by directly applying other machine learning methods. This paper is a first shot at using GP for this purpose. Therefore, results obtained in this paper correspond to the most straightforward and simple approach, that will test the feasibility of GP, and can be used as a base to compare (and improve) future research.

This paper is organized as follows. Section 2 introduces Genetic Programming. Section 3 describes our approach. Section 4 reports the experiments carried out. And finally, Section 5 draws the main results of the paper and prepares for future research.

2 Genetic Programming

Genetic Programming (GP) is an evolutionary technique designed to generate programs automatically [7]. It has three main elements:

- A population of individuals. In this case, the individuals are computer programs.
- A fitness function. It is used to measure the goodness of the computer program represented by the individual.
- A set of genetic operators. In GP, the basic operators are reproduction, mutation, and crossover.

The GP algorithm enters into a cycle of fitness evaluation and genetic operator application, producing consecutive generations of populations of computer programs, until a good enough individual is found. Every genetic operator has a probability of being applied each time we need to generate an offspring individual for the next generation. Also, GP has many other parameters, the most important ones being the size of the population (M) and the maximum number of generations (G).

3 The Approach

We will learn from a set E of n examples expressed in a space U of N dimensions. Our objective is to be able to represent the examples in the space V, of M dimensions, and in which the examples will be linearly separable.

As we already have seen, Our method have two different applications: on one hand, the improvement of classification tasks by means of a transformation of the dataset; on the other hand, the reduction of dimensionality by constructing new attributes that are as good, at least, as the original ones.

Our method uses standard GP to evolve individuals made of M subtrees (as many of dimensions of the projected space V). Then, data is projected from U to V by means of applying the individual to the original data. Fitness is computed by measuring the degree of linear separation after the projection. The system stops if a 100% linear separation has been achieved or if the maximum number of generations is reached. Otherwise, the system outputs the individual that separated better the training data.

For the implementation of our application, we have used Lilgp 1.1, the software package for Genetic Programming developed in Michigan State University by Douglas Zongker and Bill Punch, members of the group GARAGe (Genetic Algorithms Research and Applications Group) (http://garage.cse.msu.edu/).

Terminal and Function Set
In our problem, terminal set will be formed by the attributes of the problem expressed in coordinates of U (u_0,u_1...,u_N), and by Ephemeral Random Constants [7].

The set of functions to use is difficult to determine: it must be sufficient for, along with the set of terminals, being able to express the solution to the problem, but they must not be too many as for uselessly increase the search space. Of course, for different domains, different terminal and function sets will be more appropriate. In this case, we have tried with just arithmetic functions $(+, -, *, /)$.

In the future, we would like to explore better grounded, domain-oriented sets of functions, like FFT or wavelet transforms.

GP Individuals
Instead of having individuals work with vectorial data and return a vector of M dimensions, every individual will contain M subtrees, using the same set of functions and terminals, that will be ran independently. Thus, a projection consists of a series of trees labelled (v_0,v_1...,v_M) that represent combinations of all the terminals (u_0,u_1...,u_N) and functions.

Genetic Operators
In our approach, we use the three genetic operators typically used in GP. Reproduction chooses an individual and copies it verbatim into the new population;

Mutation chooses an individual and a subtree, then the subtree is deleted and replaced with a randomly generated subtree; and Crossover selects two individuals, a subtree is selected on each individual, and the subtrees are swapped.

The Fitness Function

We already have introduced the basic mechanism of the fitness function. It takes the examples expressed in space U, project them using the GP individual, and obtain the examples expressed in space V of M dimensions. Next, a classification algorithm is applied to the projected data. In this case, we have choosen to apply a Simple linear Perceptron. The Perceptron is run for 500 cycles (experimentally we have checked that this is more than enough). If the SP converges, the projection would be producing a linear separation of the data and it would be the solution to the problem. If the SP does not converge, the fitness assigned to the individual is the number of examples that the SP has been able to correctly classify in the best cycle. We choose the punctuation of the best cycle because if projected data is not linearly separable, the SP will oscillate. Storing the best value guarantees stability of the fitness value. This way, fitness is gradual enough and has the resolution necessary to be able to exert a real selective pressure.

If the method does not find a linear (or nearly linear) classification of the examples in the space V, we can change the number of dimensions desired for V and launch the method again. As we already have stated, our main goal is to find a linear separation of examples and, in order to achieve this, it is possible for our application to increase M to values greater than N. However, in BCI research, the original number of dimensions N is usually very large, so it does not make sense to increase them in the new space V. This is the reason for not documenting in this paper the possibility of a M value greater than N.

4 Experiments

This section describes our first experiments using the NIPS 2001 Brain Computer Interface Workshop dataset [10].[1] This dataset was recorded from a normal subject during a no-feedback session. The subject sat in a normal chair, fingers in the standard typing position at the computer keyboard. The task of the subject was to press with the index and little fingers the corresponding keys in a self-chosen order and timing.

The classification task is to create a classifier to predict which of the two fingers the user intends to use (before pressing the key).

For validation purposes, we have divided the dataset into a training and a test set, containing 412 (80%) and 102 (20%) instances respectively. Due to the long runs required by GP, crossvalidation was not feasible. We took the raw data and selected the 20 last instants of every channel. Allegedly, these instants are more relevant for the classification because they are closer to the time the person makes the decision. Therefore, as there are 27 channels, the initial space has $N = 27 * 20 = 540$ dimensions. The goal is to project this data to a space

[1] http://liinc.bme.columbia.edu/competition.htm

Table 1. Summary of experiments carried out

Experiment	Population size	Generations	Max Nodes
GP1	1000	500	200
GP2	3000	500	70
GP3	3000	500	40

with $M = 3$ dimensions where data can be classified linearly. Table 1 summarizes the 3 experiments carried out in this paper. It displays details about the population size, the maximum number of generations, and the maximum size allowed for the evolved expression (the latter is meant to limit overfitting).

Table 2. Summary of best/median test results for experiments GP1, GP2, and GP3

Experiments	% Test
GP1 (best)	87.5/83.175
GP2 (best)	94.2/86.54
GP3 (best)	94.2/87.5
SMO	93.3
Simple logistics	96.2
ANN	95.1

For every experiment in Table 1, the GP system was run 6 times. The best and median results for every experiment on the test set are summarized in Table 2. Other machine learning approaches have also been tested on the same data. SMO (support vector machine) and Simple Logistic Regression come from the Weka package, using default parameters. They both performed very well in relation to other Weka algorithms. ANN is a backpropagation neural network.

In summary, the best GP individual (94.2%) obtains comparable results to the other systems tested. This result is positive, considering that this is preliminary research, and that we have obtained an expression that projects from 540 to 3 dimensions, where almost linear classification can be carried out.

5 Conclusions

In this paper, we have applied Genetic Programming to evolve functions that project data from spaces with N dimensions to spaces with M dimensions. The main goal is that in the M-space, projected data can be classified linearly (or close to). The secondary goal is to project data into a smaller space, so $M << N$. This approach has been applied to classification of EEG data coming from a Brain Computer Interface competition.

Here, we have applied a straightforward configuration, where the primitives used by GP are arithmetic functions. Results are comparable or inferior to other machine learning methods, but the dimensionality of the problem has been reduced from 540 to only 3. Yet, even for this simple approach and for the important reduction of the problem, we have obtained a projection that classifies linearly the projected data with an accuracy of 94.2%. This shows that the approach has some merit, although our goal of evolving original expressions, better than what can be achieved by traditional machine learning methods, or by human expertise, has not been achieved. In the future, we expect that by carefully choosing primitive functions, known to perform well in the EEG domain, results will be much improved.

Acknowledgements

This article has been financed by the Spanish founded research MCyT project TRACER, Ref:TIC2002-04498-C05-04M.

References

1. Fawcett, T., Utgoff, P.: A hybrid method for feature generation. In: Proceedings of the Eighth International Workshop on Machine Learning, Evanston, IL (1991) 137–141
2. Kramer, S.: Cn2-mci: A two-step method for constructive induction. In: Proceedings of ML-COLT'94. (1994)
3. Pfahringer, B.: Cipf 2.0: A robust constructive induction system. In: Proceedings of ML-COLT'94. (1994)
4. Hoya, T., Hori, G., Bakardjian, H., Nishimura, T., Suzuki, T., Miyawaki, Y., Funase, A., Cao, J.: Classification of single trial eeg signals by a combined principal + independent component analysis and probabilistic neural network approach. In: Proceedings of the fourth International Symposium on Independent Component Analysis and Blind Signal Separation (ICA2003). (2003) 197–202
5. Schlogl, A.: "The Electroencephalogram and the Adaptive Autoregressive Model: theory and applications". Shaker Verlag (2000)
6. Felzer, T.: On the possibility of developing a brain-computer interface (bci). Technical report, Technical University of Darmstadt, Germany (2001)
7. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA, USA (1992)
8. Koza, J.R.: Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press, Cambridge Massachusetts (1994)
9. Koza, J.R., Keane, M.A., Streeter, M.J., Mydlowec, W., Yu, J., Lanza, G.: Genetic Programming IV: Routine Human-Competitive Machine Intelligence. Kluwer Academic Publishers (2003)
10. Blankertz, B., Curio, G., Müller, K.R.: Classifying single trial eeg: Towards brain computer interfacing. In: Advances in Neural Inf. Proc. Systems 14 (NIPS 01). (2002)

SOM-Based Wavelet Filtering for the Exploration of Medical Images

Birgit Lessmann[1], Andreas Degenhard[1], Preminda Kessar[3], Linda Pointon[3], Michael Khazen[3], Martin O. Leach[3], and Tim W. Nattkemper[2]

[1] Theoretical Physics,University of Bielefeld, Bielefeld, Germany
lessmann@physik.uni-bielefeld.de
[2] Applied Neuroinformatics Group, University of Bielefeld, Bielefeld, Germany
[3] Clinical MR Research Group, Institute of Cancer Research, Royal Marsden Hospital, Sutton, Surrey, UK

Abstract. In medical image analysis there are many applications that require the definition of characteristic image features. Especially computationally generated characteristic image features have potential for the exploration of large datasets. In this work, we propose a method for investigating time series of medical images using a combination of the Discrete Wavelet Transform and the Self Organizing Map. Our approach allows relevant image information to be identified in wavelet space. This enables us to develop a filter algorithm suitable to find and extract the characteristic image features and to suppress interfering non-relevant image information.

1 Introduction

The exploration of large medical image databases is currently a very active field of research, especially in relation to the field of Content Based Image Retrieval [1]. For this purpose it is essential to generate characteristic image features for the respective application. One method to compute characteristic features is the Discrete Wavelet Transform (DWT). However, in medical applications the amount of image data is usually very large, whereas the interesting part in the image may be very small. Furthermore it can be difficult to distinguish the relevant features from those derived from non-relevant parts in the image. Therefore it is practical to combine the search for characteristic wavelet features with methods from the field of data-mining and unsupervised learning algorithms. In this work a self organizing map (SOM) is used to explore and visualize the wavelet features of the input space (1). The SOM projects the feature space on a low dimensional grid. This dimension reduction based visualization provides an interface for an interactive manual filtering in the wavelet domain.

We have applied this approach to the exploration of data from magnetic resonance imaging (MRI). This imaging technique is currently being investigated for the detection and classification of breast cancer. In particular, the administration of a contrast agent (Gd-DTPA) gives rise to characteristic signal changes in areas of abnormalities (Dynamic contrast enhanced (DCE) MRI) [2]. In the UK Breast

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 671–676, 2005.

Screening Study, premenopausal women at high genetic risk of developing breast cancer are being evaluated using an imaging and analysis protocol based on DCE-MRI [3]. Two 3D MR images are acquired prior to (pre-contrast) and five images are acquired after contrast agent injection (post-contrast), each with an acquisition time of 90 secs. In this time series of MR images, the specificity is strongly affected by further contrast enhancement within the normal breast parenchyma or within the heart. The region of interest (ROI) containing the enhancing lesions comprises less than 1 % of the whole image. Furthermore, in feature space the wavelet features of this ROI are close to those produced by additional enhancement particularly arising in the region of the heart. Using a time series containing the first pre-contrast and all five post-contrast images we show the possibility of distinguishing the different types of contrast enhancement. This enables us to extract the image information describing the lesion and to suppress the non-relevant spatiotemporal information describing contrast agent uptake in the region of the heart.

2 Discrete Wavelet Transform

Wavelet analysis, including multiresolution analysis, enables the assessment of localized and scale-dependent information in signals and images [5]. A signal f is decomposed using the *Discrete (Dyadic) Wavelet Transform* into a basis of shifted and dilated versions of a *mother wavelet* ψ [4]

$$f(x) \;=\; \sum_{(j,k)} d_{j,k}\psi_{j,k}(x) \;, \qquad \text{with} \quad \psi_{j,k}(x) \;=\; 2^{j/2}\psi(2^j x - k) \;. \tag{1}$$

The index j indicates the dilation or *scaling step* while k refers to translation or shifting. The wavelet coefficients $d_{j,k}$ are given by the scalar product $d_{j,k} = \langle f(x), \psi_{j,k}(x)\rangle$ or $d_{j,k} = \langle f(x), \tilde{\psi}_{j,k}(x)\rangle$ in the case of *biorthogonal wavelets* with the dual wavelet $\tilde{\psi}$ [4]. An efficient calculation of these coefficients is accomplished by the *Fast Wavelet Transform* (FWT), an algorithm allowing the coefficients to be calculated in a stepwise manner. To perform a FWT a scaling function $\phi(x)$ is required such that [4]

$$\phi(x) = \sqrt{2}\sum_k h(k)\phi(2x-k) \qquad \text{and} \qquad \psi(x) = \sqrt{2}\sum_k g(k)\phi(2x-k) \;. \tag{2}$$

The coefficients $h(k)$ and $g(k)$ are termed *Filter coefficients* and they determine the wavelet. On the first scale the signal is decomposed into its *details* and the remaining signal, i.e. the *approximation*, reflecting the particular scale. The details are described by the wavelet coefficients of this scale while the approximation is represented by scaling coefficients corresponding to the scaling function. The procedure can be iterated by a further decomposition of the approximation into details and approximation of the next coarser scale.

3 Self Organizing Maps

The Self Organizing Map (SOM) is a clustering approach from the field of artificial neural networks [5]. A set of reference or prototype vectors $\{\mathbf{u}_i\}$ is trained according to a given data set of feature vectors $\{x_i\}$. The SOM is represented by a two-dimensional square grid, which consists of nodes each correlated with one of the reference vectors. In this work, we use a two-dimensional square grid. During each learning step t a vector from the input space is mapped on the nearest reference vector (winner node). This node and the ones identified as neighbours are updated according to $\mathbf{u}_j(t+1) = \mathbf{u}_j(t) + h_{ij}(t)[x_i - \mathbf{u}_j(t)]$. The function h_{ij} is defined as $h_{ij} = \alpha(t) \exp \frac{-|r_i - r_j|^2}{2\sigma^2(t)}$, where $\alpha(t)$ is the learning rate factor and $\sigma(t)$ defines the neighbourhood function, both monotonically decreasing. The r_i and r_j are the nodes corresponding to the reference vectors $\mathbf{u}_i$ and $\mathbf{u}_j$. After a successful training procedure the reference vectors depict the data distribution in the data set.

4 SOM-Based Wavelet-Feature Extraction

Wavelet Transform of datasets: Each of the six volumes was transformed separately. By combining the wavelet coefficients possessing the same combination of indices at different time points we derived time series of wavelet coefficients. The decomposition included five decomposition steps using the Cohen-Daubechies-Feauveau (CDF)(2,2)-wavelet which has been shown to be appropriate for our kind of filtering procedures [8].

Local dynamic windowing: To explore image databases, the image volumes need to be filtered for selected signals. To this end, one needs to distinguish the feature vectors describing contrast enhancement in tumour tissue (x_i^{T}) from those produced by signal enhancement in the region of the heart (x_i^{H}). For this purpose we used a dynamic local windowing technique, which is based on defining a region of interest (ROI) and a region of exclusion (ROE). The ROI is defined based on an expert label whereas the ROE is given by a labeling of the heart region. We reconstructed a new artificial time series of MR Images by using the dynamic information from the post-contrast image within the ROI/ROE and the static pre-contrast information outside these regions. After carrying out the DWT of these images, the wavelet coefficients describing the specific type of tissue are easy to identify through their significant variation over time. We obtained one set containing about 10^3 feature vectors for contrast agent uptake of tumour tissue and one set containing about 10^5 feature vectors for contrast enhancement in the region of the heart.

Training and filtering procedure: A satisfying projection result was achieved by selecting the difference between coefficients of consecutive time points:

$$x_i = (d_i(2) - d_i(1), d_i(3) - d_i(2), ..., d_i(6) - d_i(5)) * \mathrm{sgn}(d_i(2) - d_i(1)) \qquad (3)$$

where i indicates one combination of scaling and shifting indices. Note that the feature vectors are normalised to the sign of the first difference, represented by the sign function sgn. After the training procedure, it is necessary to identify the reference vectors which dominate the characteristics of the two types of signal enhancement. The total number of feature vectors mapped to one node is not a significant measure because a large number of only slightly varying coefficients may indicate weak structures in the image. Therefore we defined a new energy-measure E, which is correlated to the energy in the subtraction images produced by each time series of coefficients:

$$E(x_i) = \sum_{t=1}^{6} |d_i(t) - d_i(1)| \tag{4}$$

By calculating sums over all feature vectors mapped to one node $\mathbf{u_j}$ we obtain the energy-measures of wavelet coefficients per SOM node for tumour and heart features ($E_{\text{T}}(\mathbf{u_j})$ and $E_{\text{H}}(\mathbf{u_j})$) seperately:

$$E_{\text{T}}(\mathbf{u_j}) = \frac{1}{E_{\text{tot}}^{\text{T}}} \sum_{x_i^{\text{T}} \to \mathbf{u_j}} E(x_i^{\text{T}}) \quad \text{with} \quad E_{\text{tot}}^{\text{T}} = \sum_{\mathbf{u_j}} \sum_{x_i^{\text{T}} \to \mathbf{u_j}} E(x_i^{\text{T}}), \tag{5}$$

and $E_{\text{H}}(\mathbf{u_j})$ respectively with $E_{\text{tot}}^{\text{T}}$ and $E_{\text{tot}}^{\text{H}}$ as normalisation factors. The feature vectors generated from the wavelet coefficients of the scales 2 up to 5 from both tumour and heart tissue were trained using a 20×20 SOM. The feature vectors based on the remaining scaling coefficients were clustered separately using a 10×10 SOM. After the training procedure the nodes with the highest values for E_{T} have been identified as characteristic for the description of tumour enhancement. In the filtering procedure datasets of eight patients with ten tumours, both benign and malignant, were included. The feature vectors of seven patients, calculated using the described local windowing technique were used to train the SOM. Then eight nodes in the SOM with the wavelet coefficients and four nodes in the SOM with scaling coefficients were chosen for the filtering of lesion-related signal enhancement. The original MR images of the eighth patient were then transformed and mapped to the SOM. Only those coefficients whose features were mapped to one of the selected nodes were retained. All other coefficients were set to zero. Finally subtraction images were reconstructed from the remaining wavelet and scaling coefficients, showing the tumour with enhanced grayvalues. Applying thresholding to the subtraction images achieves tumour segmentation with increased accuracy, which has been evaluated using ROC analysis [7] as shown in [8].

5 Results

The complete algorithm is illustrated in figure 1. The SOM was trained with the wavelet features of all datasets. The icons represent the SOM nodes, the box plot in each icon depicts the reference vector of the specific node. The grayvalue within the icon indicates the value E_{T} for each node. While the corresponding value E_{H}, marked by the linewidth of the box, is localized in the margin of the complete SOM, there are only a few nodes showing a high value for E_{T}. At the bottom of the figure 1 one result of the filtering procedure is exemplified.

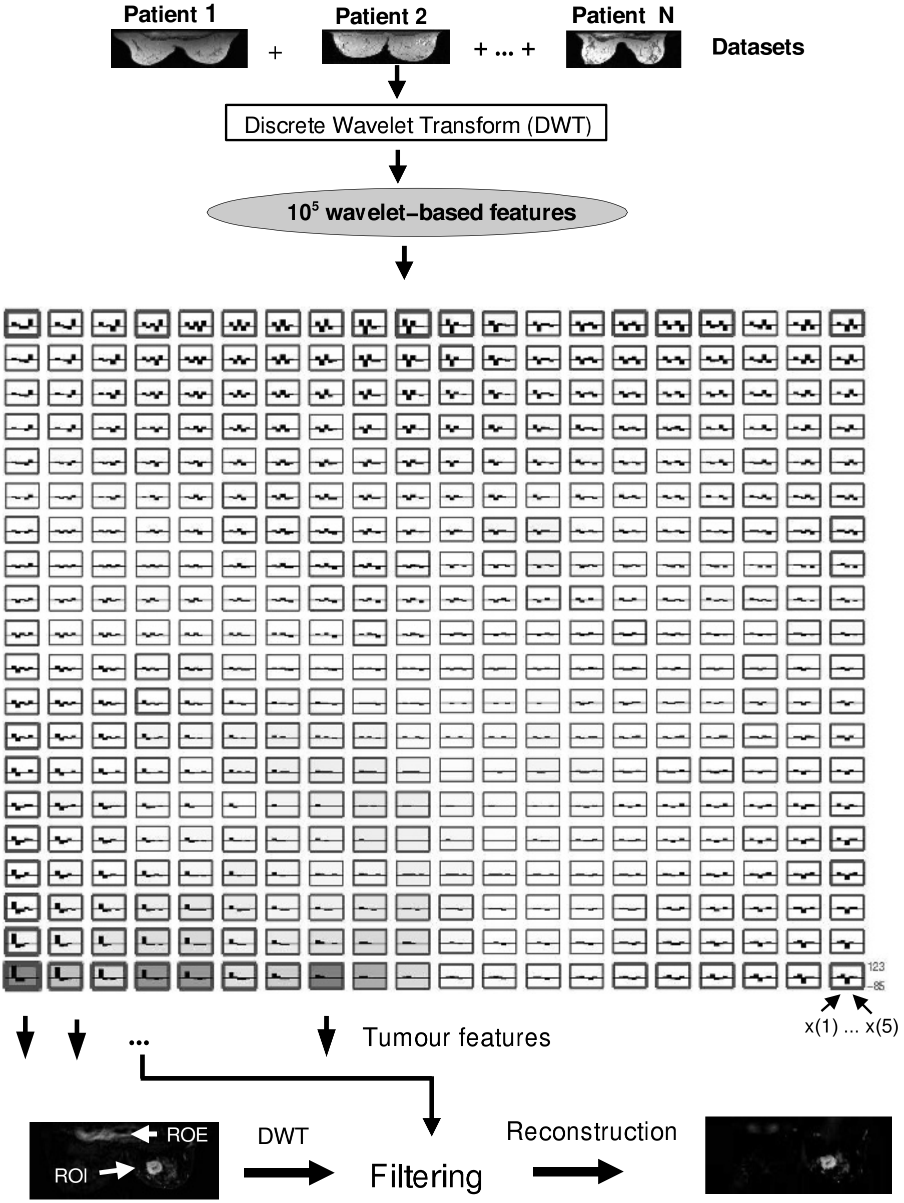


Fig. 1. Complete algorithm including the result of a SOM-training and a filtering step. The plots in the grid depict the reference vectors. The grayvalue of the background marks the value of E_{T}, the linewidth of the box indicates the value of E_{H} represented by the specific node. At the bottom two subtraction images are visible, before (left) and after (right) application of the filtering procedure.

The subtraction image at the left was calculated from the raw images (first precontrast and first post-contrast) while the subtraction image at the right was computed from the filtered images. Obviously additional enhancement, in particular visible in the region of the heart, is significantly decreased. Table 1 shows the results of the ROC-analysis. The areas under the curve (AUC) for specificity parameters higher than 0.95 for the original and the filtered subtraction images were calculated. The AUC-values have been normalised to this area. The sensitivity and thus the AUC-value is significantly increased in all of these cases, due to the filtering of additional signal enhancement.

Table 1. Areas under the curve for specificity ≥ 0.95

patient	1	2	3	4	5	6	7	8
original subtraction image	0.820	0.552	0.664	0.570	0.000	0.785	0.805	0.721
filtered subtraction image	**0.853**	**0.745**	**0.713**	**0.813**	**0.002**	**0.876**	**0.895**	**0.946**

6 Discussion and Conclusion

We have demonstrated the potential of a self organizing map for the feature space exploration of medical image datasets. Computationally generated wavelet features can be visualized efficiently which allows a customized manual filtering procedure. Applied to DCE-MRI datasets this approach leads to filtered datasets showing a significantly increased sensitivity for high specificity. This is due to the extraction of the small amount of lesion-related information from the large amount of additional interfering signal enhancement produced by contrast agent uptake in other types of tissue. A thorough comparison to other mammographic analysis methods is shifted to a more detailed paper.

References

[1] Mueller, H., Michoux, N., Bandon, D., Geissbuhler, A.: A review of Content-Based Image Retrieval Systems in Medical Applications - Clinical Benefits and Future Directions. Int. J. Med. Inf. **73** (2004) 1-23

[2] Heywang-Koebrunner, S.H., Beck, R.: Contrast-Enhanced MRI of the Breast. Springer-Verlag, Berlin (1996)

[3] UK MRI Breast Screening Study Advisory Group: Magnetic resonance imaging screening in women at genetic risk of breast cancer: imaging and analysis protocol for the UK multicentre study. Magn. Reson. Imaging **18** (2000) 765

[4] Daubechies, I.: Ten Lectures on Wavelets. CBMS-NFS Series Appl. Math., SIAM (1991)

[5] Mallat, S., Zhong, S.: Characterization of Signals from Multiscale Edges. IEEE Trans. Patt. Anal. Machine Intell. **14** (1992) 710

[6] Kohonen, T.: Self Organizing Maps. Springer, Berlin, Heidelberg (1995)

[7] Fawcett, T.: ROC Graphs: Notes and Practical Considerations for Researchers. HP Labs Tech Report HPL-2003-4, (2003)

[8] Lessmann, B., Twellmann, T., Degenhard, A., Nattkemper, T.W., Leach., M.O. et al: Wavelet-Features for improved Tumour Detection in DCE-MRI. Proc. Medical Image Understanding and Analysis, BVMA, (2004) 93-96

Functional MRI Analysis by a Novel Spatiotemporal ICA Algorithm

Fabian J. Theis, Peter Gruber, Ingo R. Keck, and Elmar W. Lang

Institute of Biophysics, University of Regensburg, 93040 Regensburg, Germany

Abstract. Data sets acquired from functional magnetic resonance imaging (fMRI) contain both spatial and temporal structures. In order to blindly extract underlying activities, the common approach however only uses either spatial or temporal independence. More convincing results can be achieved by requiring the transformed data to be as independent as possible in both domains. First introduced by Stone, spatiotemporal independent component analysis (ICA) is a promising algorithm for fMRI decomposition. We propose an algebraic spatiotemporal ICA algorithm with increased performance and robustness. The feasibility of the algorithm is demonstrated in an application to the analysis of an fMRI data sets of a human brain performing an auditory task.

1 Introduction

Spatiotemporal data analysis in contrast to the more common methods of either spatial or temporal analysis tries to achieve both spatial and temporal separation by optimizing a joint energy function. First proposed by Stone et al [4], it is a promising method, which has potential applications in biomedical data analysis. We extend his approach by generalizing algebraic ICA algorithms to the spatiotemporal case. In [6] we introduce a framework for spatiotemporal data analysis based on so-called double-sided joint diagonalization. In this paper, after quickly recalling these results, we apply the diagonalization to fourth-order cumulant in order to get stJADE, a generalization of the well-known temporal JADE algorithm. The power of stJADE is illustrated in an application to data acquired from functional magnetic resonance imaging (fMRI).

2 Spatiotemporal Independent Component Analysis

Source Separation. Noiseless *blind source separation* (BSS) denotes the following problem: let $\mathbf{x}(t)$ be an (observed) stationary m-dimensional random vector and $\mathbf{A}$ a full rank matrix such that

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) \tag{1}$$

where the n-dimensional source signals $\mathbf{s}(t)$ are commonly assumed to be stochastically independent: $p_{\mathbf{s}}(s_1, \ldots, s_n) = p_{s_1}(s_1) \ldots p_{s_n}(s_n)$. Recovering the underlying sources given only $\mathbf{x}(t)$ using the independence assumption is called *independent component analysis (ICA)*.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 677–682, 2005.

Due to independence, the fourth-order cross cumulants of the sources have to be trivial. In order to find transformations of the mixtures fulfilling this property, the well-known JADE algorithm [1] jointly diagonalizes the *contracted quadricovariance matrices* defined by

$$\mathbf{C}_{ij}(\mathbf{x}) := E\left(\mathbf{x}^\top \mathbf{E}_{ij}\mathbf{x}\mathbf{x}\mathbf{x}^\top\right) - \mathbf{R}_\mathbf{x}\mathbf{E}_{ij}\mathbf{R}_\mathbf{x} - \operatorname{tr}(\mathbf{E}_{ij}\mathbf{R}_\mathbf{x})\mathbf{R}_\mathbf{x} - \mathbf{R}_\mathbf{x}\mathbf{E}_{ij}\mathbf{R}_\mathbf{x}.$$

Here $\mathbf{E}_{ij}$ is a set of eigen-matrices of $\mathbf{C}_{ij}$, $1 \leq i,j \leq m$. One simple choice is to use m^2 matrices $\mathbf{E}_{ij}$ with zeros everywhere except 1 at index (i,j). More elaborate choices of eigen-matrices (with only $m(m+1)/2$ or even m entries) are possible.

Spatiotemporal Structure. Real-world data sets often possess structure in addition to the necessary instantaneous independence required by ICA. For example fMRI measurements contain both temporal and spatial indices so a data entry $x = x(a,b,c,t)$ can depend on position (a,b,c) as well as time t. More generally, we want to consider data sets $x(\mathbf{r},t)$ depending on two indices $\mathbf{r}$ and t, where $\mathbf{r} \in \mathbb{R}^n$ can be a multidimensional index and t indexes the time axis. In reality this generalized random process is realized by a finite number of samples. For example in the case of fMRI scans we could assume $t \in [1:T] := \{1,2,\ldots,T\}$ and $\mathbf{r} \in [1:h]\times[1:w]\times[1:d]$, where T is the number of scans, which were of size $h \times w \times d$. So the number of spatial observations is ${}^{\mathbf{s}}m := hwd$ and the number of temporal observations ${}^{\mathbf{t}}m = T$.

In the following, the spatial multi-dimensional index $\mathbf{r}$ is contracted into a one-dimensional index r by row concatenation. Then the data set $x(r,t) =: x_{rt}$ can be represented by a data matrix $\mathbf{X}$ of dimension ${}^{\mathbf{s}}m \times {}^{\mathbf{t}}m$, and the goal is to determine either a spatial source matrix ${}^{\mathbf{s}}\mathbf{S}$ or a temporal source matrix ${}^{\mathbf{t}}\mathbf{S}$ (with corresponding mixing matrices ${}^{\mathbf{s}}\mathbf{A}$ and ${}^{\mathbf{t}}\mathbf{A}$ respectively). After mean removal we can without loss of generality assume that the mixtures are spatiotemporally centered.

Spatiotemporal ICA. Temporal ICA is equivalent to the matrix factorization $\mathbf{X} = {}^{\mathbf{t}}\mathbf{A}{}^{\mathbf{t}}\mathbf{S}$, whereas spatial ICA implies the factorization $\mathbf{X}^\top = {}^{\mathbf{s}}\mathbf{A}{}^{\mathbf{s}}\mathbf{S}$ or equivalently $\mathbf{X} = {}^{\mathbf{s}}\mathbf{S}^\top{}^{\mathbf{s}}\mathbf{A}^\top$. Hence

$$\mathbf{X} = {}^{\mathbf{t}}\mathbf{A}{}^{\mathbf{t}}\mathbf{S} = {}^{\mathbf{s}}\mathbf{S}^\top{}^{\mathbf{s}}\mathbf{A}^\top \tag{2}$$

So both source separation models can be interpreted as matrix factorization problems; in the temporal case independence is required from the second factor, in the spatial case from the first one. Now instead of recovering a single spatiotemporally independent source data set we try to find *two* source matrices, a spatial and a temporal source matrix, and the conditions are put onto the matrices separately. So the *spatiotemporal ICA* model can be formulated by the factorization problem

$$\mathbf{X} = {}^{\mathbf{s}}\mathbf{S}^\top{}^{\mathbf{t}}\mathbf{S} \tag{3}$$

with spatial source matrix ${}^{\mathbf{s}}\mathbf{S}$ and temporal source matrix ${}^{\mathbf{t}}\mathbf{S}$ being as independent as possible.

Independence is invariant under scaling and permutation, so the above model contains the same indeterminacy — indeed the spatial and temporal sources can interchange scaling ($\mathbf{L}$) and permutation ($\mathbf{P}$) matrices, ${}^{\mathbf{s}}\mathbf{S}^{\top}{}^{\mathbf{t}}\mathbf{S} = (\mathbf{L}^{-1}\mathbf{P}^{-1}{}^{\mathbf{s}}\mathbf{S})^{\top}$ $(\mathbf{L}\mathbf{P}{}^{\mathbf{t}}\mathbf{S})$. Apart from that, in the case in which the conditions are fulfilled perfectly, the proofs of temporal uniqueness [3, 5] can easily be transferred to the above problem. However, if the source conditions hold jointly but only approximately for ${}^{\mathbf{s}}\mathbf{S}$ and ${}^{\mathbf{t}}\mathbf{S}$, uniqueness results are unknown so far.

3 An Algorithm for Spatiotemporal ICA

Stone [4] first proposed the model from equation (3), where he employs a joint energy function based on mutual entropy and infomax. Apart from the many parameters used in the algorithm, the involved gradient descent optimization is susceptible to noise, local minima and inappropriate initializations, so we propose a novel, more robust algebraic approach based on joint diagonalization in the following.

Double-Sided Joint Diagonalization. We will recall the algorithm derivation for the spatiotemporal ICA, which in a more general form has been recently introduced in [6]; it is based on the joint diagonalization of cumulant matrices posed not only temporally but also spatially.

Shifting to matrix notation, we interpret $\mathbf{C}_{ij}(\mathbf{X}) := \mathbf{C}_{ij}({}^{\mathbf{t}}\mathbf{x}(t))$ as the (i, j)-th temporal cumulant matrix, whereas $\mathbf{C}_{ij}(\mathbf{X}^{\top}) := \mathbf{C}_{ij}({}^{\mathbf{s}}\mathbf{x}(r))$ is to denote the corresponding spatial cumulant matrix. Application of the spatiotemporal mixing model from equation (3) together with the transformation properties of the cumulants yields

$$\mathbf{C}_{ij}({}^{\mathbf{t}}\mathbf{S}) = {}^{\mathbf{s}}\mathbf{S}^{\dagger\top}\mathbf{C}_{ij}(\mathbf{X}){}^{\mathbf{s}}\mathbf{S}^{\dagger} \quad \text{and} \quad \mathbf{C}_{ij}({}^{\mathbf{s}}\mathbf{S}) = {}^{\mathbf{t}}\mathbf{S}^{\dagger\top}\mathbf{C}_{ij}(\mathbf{X}^{\top}){}^{\mathbf{t}}\mathbf{S}^{\dagger} \tag{4}$$

because ${}^{*}m \geq n$ and hence ${}^{*}\mathbf{S}{}^{*}\mathbf{S}^{\dagger} = \mathbf{I}$. By assumption the matrices $\mathbf{C}_{ij}({}^{*}\mathbf{S})$ are as diagonal as possible. In order to separate the data, we have to find diagonalizers for both $\mathbf{C}_{ij}(\mathbf{X})$ and $\mathbf{C}_{ij}(\mathbf{X}^{\top})$ such that they satisfy the spatiotemporal model (3). As $\mathbf{X}$ (or matrices derived from it) have to be diagonalized in terms of both columns and rows, this is denoted by *double-sided approximate joint diagonalization*. This process will be reduced to the common approximate joint diagonalization in the following.

Dimension Reduction. In order to get robust cumulant estimates, dimension reduction is essential, i.e. we want to extract only $n \ll \min\{{}^{\mathbf{s}}m, {}^{\mathbf{t}}m\}$ sources. For this consider the singular value decomposition $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^{\top}$ of $\mathbf{X}$, and permute the diagonal matrix $\mathbf{D}$ (and corresponding columns of $\mathbf{U}$ and $\mathbf{V}$) such that $\mathbf{D}$ contains the eigenvalues in decreasing order in its main diagonal. By only choosing the first n columns of $\mathbf{U}$ and $\mathbf{V}$ and the upper-left $n \times n$ submatrix of $\mathbf{D}$, we get a decomposition again denoted by $\hat{\mathbf{X}} := \mathbf{U}\mathbf{D}\mathbf{V}^\top$, which is an estimate of $\mathbf{X}$ using only the n largest eigenvalues. The matrices $\mathbf{U} \in \mathbb{R}^{{}^{\mathbf{s}}m \times n}$ and $\mathbf{V} \in \mathbb{R}^{{}^{\mathbf{t}}m \times n}$ are again pseudo-orthogonal, and $\mathbf{D}$ is diagonal. So $\mathbf{X} \approx \mathbf{U}\mathbf{D}\mathbf{V}^\top = \left(\mathbf{U}\mathbf{D}^{1/2}\right)\left(\mathbf{V}\mathbf{D}^{1/2}\right)^\top$. This is a matrix factorization of $\mathbf{X}$ into two decorrelated signals $\mathbf{U}\mathbf{D}^{1/2}$ and $\mathbf{V}\mathbf{D}^{1/2}$. After dimension reduction, the spatiotemporal BSS model (3) can only hold approximately: $\mathbf{X} \approx \hat{\mathbf{X}} = {}^{\mathbf{s}}\mathbf{S}^\top {}^{\mathbf{t}}\mathbf{S}$ — now ${}^{\mathbf{s}}\mathbf{S}$ and ${}^{\mathbf{t}}\mathbf{S}$ are of reduced (row) size n. Plugging this model into the above equation together with the pseudo-orthogonality of $\mathbf{U}$ and $\mathbf{V}$ yields $\left(\mathbf{U}\mathbf{D}^{-1/2}\right)^\top {}^{\mathbf{s}}\mathbf{S}^\top {}^{\mathbf{t}}\mathbf{S}\left(\mathbf{V}\mathbf{D}^{-1/2}\right) = \mathbf{I}$. Hence $\mathbf{W} := {}^{\mathbf{t}}\mathbf{S}\mathbf{V}\mathbf{D}^{-1/2}$ is an invertible $n \times n$ matrix.

Algorithm. By model (3) we get ${}^{\mathbf{s}}\mathbf{S}^\top = \mathbf{X}{}^{\mathbf{t}}\mathbf{S}^\dagger$. Applying this to the first formula in equation (4) yields, after some calculation,

$$\mathbf{C}_{ij}({}^{\mathbf{t}}\mathbf{S}) = {}^{\mathbf{t}}\mathbf{S}\mathbf{X}^\dagger \mathbf{C}_{ij}(\mathbf{X})\mathbf{X}^{\dagger\top}{}^{\mathbf{t}}\mathbf{S}^\top = \mathbf{W}\mathbf{C}_{ij}(\mathbf{D}^{1/2}\mathbf{V}^\top)\mathbf{W}^\top.$$

By using $\mathbf{W}^{-1} = \mathbf{D}^{-1/2}\mathbf{V}^\top {}^{\mathbf{t}}\mathbf{S}^\dagger$, we can derive a similar result from (4), 2nd term:

$$\mathbf{C}_{ij}({}^{\mathbf{s}}\mathbf{S}) = {}^{\mathbf{t}}\mathbf{S}^{\dagger\top}\mathbf{C}_{ij}(\mathbf{X}^\top){}^{\mathbf{t}}\mathbf{S}^\dagger = \mathbf{W}^{-\top}\mathbf{C}_i(\mathbf{D}^{1/2}\mathbf{U}^\top)\mathbf{W}^{-1}$$

which we can now invert to get $\mathbf{C}_{ij}({}^{\mathbf{s}}\mathbf{S})^{-1} = \mathbf{W}\mathbf{C}_{ij}(\mathbf{D}^{1/2}\mathbf{U}^\top)^{-1}\mathbf{W}^\top$. So the double-sided joint diagonalization can be simply performed by jointly diagonalizing the twice as large set of matrices

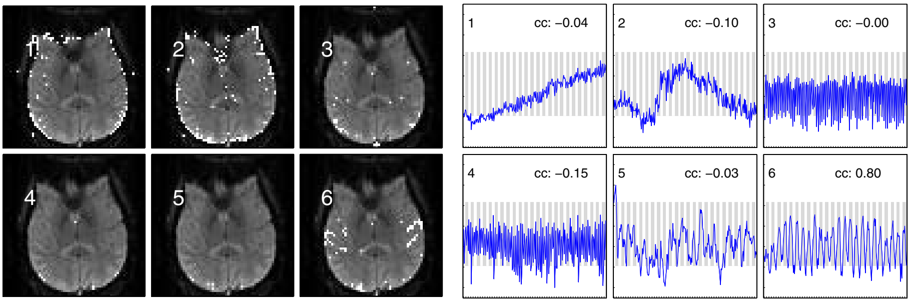
$$\{\alpha\mathbf{C}_{ij}(\mathbf{D}^{1/2}\mathbf{V}^\top),\ (1-\alpha)\mathbf{C}_{ij}(\mathbf{D}^{1/2}\mathbf{U}^\top)^{-1} \mid i = 1, \ldots\} \tag{5}$$

where the weighting factor $\alpha \in [0, 1]$ has been introduced to balance between either spatial or temporal separation. Joint diagonalization is usually performed by optimizing an off-diagonal criterion, so different scale factors in the matrices indeed yield different optima if the diagonalization cannot be achieved fully. Furthermore, the higher α the more temporal separation is stressed.

Similar to the temporal ICA algorithm JADE [1], we algorithmically perform joint diagonalization using an iterative construction of $\mathbf{A}$ by Givens rotation in two coordinates [2] or an non-orthogonal extension of this idea [7]. If $\mathbf{A}$ is a joint diagonalizer of (5), the sources are estimated by ${}^{\mathbf{t}}\hat{\mathbf{S}} = \mathbf{A}^\top\mathbf{D}^{1/2}\mathbf{V}^\top$ and ${}^{\mathbf{s}}\hat{\mathbf{S}} = \mathbf{A}^{-1}\mathbf{D}^{1/2}\mathbf{U}^\top$. This algorithm, denoted by *spatiotemporal JADE (stJADE)*, is freely available as MATLAB-implementation at `http://fabian.theis.name/`.

4 Application to fMRI

We analyze an fMRI data set that was recorded from a healthy male subject (normal hearing) listening to an auditory stimulus consisting of beeps and words. Here a beep denotes a sinusoidal sound of frequencies (uniformly) randomly chosen from 400 to 600 Hz, and words were chosen at random from a database of 100 German words spoken by a single female speaker. The design consisted of blocks of 10 seconds of audio and 10 seconds of silence. The recorded time



(a) component maps

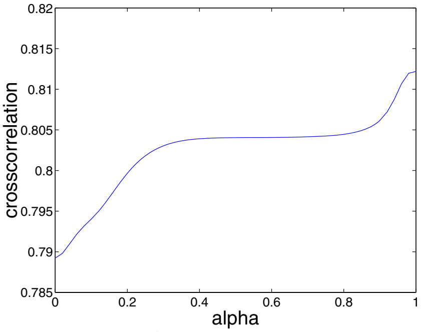
(b) time courses

Fig. 1. fMRI analysis using stJADE ($\alpha = 0.5$). The data was reduced to the first 6 principal components. (a) shows the recovered component maps (white points indicate values stronger than 2.5 standard deviations), and (b) their time courses. Component 6 is the desired stimulus component, which is mainly active in the auditory cortex; its time-course closely follows the on-off stimulus (indicated by the gray boxes) — their crosscorrelation lies at $cc = 0.8$ — with a delay of roughly 6 seconds induced by the BOLD effect.

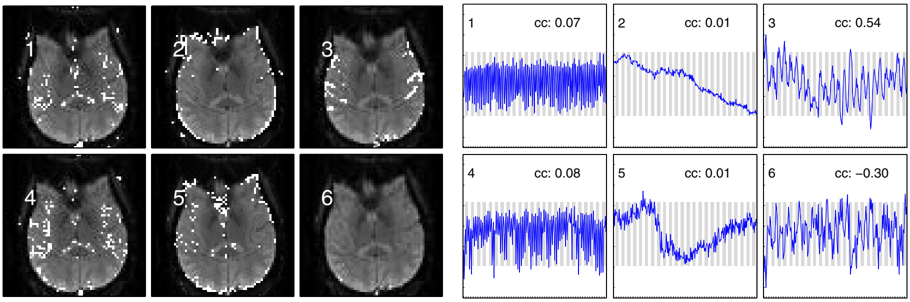
was 480 seconds and fMRI scans were obtained every 2 seconds. Thus, the data consist of 240 time points. In the audio blocks, the proportion of words and beeps varied and the exact starting times of beeps and words were randomized.

As preprocessing a single data slice is extracted (after motion correction and realignment) and the data is masked to include only voxels from within the brain. The stJADE algorithm is applied with $\alpha = 0.5$ and orthogonal matrix recovery. Figure 1 shows the spatial sources together with the recoveries using stJADE. The algorithm is able to recover the auditory stimulus well with a high crosscorrelation of 0.9.

The figure on the right-hand side also shows the behavior of stJADE with varying α. Apparently due to the strong temporal structure within the data sets, temporal extraction of the stimulus component outperforms spatial one. However this is problem-specific, and often spatial separation is preferable or some method in between, hence an adaptive choice of α as proposed in spatiotemporal ICA is superior.



For comparison, we also apply the often used fastICA algorithm to spatially separate the data. The result is shown in figure 2. FastICA performs considerably worse and cannot fully detect the stimulus component (crosscorrelation 0.5). This is most probably due to the fact that spatial separation alone is not enough and the kurtosis condition in fastICA is not properly fulfilled by the data.



(a) component maps

(b) time courses

Fig. 2. fMRI analysis using spatial fastICA. The stimulus component has also been extracted (3), however its crosscorrelation with the stimulus is remarkably lower $cc = 0.5$ than the one from stJADE.

5 Conclusion

We have studied a novel spatiotemporal ICA algorithm, stJADE, based on the double-sided joint diagonalization as generalization of the often applied 'single-sided' joint diagonalization in temporal-only BSS. The presented results for fMRI data sets are promising, and the weighting parameter α allows for additional flexibility in the extraction.

Acknowledgements. The data has been kindly provided by Roland Rutschmann, Oliver Baumann, Ferenc Acs and Marc Greenlee, Department of Psychology, University of Regensburg. The authors gratefully acknowledge partial financial support by the DFG (GRK 638) and the BMBF (project 'ModKog').

References

1. J.-F. Cardoso and A. Souloumiac. Blind beamforming for non gaussian signals. *IEE Proceedings - F*, 140(6):362–370, 1993.
2. Jean-François Cardoso and Antoine Souloumiac. Jacobi angles for simultaneous diagonalization. *SIAM J. Mat. Anal. Appl.*, 17(1):161–164, January 1995.
3. P. Comon. Independent component analysis - a new concept? *Signal Processing*, 36:287–314, 1994.
4. J.V. Stone, J. Porrill, N.R. Porter, and I.W. Wilkinson. Spatiotemporal independent component analysis of event-related fmri data using skewed probability density functions. *NeuroImage*, 15(2):407–421, 2002.
5. F.J. Theis. A new concept for separability problems in blind source separation. *Neural Computation*, 16:1827–1850, 2004.
6. F.J. Theis, P. Gruber, I.R. Keck, A. Meyer-Bäse, and E.W. Lang. Spatiotemporal blind source separation using double-sided approximate joint diagonalization. In *Proc. EUSIPCO 2005 accepted*, Antalya, Turkey, 2005.
7. A. Yeredor. Non-orthogonal joint diagonalization in the leastsquares sense with application in blind source separation. *IEEE Trans. Signal Processing*, 50(7):1545–1553, 2002.

Early Detection of Alzheimer's Disease by Blind Source Separation, Time Frequency Representation, and Bump Modeling of EEG Signals

François Vialatte[1], Andrzej Cichocki[2], Gérard Dreyfus[1],
Toshimitsu Musha[3], Sergei L. Shishkin[2], and Rémi Gervais[4]

[1] ESPCI (ParisTech), Laboratoire d'Electronique (CNRS UMR 7084),
10 rue Vauquelin, 75005 Paris, France
{francois.vialatte, gerard.dreyfus}@espci.fr
[2] BSI RIKEN ABSP Lab 2-1 Hirosawa, Wako, Saitama, 351-0198, Japan
cia@brain.riken.jp
[3] Brain Functions Laboratory Inc., KSP Building E211, Sakado, Takatsu Kawasaki-shi,
Kanagawa, 213-0012, Japan
musha@bfl.co.jp
[4] Equipe Neurobiologie de la Mémoire Olfactive, Institut des Sciences Cognitives
(UMR 5015 CNRS UCB), 67 Boulevard Pinel, 69675 Bron Cedex, France
gervais@isc.cnrs.fr

Abstract. The early detection Alzheimer's disease (AD) is an important challenge. In this paper, we propose a novel method for early detection of AD using electroencephalographic (EEG) recordings: first a blind source separation algorithm is applied to extract the most significant spatio-temporal components; these components are subsequently wavelet transformed; the resulting time-frequency representation is approximated by sparse "bump modeling"; finally, reliable and discriminant features are selected by orthogonal forward regression and the random probe method. These features are fed to a simple neural network classifier. The method was applied to EEG recorded in patients with Mild Cognitive Impairment (MCI) who later developed AD, and in age-matched controls. This method leads to a substantially improved performance (93% correctly classified, with improved sensitivity and specificity) over classification results previously published on the same set of data. The method is expected to be applicable to a wide variety of EEG classification problems.

1 Introduction

Alzheimer's disease (AD) is the most common neurodegenerative disorder. Since the number of individuals with AD is expected to increase in the near future, early diagnosis and effective treatment of AD are critical issues in neurophysiological research [1], [2]. Finding a computational method for early identification of patients who are to progress towards Alzheimer's disease (before onset of AD), but do not exhibit any clinical signs of AD at the time of the test, is thus an important challenge. Furthermore, an early detection method should be inexpensive, in order to allow mass screening of elderly patients [1]–[6]. Electroencephalography (EEG) is one of the most promising candidates in that respect.

W. Duch et al. (Eds.): ICANN 2005, LNCS 3696, pp. 683 – 692, 2005.

Due to the high complexity and variability of EEG signals, early detection of AD from EEG recordings requires the development of efficient signal processing tools [2]. In [5], Blind Source Separation (BSS) was first applied for these purposes, while standard methods were used for feature extraction and classification. In the present paper, we propose a multistage procedure employing blind source separation for filtering/enhancement of EEG, time frequency representation, subsequent bump modeling for feature generation and dimensionality reduction, and statistical feature selection. We show that it provides a further improvement in classification of AD patients and healthy subjects as compared to similar classification results obtained previously [1], [5] on the same data set.

2 Methods

2.1 Blind Source Separation for Signal Filtering

According to the currently prevailing view of EEG signal processing, a signal can be modeled as a linear mixture of a finite number of brain sources, with additive noise [5,6]. Therefore, blind source separation techniques can be used advantageously for decomposing raw EEG data to brain signal subspace and noise subspace.

In [5], the AMUSE (Algorithm for Multiple Unknown Signals Extraction [7], [8], [9], [10]) algorithm was used in order to select the five significant components of the signal that had the best linear predictability. That algorithm belongs to the group of second-order-statistics spatio-temporal decorrelation (SOS-STD) blind source separation algorithms. It relies on the idea that the estimated components should be spatio-temporally decorrelated, and be less complex (i.e., have better linear predictability) than any mixture of those sources. Therefore, the components are ranked in order of decreasing singular values of a time-delayed covariance matrix. As in PCA (Principal Component Analysis), and unlike in many ICA algorithms, all components estimated by AMUSE are uniquely defined (i.e., any run of algorithms on the same data will always produce the same components) and consistently ranked.

The algorithm can be considered as two consecutive PCAs: first, PCA is applied to input data; secondly, PCA (SVD) is applied to the time-delayed covariance matrix of the results of the previous PCA. In the first step, standard or robust prewhitening (sphering) is applied as a linear transformation

$$\mathbf{z}(t) = \mathbf{Q}\,\mathbf{x}(t) \tag{1}$$

where $\mathbf{Q} = \mathbf{R}_x^{-\frac{1}{2}}$, $\mathbf{R}_x$ is the standard covariance matrix $\mathbf{R}_x = \mathrm{E}\left\{\mathbf{x}(t)\mathbf{x}^{\mathrm{T}}(t)\right\}$ and $\mathbf{x}(t)$ is a vector of observed data at time t. Next, SVD is applied to a time-delayed covariance matrix of pre-whitened data:

$$\mathbf{R}_z = \mathrm{E}\left\{\mathbf{z}(t)\mathbf{z}^{\mathrm{T}}(t-1)\right\} = \mathbf{U\Sigma V}^{\mathrm{T}} \tag{2}$$

where $\mathbf{\Sigma}$ is a diagonal matrix with decreasing singular values and $\mathbf{U}$, $\mathbf{V}$ are matrices of eigenvectors. Then, a demixing (separating) matrix is estimated as:

$$\mathbf{W} = \hat{\mathbf{A}}^{-1} = \mathbf{U}^{\mathrm{T}}\mathbf{Q} \tag{3}$$

The algorithm is much faster than the vast majority of BSS/ICA algorithms (its computation time depends essentially on the duration of the PCA procedure) and is very easy to use, because no parameters are required. It is implemented as a part of package "ICALAB for signal processing" [11], freely available online. However, the algorithm is quite sensitive to sensor (measurement) noise; therefore, alternative ICA/BSS algorithms with suitable ranking of components can be considered in further studies [7,11].

2.2 Database

The "at-risk" state for AD is commonly referred to as Mild Cognitive Impairment (MCI) [3]. In the course of a clinical study [1], patients who complained of memory impairment only, but had no apparent loss in general cognitive, behavioral, or functional status, were recruited. Both patients and controls underwent general medical, neurological, psychiatric, and neuroimaging (SPECT, CT and MRI) investigation for more accurate diagnosis. EEG was recorded from all patients and controls, within one month of study inception; the present analysis made use of EEG recorded from the patients who later developed AD, and age-matched controls. Electrodes were located on 21 sites according to the 10-20 system, with the reference electrode on the right ear-lobe. Sampling rate was 200 Hz, analog filter bandpass 0.5-250 Hz. In [5], the first continuous artifact-free 20 s interval of each recording were used to create two datasets:

- the MCI set, featuring 22 EEG recordings of elderly patients matching the criteria of mild cognitive impairment, who developed AD within one year and a half;
- a control set, featuring 38 recordings from age-matched family members of the patients.

In the present paper, the same pre-processing method as in [5] was used on the same 60 recordings, as a baseline for assessing the efficiency of the detection obtained by the present method: a database (hereinafter referred to as *D*) containing those five components was generated (five top ranked components obtained using AMUSE).

2.3 Time-Frequency Maps and Bump Modeling for Feature Generation

In order to obtain a compact representation of the signals of *D*, suitable for automatic discrimination of MCI patients from control individuals, the signals were first analyzed in the time-frequency domain by a wavelet transformation, and the resulting time-frequency maps were modeled by bumps [12], as described below.

2.3.1 Wavelet Transformation and Time-Frequency Map Generation

EEG signals were first transformed to time-frequency maps using wavelets (see [13] for details). Complex Morlet wavelets [14] are appropriate for time-frequency analysis of electroencephalographic signals ([15], [16], [17], [18]). Complex Morlet wavelets $w(t)$ of Gaussian shape in time (deviation σ_t) are defined as:

$$w(t) = A\exp\left(-t^2/2\sigma_t^2\right)\exp\left(2i\pi ft\right) \tag{4}$$

where σ_t and f are appropriately chosen parameters; they cannot be chosen independently, since the product $\sigma_t f$ determines the number of periods that are present in the wavelet. In the present investigation, the wavelet family defined by $2\pi\sigma_t f = 7$ was chosen, as described in [15].

The signals present in database *D* were wavelet-transformed in the frequency range 1.5 to 31.5 Hz, discretized in 0.25 Hz frequency bins.

2.3.2 Bump Modeling

The bump modeling technique [12] is a 2-dimensional generalization of the Gaussian mesa function modeling technique that was initially designed for one-dimensional signals (electrocardiogram analysis- ECG) [19], [20]. In the present study, it was used for extracting information from the time-frequency maps. In previous investigations, it was also successfully applied to the analysis of local field potential signals, gathered from electrophysiological (invasive) measurements [21],[12]; the present paper reports the first application of bump modeling to surface EEG signals.

The main idea of this method is to approximate a time-frequency map with a set of predefined elementary parameterized functions called bumps (non-overlapping or overlapping); therefore, the map is represented by the set of parameters of the bumps, which is a very sparse encoding of the map, resulting in information compression rates that range from one hundred to one thousand (further details are given in [12], [19], [20]).

The algorithm performs the following steps on the time-frequency maps (after appropriate normalization [12]):

- (i) window the map in order to define the zones to be modeled (those windows form a set of overlapping sub-areas of the map),
- (ii) find the window that contains the maximum amount of energy,
- (iii) adapt a bump φ_b to the selected zone, and withdraw it from the original map. The parameters of the bumps are computed using the BFGS algorithm [22] in order to minimize the cost function *C* defined by:

$$C = \frac{1}{2}\sum_{t,f\in W}\left(z_{f\,t} - \varphi_b(f,t)\right)^2 \tag{5}$$

where the summation runs on all pixels within the window *W*, $z_{f\,t}$ are the time-frequency coefficients at time *t* and frequency *f*, and $\varphi_b\left(f,t\right)$ is value of the bump function at time *t* and frequency *f*;

- (iv) if the amount of information modeled by the bumps reaches a threshold, stop; else return to (iii).

Half ellipsoids were found to be the most appropriate bump functions for the present application (Figure 2 shows a typical example of bump modeling of the time-frequency map of an EEG recording). Each bump is described by 5 parameters: its coordinates on the map (2 parameters), its amplitude (one parameter) and the lengths of its axes (2 parameters). Half ellipsoids (Figure 1) are defined by:

$$\begin{aligned} \varphi_b(f,t) &= a\sqrt{1-v} && \text{for } 0 \le v \le 1 \\ \varphi_b(f,t) &= 0 && \text{for } v > 1 \end{aligned} \tag{6}$$

where $v = (e_f^2 + e_t^2)$ with $e_f = (f - \mu_f)/\, l_f$ and $e_t = (t - \mu_t)/\, l_t$. μ_f and μ_t are the coordinates of the centre of the ellipsoid, l_f and l_t are the half-lengths of the principal axes, a is the amplitude of the function, t is the time and f the frequency.

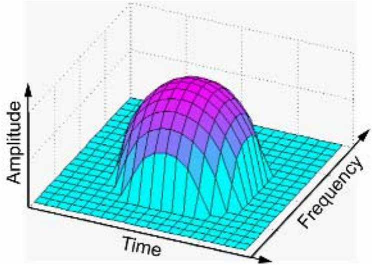


Fig. 1. Half ellipsoid function

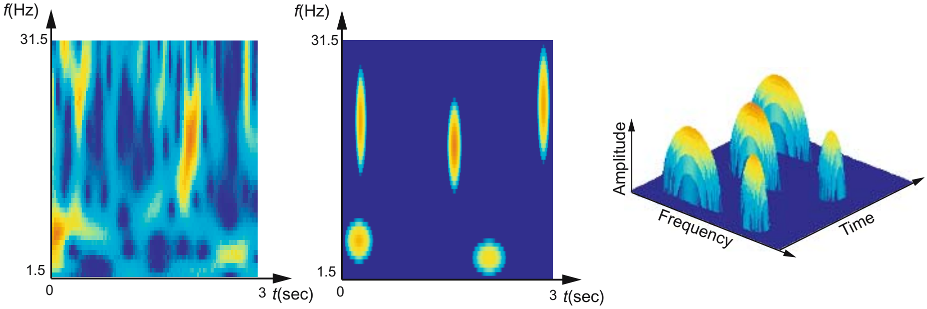


Fig. 2. Left: normalized time-frequency map of the first ICA source of an EEG recording (Control set); middle and right: 2D and 3D bump modeling of the map

After bump modeling, the parameters of the bumps are candidate features for classification. Although the model is sparse, feature selection is mandatory because of the small size of the data set.

2.4 Feature Selection

After bump modeling, the signals under investigation are represented by the set of parameters that describe the bumps. Within that set, an even more compact representation was sought, based on expert knowledge on the frequency sub-bands of interest. For ease of comparison of the results, the boundaries for five standard EEG sub-bands were defined as in the previous study [5] of the same data set: θ(3.5-7.5 Hz), α_1 (7.5-9.5 Hz), α_2 (9.5-12.5 Hz), β_1 (12.5-17.5Hz) and β_2 (17.5-25Hz). The following features were defined and computed for each sub-band:

- F_1: the number of bumps,
- F'_1: the number of high-amplitude bumps (normalized amplitude > 0.7),
- F_2: the sum of the amplitudes of the bumps present,
- F'_2: the sum of the amplitudes of the high-amplitude bumps present,
- F_3: the maximal amplitude of the bumps present.

Two groups of candidate features were defined: group A contains $\{F_1, F'_1, F_2, F'_2\}$ and group B contains $\{F_1, F'_1, F_3\}$ only. Thus, either 3 or 4 features were computed for each sub-band, depending on the group of features under consideration. Therefore, for database D (5 time-frequency maps), the number of candidate features N_f was either 75 or 100. Since the number of candidate features was still too large given the number of examples in the database (only 60), feature selection was performed by orthogonal forward regression (OFR) algorithm [23, 24] and the random probe method.

First, the candidate features are ranked in order of decreasing relevance by OFR. OFR operates in observation space, i.e. in a space whose dimension is equal to the number of observations (equal to 60 in the present work). In that space, the quantity to be modeled, and the candidate features, are represented by vectors denoted by **y** (desired outputs) and $\mathbf{u}_i$, $i = 1$ to N_f (inputs). The OFR algorithm performs the following steps:

- (i) compute the angle Θ_i between each candidate feature $\mathbf{u}_i$ and the quantity to be modeled **y** and select the candidate feature $\mathbf{u}_j$ that has the smallest angle with **y**, i.e. the candidate feature that is most correlated to **y**:

$$\mathbf{u}_j = \arg\max_i \{\cos^2(\Theta_i)\} \tag{7}$$

- (ii) project **y** and all the remaining candidate features onto the null space of the selected feature;

The above two steps can be iterated in subspaces of decreasing dimensions until all candidate features are ranked. Subsequently, in order to select the optimal number of features from their ranked list, the random probe method [24] is applied. One hundred "probes", i.e. realizations of random variables, are computed and appended to the feature set. A risk level P is defined [25], which corresponds to the risk that a feature might be kept although, given the available data, it might be less relevant than the probe. The following steps are performed iteratively:

- (i) obtain a candidate feature from OFR,
- (ii) compute the value of the cumulative distribution function of the rank of the probe for the rank of the candidate feature,
- (iii) if that value is smaller than the risk, select the feature and return to (i);
- (iv) else, discard the candidate feature under consideration and terminate.

In the present case, an ensemble feature ranking method [25] was used: 60 subsets were built by iteratively removing one example from the database. OFR and the random probe method were then applied to those subsets. The overall distribution of features, and the average number N_k of selected features were computed; finally, the N_k overall best features were selected.

3 Results

Each dataset was used for training and validating a neural network classifier (multilayer perceptron model, see for instance [26]). The generalization performance was estimated using the leave-one-out cross-validation method [27] which was also used under the name of "jackknifing" in the previous study of the data set [5]. The best results, shown in Table 1, were obtained with linear classifiers (no hidden layer).

The method used in [5] to asses BSS performances was applied with a simple PCA for comparison: with the five first PCA components back-projected, relative spectral

Table 1. Number of subjects correctly and incorrectly classified by neural network models, using D, depending on the feature group (A contains $\{F_1, F'_1, F_2, F'_2\}$ and B contains $\{F_1, F'_1, F_3\}$). Results were obtained using the leave-one-out cross-validation method, validation set results are presented below. P is the risk of a false positive feature, as defined in the text.

Datasets	Misclassified		Correctly classified %		
	MCI $N = 22$	Controls $N = 38$	MCI $N = 22$	Controls $N = 38$	All $N = 60$
A group → F =12, P = 12% Components 1,2,3 and 5 +Bumps	2	2	91.0	94.7	93.3
A group → F = 6, P = 9% Components 1-5 +Bumps	4	4	81.8	88.6	86.7
B group → F = 11, P = 10% Components 1-5 + Bumps	2	3	91.0	92.1	91.7
Previous study, best results [5] (without bumps)	6	6	72.7	84.2	80.0
PCA components 1-5 (without bumps)	13	10	40.9	73.7	61.7

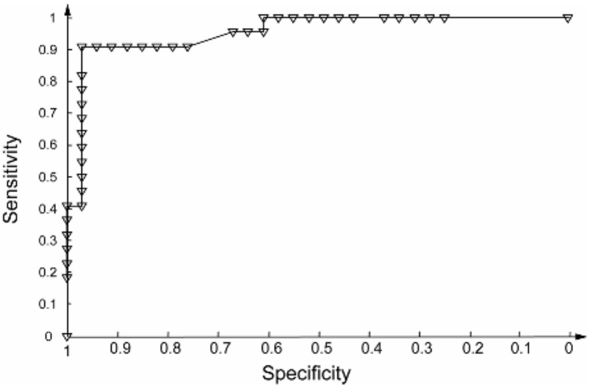


Fig. 3. R.O.C curve for the best classification results, obtained using a neural network on data set D, with components 1-3 and 5 found by the AMUSE algorithm (but without component 4)

powers of signals were computed by dividing the power in δ(1.5- 3.5 Hz), θ(3.5-7.5 Hz), α_1 (7.5-9.5 Hz), α_2 (9.5-12.5 Hz), β_1 (12.5-17.5Hz) and β_2 (17.5-25Hz) bands by the power in the 1.5-25 Hz band; those values were subsequently normalized using the transformation $\log\frac{p}{1-p}$, where p is the relative spectral power; finally the band power values were averaged over all 21 channels (see [5] for details). PCA did not exhibit good performance compared to BSS (Table 1, last two rows).The best results were obtained with the group of candidate features A, with components 1-3 and 5 found by the AMUSE algorithm, but without component 4 (component 4 was removed before OFR and the random probe method, R.O.C. curve is represented in Figure 3). For comparison, the fourth row of the Table reports previous results [5] obtained with the same EEG recordings with a different representation.

4 Discussion

In the present paper, we reported the first application of blind source separation combined with time frequency representation and sparse bump modeling to the automatic classification of EEG data for early detection of Alzheimer's disease. The developed method was applied to recordings that had been analyzed previously [5] with standard feature extraction and classification methods. With respect to that previous analysis, a substantial improvement was achieved, the overall correct classification rate being raised from 80% to 93% (sensitivity 91.0% and specificity 94.7%).

The task was the discrimination of EEG recordings of normal individuals from EEG recordings of patients who developed Alzheimer's disease one year and a half later. Therefore, the present study provides exciting prospects for early mass detection of the disease. The method is very cheap as compared to PET, SPECT and fMRI, requiring only a 21-channel EEG apparatus. Note that short intervals (20 seconds) of artifact-free recording of spontaneous EEG was already sufficient for high accuracy of classification.

PCA showed much poorer results than AMUSE algorithm, which demonstrates the significance of AMUSE (or more generally, BSS/ICA algorithms with suitable ranking and clustering) for EEG filtering/enhancement. Furthermore, sparse bump modeling appeared to be a valuable tool for compressing information contained in EEG time-frequency maps. Amplitude variations and bursts of EEG oscillations are highly related to the brain state dynamics [28]. Bump modeling can provide a good approximation of time-frequency maps; since it models appropriately important features of EEG oscillations, it is a promising tool for compact feature extraction, as demonstrated in the present paper.

Although our preliminary results are quite promising, a full validation of the method requires investigating more extensive databases. Furthermore, there is presumably a lot of information present in the recordings that is not yet exploited, such as the dynamics of the bumps and the brain functional connectivity. This will be the subject of future research.

References

1. Musha, T., Asada, T., Yamashita, F., Kinoshita, T., Chen, Z., Matsuda, H., Masatake, U., Shankle, W.R., A new EEG method for estimating cortical neuronal impairment that is sensitive to early stage Alzheimer's disease. Clinical Neurophysiology, 2002, 113(7): 1052-1058.
2. Jeong, J., EEG dynamics in patients with Alzheimer's disease. Clinical Neurophysiology, 2004, 115:1490-1505.
3. DeKosky, S.T., Marek, K., Looking backward to move forward: early detection of neurodegenerative disorders. Science, 2003, 302(5646):830-834.
4. Nestor, P.J., Scheltens, P., Hodges J.R., Advances in the early detection of Alzheimer's disease. Nature Medicine, 2004, 10 Suppl.:S34-41. Review.
5. Cichocki, A., Shishkin, S.L., Musha, T., Leonowicz, Z., Asada, T., Kurachi, T., EEG filtering based on blind source separation (BSS) for early detection of Alzheimer's disease. Clinical Neurophysiology, 2005, 116(3):729-737.
6. Cichocki, A., Blind Signal Processing Methods for Analyzing Multichannel Brain Signals. International Journal of Bioelectromagnetism, 2004, Vol. 6, N°1.
7. Cichocki, A., Amari, S., Adaptative Blind Signal and Image Processing: Learning Algorithms and Applications. New York, NY: Wiley, 2003.
8. Tong, L., Soon, V., Huang, Y.F., Liu R., Indeterminacy and identifiability of blind identification. IEEE Transactions CAS, 1991, 38:499-509.
9. Tong, L. Inouye, Y., Liu, R., Waveform-preserving blind estimation of multiple independent sources. IEEE Transactions on Signal Processing, 1993, 41(7):2461-2470.
10. Szupiluk, R., Cichocki, A., Blind signal separation using second order statistics. Proceedings of SPETO, 2001, 485-488.
11. Cichocki, A., Amari, S., Siwek, K., Tanaka, T., et al. ICALAB toolboxes [available online at http://www.bsp.brain.riken.jp/ICALAB]
12. Vialatte, F., Martin, C., Dubois, R., Quenet, B., Gervais R., Dreyfus G. A machine learning approach to the analysis of time-frequency maps, and its application to neural dynamics. Neural Networks (submitted).
13. Poularikas, A.D., The transforms and applications handbook. CRC Press, 1996.
14. Kronland-Martinet, R., Morlet, J., Grossmann, A., The wavelet Transform, in Expert Systems and Pattern Analysis, C.H Chen Edt, World Scientific, 1987, pp 97-126.
15. Tallon-Baudry, C., Bertrand, O., Delpuech, C., Pernier, J., Stimulus specificity of phase-locked and non-phase-locked 40 Hz visual responses in human. Journal of Neuroscience, 1996, 16:4240-4249.
16. Ohara, S., Crone, N.E., Weiss, N., Lenz, F.A., Attention to a painful cutaneous laser stimulus modulates electrocorticographic event-related desychronization in humans. Clinical Neurophysiology, 2004, 115:1641-1652.
17. Caplan, J.B., Madsen, J.R., Raghavachari, S., Kahana, M.J., Distinct patterns of brain oscillations underlie two basic parameters of human maze learning . Journal of Neurophysiology, 2001, 86:368-380.
18. Düzel, E., Habib, R., Schott, B., Schoenfeld, A., Lobaugh, N., McIntosh, A.R., Scholz, M., Heinze, H.J., A multivariate, spatiotemporal analysis of electromagnetic time-frequency data of recognition memory. Neuroimage, 2003, 18:185-197.
19. Dubois, R., Application de nouvelles méthodes d'apprentissage à la détection précoce d'anomalies en électrocardiographie. PhD thesis, Université Pierre et Marie Curie - Paris VI, 2004. [Available from http://www.neurones.espci.fr/Francais.Docs/dossier_recherche/bibliographie/theses_soutenues.htm#ancre41560]

20. Dubois, R., Quenet, B., Faisandier, Y., Dreyfus, G., Building meaningful representations in knowledge-driven nonlinear modeling. Neurocomputing, , 2005 (in print).
21. Vialatte, F., Martin, C., Ravel, N., Quenet, B., Dreyfus, G., Gervais, R., Oscillatory activity, behaviour and memory, new approaches for LFP signal analysis. 35 th annual general meeting of the European brain and behaviour society, 17-20 September 2003, Barcelona, Spain – Acta Neurobiologiae Experimentalis, Vol. 63, supplement 2003.
22. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T., Numerical Recipes in C: The Art of Scientific Computing, 425 - 430. 1992, Cambridge Univ. Press, New York.
23. Guyon, I., Elisseef, A., An Introduction to Variable and Feature Selection. Journal of Machine Learning Research, 2003, 3:1157-1182.
24. Chen, S., Billings, S. A. and Luo, W., Orthogonal least squares methods and their application to non-linear system identification. International Journal of Control, 1989, 50:1873-1896.
25. Jong, K., Marchiori, E., Sebag, M., Ensemble Learning with Evolutionary Computation: Application to Feature Ranking. 8th International Conference on Parallel Problem Solving from Nature (PPSN). Lecture Notes in Computer Science, 2004, 1133-1142. Springer.
26. Stoppiglia, H., Dreyfus, G., Dubois, R., Oussar, Y., Ranking a Random Feature for Variable and Feature Selection. Journal of Machine Learning Research, 2003, 3: 1399-1414.
27. Haykin, S., Neural Networks – a comprehensive foundation. 2° edition, 1999, Prentice Hall.
28. Stone, M., Cross-validatory choice and assessment of statistical predictions (with discussion). Journal of the Royal Statistical Society: Series B, 1974, 36:111–147.
29. Klimesch, W., EEG alpha and theta oscillations reflect cognitive and memory performance: a review and analysis. Brain Research Reviews 1999, 29: 169-95.

Acknowledgements to the Reviewers

We wish to express our sincere gratitude to the following reviewers and advisors for their valuable remarks and sugestions:

C. Aaron
S. Abe
R. Adamczak
F. Aiolli
A. Albrecht
E. Alhoniemi
H. Amin
A. Anastasiadis
R. Andonie
P. Andras
D. Anguita
C. Angulo-Bahon
B. Apolloni
C. Archambeau
M. Atencia
J. Avrithis
Y. Avrithis
G. Bedoya
M. Bianchini
R. Birkenhead
L. Bobrowski
M. Bogdan
S. Bohte
A. Boni
E. Bottino
S. Breutel
J. Bródka
D. Buldain
P. Campadelli
A. Cangelosi
B. Caputo
G. Cawley
L.-W. Chan
M. Chetouani
A. Chung Tsoi
A. Cichocki
E. Corchado
M. Cottrell
S. Coupland
N. Crook
L. Csato
P. Dario
N. Delannay
C. Dimitrakakis
K. Doherty
E. Dominguez Merino
G. Dorffner
W. Duch
D. Elizondo
P. Erdi
M. Faundez-Zanuy
M. Fernandez-Redondo
A. Flanagan
P. Fleury
R. Folland
D. Franois
C. Fyfe
C. Garcia-Osorio
N. Garcia-Pedrajas
B. Gas
S. Gielen
M. Gola
M. Gongora
L. Gonzales Abril
K. Goser
B. Gosselin
K. Grąbczewski
M. Grana
R. Grothmann
M. Grzenda
B. Hammer
R. Haschke
G. Heidemann
J. Henderson

J. Himberg
E. Hines
J. Hollmen
A. Honkela
O. Hryniewicz
D. Huang
T. Huang
M. Huelse
A. Hussain
C. Igel
G. Indiveri
S. Ishii
Y. Ito
N. Jankowski
M. Jirina
R. John
G. Joya
A. Kaban
J. Kacprzyk
M. de Kamps
B. Kappen
J. Karhunen
N. Kasabov
S. Kasderidis
D. Kim
M. Kimura
S. Kollias
J. Korbicz
J. Koronacki
W. Kosiński
M. Koskela
O. Kouropteva
R. Kozma
M. Krawczak
L. Kruś
F. Kurfess
M. Kurzyński
J. Laaksonen
E. Lang
V. Laxmi
E. Leclercq
J. Lee
P. Lehtima̋ki
K. Leiviska̋
A. Lendasse
A. Likas
T. Lourens
C. Lu
B. Macukow
J. Madrenas
M. Maggini
B. Maillet
N. Mammone
D. Mandic
J. Mańdziuk
T. Marcu
U. Markowska-Kaczmar
M. Martin-Merino
T. Martinetz
F. Masulli
J. Meeus
J. Meller
A. Menciassi
A. Micheli
L. Morra
F. Moutarde
N. Mtetwa
R. Muresan
M. Nakayama
N. Nedjah
L. Niklasson
K. Nikolopoulos
W. Nowak
D. Obradovic
E. Oja
F. Okulicka
M. Olteanu
D. Ortiz Boyer
S. Osowski
X. Parra
H. Paugam-Moisy
C. Pedreira
W. Pedrycz
K. Pelckmans
B. Pelletier
J. Pestian
G. Peters
A. Piegat
J. Pizarro Junquera
D. Polani

M. Porrmann
R. R. Poznański
J. Prévotet
C. Puntonet
D. Puzenat
F. Queirolo
T. Raiko
K. Raivio
C. Reyes Garca
M. Rocha
O. Rochel
M. Rodriguez-Alvarez
A. Romariz
F. Rossi
S. Rovetta
D. Rutkowska
L. Rutkowski
J. Rynkiewicz
J. Salojarvi
J. Salotti
J. Sarela
B. Schrauwen
F. Schwenker
U. Seiffert
M. Sfakiotakis
V. Siivola
G. Simon
O. Simula
R. Słowiński
D. Sona
A. Sperduti
A. Stafylopatis
G. Stamou
J. Steil
M. Steuer
M. Strickert
M. Sugiyama
J. Suykens
P. Szczepaniak
R. Tadeusiewicz
R. Tagliaferri
J. G. Taylor
M. Terra
F. Theis
A. Tomé
E. Trentin
D. Tsakiris
V. Tzouvaras
S. Usui
M. Valle
F. van der Velde
T. Van Gestel
S. Van Looy
M. Vannucci
G. Vaucher
J. Venna
J. Verbeek
M. Verleysen
N. Viet
V. Vigneron
A. E. P. Villa
T. Villmann
J. Vitay
M. Wallace
Z. Waszczyszyn
N. Watanabe
T. Wennekers
S. Wermter
W. Wiegerinck
B. Wilamowski
A. Wilbik
K. Wills
P. Wira
A. Wróbel
B. Wyns
S. Xavier de Souza
I. Yaesh
A. Yanez Escolano
Y. Yang
Z. Yang
J. Yearwood
S. Zadrożny
H. Zimmermann
A. Żochowski
J. Żurada

Author Index