ADVANCES IN INTELLIGENT SYSTEMS AND COMPUTING 198

Pablo A. Estévez
José C. Príncipe
Pablo Zegers (Eds.)

Advances in Self-Organizing Maps

Springer

Advances in Intelligent Systems and Computing 198

For further volumes:
http://www.springer.com/series/11156

Pablo A. Estévez, José C. Príncipe,
and Pablo Zegers (Eds.)

Advances in Self-Organizing Maps

9th International Workshop, WSOM 2012
Santiago, Chile, December 12–14, 2012
Proceedings

Springer

Editors

Prof. Pablo A. Estévez
Department of Electrical Engineering
University of Chile
Santiago
Chile

Prof. Pablo Zegers
Facultad de Ingeniería y Ciencias Aplicadas
Universidad de los Andes
Santiago
Chile

Prof. José C. Príncipe
Computational NeuroEngineering Laboratory
University of Florida
Gainesville, FL
USA

ISSN 2194-5357
e-ISSN 2194-5365
ISBN 978-3-642-35229-4
e-ISBN 978-3-642-35230-0
DOI 10.1007/978-3-642-35230-0
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012952597

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This book contains refereed papers presented at the 9th Workshop on Self-Organizing Maps (WSOM 2012) held at the Universidad de Chile, Santiago, Chile, on December 12–14, 2012. The workshop brought together researchers and practitioners in the field of self-organizing systems. Among the book chapters there are excellent examples of the use of SOMs in agriculture, computer science, data visualization, health systems, economics, engineering, social sciences, text and image analysis, and time series analysis. Other chapters present the latest theoretical work on SOMs as well as Learning Vector Quantization (LVQ) methods.

Our deep appreciation is extended to Teuvo Kohonen, for serving as Honorary General Chair and for enthusiastically supporting the idea of holding the workshop for the first time in Latin-America.

We warmly thank the members of the Steering Committee and the Executive Committee. In particular we thank to Guilherme Barreto, Publicity Chair, and Timo Honkela, for handling the papers with conflict of interest.

Our sincere thanks go to Barbara Hammer and José Príncipe for their plenary talks. We are grateful to the members of the Program Committee and other reviewers for their excellent and timely work, and above all to the authors whose contributions made this book possible.

October 2012

Pablo A. Estévez
José C. Príncipe
Pablo Zegers

Organization

WSOM 2012 was held during December 12–14, 2012, organized by the Department of Electrical Engineering, University of Chile.

Executive Committee

Honorary Chair

Teuvo Kohonen	Academy of Finland, Finland

General Chair

Pablo A. Estévez	University of Chile, Chile

General Co-chair

José C. Príncipe	University of Florida, USA

Program Chair

Pablo Zegers	Universidad de los Andes, Chile

Publicity Chair

Guilherme A. Barreto	Federal University of Ceará, Brazil

Steering Committee

Teuvo Kohonen	Academy of Finland, Finland
Marie Cottrell	Université Paris 1, Pantheón-Sorbonne, France
Pablo Estévez	University of Chile, Chile
Timo Honkela	Aalto University, Finland
Erkki Oja	Aalto University, Finland

José Príncipe	University of Florida, USA
Helge Ritter	Bielefeld University, Germany
Takeshi Yamakawa	Kyushu Institute of Technology, Japan
Hujun Yin	University of Manchester, UK

Program Committee

José Aguilar	Universidad de los Andes, Venezuela
Guilherme Barreto	Federal University of Ceará, Brazil
Ernesto Cuadros-Vargas	San Pablo Catholic University, Peru
Yoonsuck Choe	Texas A&M University, USA
Marie Cottrell	Université Paris 1, Pantheón-Sorbonne, France
Tetsuo Furukawa	Kyushu Institute of Technology, Japan
Barbara Hammer	Bielefeld University, Germany
Timo Honkela	Aalto University, Finland
Ryotaro Kamimura	Tokai University, Japan
Olga Kurasova	Vilnius University, Lithuania
Jorma Laaksonen	Aalto University, Finland
Kai Labusch	University of Lübeck, Germany
Jean-Charles Lamirel	LORIA, France
Ezequiel López-Rubio	University of Malaga, Spain
Thomas Martinetz	University of Lübeck, Germany
Haruna Matsushita	Kagawa University, Japan
Erzsébet Merenyi	University of Rice, USA
Antonio Neme	Autonomous University of Mexico City, Mexico
Yoshifumi Nishio	Tokushima University, Japan
Erkki Oja	Aalto University, Finland
Ioannis Pitas	University of Thessanoliki, Greece
Marina Resta	University of Genova, Italy
Udo Seiffert	Otto von Guericke University of Magdeburg, Germany
Leticia Seijas	University of Buenos Aires, Argentina
Olli Simula	Aalto University, Finland
Marc Strickert	Philipps-University of Marburg, Germany
Kadim Tasdemir	European Commission Joint Research Centre, Italy
Heizo Tokutaka	SOM Japan Inc., Japan
Michel Verleysen	Université Catholique de Louvain, Belgium
Thomas Villmann	University of Applied Sciences Mittweida, Germany
Pablo Zegers, Chair	Universidad de los Andes, Chile
Jacek Zurada	University of Louisville, USA

Additional Referees

Rewbenio Frota	Federal University of Ceará, Brazil
José E. Maia	Federal University of Ceará, Brazil
César L. Cavalcante Mattos	Federal University of Ceará, Brazil
Isaque Monteiro	Federal University of Ceará, Brazil
Ajalmar R. Da Rocha Neto	Federal University of Ceará, Brazil
Luis G. Mota Souza	Federal University of Ceará, Brazil
Matashige Oyabu	Kanazawa Institute of Technology, Japan
Masaaki Ohkita	SOM Japan Inc., Japan
Claudio Held	University of Chile, Chile
Jan Chorowski	University of Louisville, USA
Henry Schütze	University of Lübeck, Germany
Jens Hocke	University of Lübeck, Germany
Ioannis Chantas	University of Thessanoliki, Greece
Alexandros Iosifidis	University of Thessanoliki, Greece
Symeon Nikitidis	University of Thessanoliki, Greece

Contents

Plenary

Theory and Methodology

Image Processing

Learning Vector Quantization

Nonlinear Analysis and Time Series

Text Mining and Language Processing

Applications of Data Mining and Analysis

How to Visualize Large Data Sets?

Barbara Hammer, Andrej Gisbrecht, and Alexander Schulz

University of Bielefeld - CITEC Centre of Excellence, Germany
bhammer@techfak.uni-bielefeld.de

Abstract. We address novel developments in the context of dimensionality reduction for data visualization. We consider nonlinear non-parametric techniques such as t-distributed stochastic neighbor embedding and discuss the difficulties which are encountered if large data sets are dealt with, in contrast to parametric approaches such as the self-organizing map. We focus on the following topics, which arise in this context: (i) how can dimensionality reduction be realized efficiently in at most linear time, (ii) how can nonparametric approaches be extended to provide an explicit mapping, (iii) how can techniques be extended to incorporate auxiliary information as provided by class labeling?

1 Introduction

Due to an increasing size and complexity of modern data sets, visualization plays a crucial role in many applications: it offers an intuitive interface based on the human vision system as one of our most powerful senses, displaying astonishing cognitive capabilities as regards e.g. instantaneous grouping of visual objects, structure or outlier detection. Visualization-based approaches are regarded as a major technology to put the human into the loop in complex data analysis tasks, as specified e.g. in the emerging research area of scalable visual analytics [20].

Within machine learning, visualization of data sets by means of dimensionality reduction techniques has encountered a recent boom, resulting in numerous popular nonlinear dimensionality reduction techniques such as t-distributed stochastic neighbor embedding (t-SNE), locally linear embedding (LLE), maximum variance unfolding (MVU), neighborhood retrieval visualizer (NeRV), Isomap, Isotop, Laplacian eigenmaps (LE), maximum entropy unfolding (MEU), and many others [13,18,17,20,19]. These techniques carry the promise to arrive at a very flexible visualization of data such that also subtle nonlinear structures can be spotted. Nevertheless, in large scale practical applications or software systems for data analysis, the visualization techniques which are almost exclusively used are classical linear principal component analysis (PCA) or variations such as multi-dimensional scaling (MDS) [2], and the self-organizing map (SOM) [12].

What are the reasons that these classical techniques are often preferred in practical applications, albeit recent nonlinear dimensionality reduction could offer more flexibility? There are a couple of reasons: Both, PCA and SOM rely on very intuitive principles as regards both, learning algorithms and their final result: they capture directions in the data of maximum variance, globally for PCA

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 1–12.
DOI: 10.1007/978-3-642-35230-0_1

and locally for SOM; online learning algorithms such as online SOM training or the Oja learning rule mimic fundamental principles as found in the human brain, being based on the Hebbian principle accompanied by topology preservation in case of SOM [12]. In addition to this intuitive training procedure and outcome, both techniques have severe practical benefits: training can be done efficiently in linear time only, which is a crucial prerequisite if large data sets are dealt with. In addition, both techniques do not only project the given data set, but they offer an explicit mapping of the full data space to two dimensions by means of an explicit linear mapping in case of PCA and a winner takes all mapping based on prototypes in case of SOM. In contrast, many recent dimensionality reduction techniques belong to the class of non-parametric techniques which do not provide direct out-of-sample extensions. Moreover, they are often based on pairwise distances and, thus, scale at least quadratically with the size of the input set, making them infeasible for large scale applications.

In this contribution, we discuss recent developments connected to the question of how to make non-parametic dimensionality reduction techniques feasible for large data sets, endowing the techniques with linear complexity. A standard approach is based on subsampling of the data only. Two crucial questions arise: while sampling, only a part of the data is mapped directly, and out-of-sample extensions for the rest are required. How can non-parametric techniques be extended to provide explicit dimensionality reduction mappings which are flexible enough to capture local nonlinearities in the data? In addition, only part of the information available in the data is used if dimensionality reduction relies on a subsample only. Thus, possibly, not enough information is yet available to extract the relevant structures from the data. How can dimensionality reduction be enriched to incorporate additional information about the data structure such that valid inference can be done based on few data points only?

Now, we will first shortly review popular dimensionality reduction techniques. Afterwards, we address the question how to enhance non-parametric techniques towards an explicit mapping prescription, emphasizing kernel t-SNE as one particularly flexible approach in this context. Finally, we consider discriminative dimensionality reduction based on the Fisher information, testing this principle in the context of kernel t-SNE and emphasizing a particularly efficient realization in the context of parametric approaches.

2 Dimensionality Reduction

Assume a high dimensional input space X is given, e.g. $X \subset \mathbb{R}^N$. Data $\mathbf{x}_i, i = 1, \ldots, m$ in X should be projected to points $\mathbf{y}_i, i = 1, \ldots, m$ in the projection space $Y = \mathbb{R}^2$ such that as much structure as possible is preserved. The notion of 'structure preservation' is ill-posed and many different mathematical specifications of this term have been used in the literature. One of the most classical algorithms is PCA which maps data linearly to the directions with largest variance, corresponding to the eigenvectors with largest eigenvalues of the data covariance matrix. PCA constitutes one of the most fundamental approaches and one example of two different underlying fundamental principles [16]: (i) PCA maximizes the data likelihood assumed data are generated linearly from a two-dimensional latent space. (ii) PCA constitutes the linear transformation which minimizes the deviation of original data and its projection in a least squares sense. The first motivation treats PCA as a generative model, the latter as a cost minimizer. Due to the simplicity of the underlying mapping, the results coincide.

This is, however, not the case for general non-linear approaches. Roughly speaking, there exist two opposite ways to introduce dimensionality reduction: the generative, often parametric approach, which takes the point of view that high dimensional data points are generated by a low dimensional structure which can be visualized directly, and the cost-function based, often non-parametric approach, which, on the opposite, tries to find low-dimensional projection points such that the characteristics of the original high-dimensional data are preserved as much as possible. A third principled approach, which we will not address here, is based on an encoding framework such as auto-encoder networks [18].

Parametric Approaches

Parametric approaches include the classical SOM and its probabilistic counterpart generative topographic mapping (GTM) [3]. Both approaches are based on a low dimensional latent space, the regular SOM grid or the real plane with a probability distribution peaked at regular grid positions for GTM. These points are associated to high dimensional coordinates in the data space, the parameters of the mapping, called prototypes $\mathbf{w}_j$, which are directly assigned to grid positions by means of the index in case of SOM, or which are images of a parameterized generalized linear function $\Phi : Y \rightarrow X$ in case of GTM. These prototypes are determined such that they generate the data as accurately as possible: for SOM, the mean reconstruction error is minimized, for GTM, the data likelihood is maximized, centering Gaussians at the prototypes. Thereby, a visualization by means of back-projection of the prototypes to the low-dimensional latent space is possible by ensuring topology preservation of the mapping: neighbored prototypes are similar because of a direct integration of neighborhood information in SOM training, while GTM achieves this fact by relying on a smooth mapping Φ.

We do not discuss training or an exact mathematical derivative in more detail, referring to the excellent literature [12,3]. We just would like to stress the following points, which make these techniques particularly suitable if large data sets are dealt with: both techniques provide an explicit mapping prescription of data points X to the low-dimensional visualization space. For SOM, a data point $\mathbf{x}$ is mapped to the grid position of the closest prototype $\mathbf{w}_i$. For GTM, the image can be determined based on the responsibilities of prototypes for the data points and the explicit generalized linear mapping Φ. This explicit out-of-sample extension of both, SOM and GTM makes them suitable if a mapping based on a small subset of points is considered only.

In addition to this benefit, SOM and GTM can be trained in linear time in the size of the training set only, whereby quantization in terms of the prototypes takes place. Hence the prototypes which determine the number of parameters of the mapping allow a problem dependent adaptation of the computational costs of the models. This is another crucial aspect which makes the techniques particular well suited if large data sets are dealt with.

Parametric Approaches for Dissimilarity Data

We would like to stress one recent line of research for such parametric models: often, data are not given as explicit vectors, rather pairwise similarities or dissimilarities d_{ij} characterize the relation between data point $\mathbf{x}_i$ and $\mathbf{x}_j$. For these settings, SOM and GTM have been extended based on kernelization or relational approaches, respectively, see e.g. [21,9]. Basically, prototypes are represented implicitly in terms of linear combinations $\sum_k \alpha_{jk}\mathbf{x}_j$ of data with coefficients α_{jk} which sum up to 1, and distances of data points and prototypes are computed by the formula $\|\mathbf{x}_i - \mathbf{w}_j\|^2 = \langle \mathbf{x}_i, \mathbf{x}_i \rangle - 2\sum \alpha_{jk} \langle \mathbf{x}_i, \mathbf{x}_k \rangle + \sum_{kk'} \alpha_{jk}\alpha_{jk'} \langle \mathbf{x}_k, \mathbf{x}_{k'} \rangle$ if data are characterized by similarities and $\|\mathbf{x}_i - \mathbf{w}_j\|^2 = \sum_k \alpha_{jk}\|\mathbf{x}_i - \mathbf{x}_k\|^2 - 0.5 \cdot \sum_{kk'} \alpha_{jk}\alpha_{jk'}\|\mathbf{x}_k - \mathbf{x}_{k'}\|^2$ if data are characterized by dissimilarities. These formulas can directly be plugged into the winner takes all computation of the mapping, only referring to the coefficients α_j and the known similarities or dissimilarities. Similarly, training is possible optimizing the coefficients α_{jk}. A solid mathematical treatment of these approaches is possible based on an embedding of data into pseudo-Euclidean space, as discussed e.g. in [10].

Thus, both, GTM and SOM can also be used in settings where complex data with dedicated similarity measures are dealt with, such as biological sequence data and alignment distances, biological networks and graph comparisons, scientific texts or textual experiment descriptions compared based on the normalized compression distance, functional data such as mass spectra and functional metrics, data incorporating temporal dependencies such as EEG and dynamic time warping, or other abstract data types such as strings, trees, graphs which are compared using corresponding structure kernels.

However, at this point, a problem occurs: while the approaches yield explicit out-of-sample extensions, their training complexity is quadratic in the number of samples since training depends on a quadratic similarity or dissimilarity matrix rather than vectorial descriptions. This makes parametric models for (dis-)similarities infeasible if large data sets are dealt with. A couple of different approximation schemes have been proposed at this point, which make use of the specific form of the involved mapping:

1. Training on a subsample only, and subsequent mapping of all data points;
2. training using the Nyström approximation;
3. patch processing of the data.

(1) is based on out-of-sample extensions by means of an explicit mapping. It relies on the assumption that training data are representative; a reliable mapping of data points which follow a different distribution than the training set cannot be guaranteed. This restriction is partially overcome by the other approaches.

In (2) the Nyström approximation of a similarity or dissimilarity matrix D is used, which samples D based on a subset of M landmarks using the approximation $D \approx D^t_{M,m} D^{-1}_{M,M} D_{M,m}$ where the subscripts refer to the rows/columns of

D corresponding to the set of landmarks by M and the full data set by m. The superscript -1 refers to the Moore-Penrose pseudoinverse. For GTM or SOM, this approximation can be integrated into the algorithms such that the result is linear in the size of the training set, but cubic in the number of landmarks [6,9]. What is the benefit of this approximation as compared to a direct sampling as proposed in (1)? Unlike the latter, the Nyström approximation yields reasonable results if the dissimilarity matrix is sampled sufficiently, a sufficient condition for an exact realization being connected to the rank, for example. Thus this approximation requires only that the intrinsic vector space is spanned sufficiently.

Method (3) relies on an even weaker assumption, taking all information implicitly into account while training. Here, data are processed in patches of fixed size, training a model on a fixed patch one after the other [9,10]. Assuming fixed patch size, this step is constant time. It is necessary to transfer the information obtained in one patch to the next one, such that information already extracted from the data is kept. This can be done relying on the specific nature of the model: SOM and GTM are based on prototypes which represent all already seen data. Thus, every patch is enriched by the already trained prototypes as additional data points, counted according to the multiplicity of their receptive field. Since prototypes are given only implicitly in case of (dis-)similarity data, they are here approximated by their k nearest exemplars. Iterating this processing, a model which can also deal with data with a severe trend results [10,9].

Nonparametric Approaches

Nonparametric methods often take a dual approach: the data points $\mathbf{x}_i$ contained in a high dimensional vector space constitute the starting point; for every point coefficients $\mathbf{y}_i$ are determined in Y such that the characteristics of these points mimics the characteristics of their high-dimensional counterpart.

We consider t-SNE in more detail, since it demonstrates the strengths and weaknesses of this principle in an exemplary way. Probabilities in the original space are defined as $p_{ij} = (p_{(i|j)} + p_{(j|i)})/(2m)$ where $p_{j|i} = (\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2))/(\sum_{k\neq i}\exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2))$ depends on the pairwise distances of points; σ_i is automatically determined by the method such that the effective number of neighbors coincides with a priorly specified parameter, the perplexity. In the projection space, probabilities are induced by the student-t distribution $q_{ij} = (1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1})/(\sum_{k\neq l}(1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1})$ to avoid the crowding problem by using a long tail distribution. The goal is to find projections $\mathbf{y}_i$ such that the difference between p_{ij} and q_{ij} becomes small as measured by the Kullback-Leibler divergence. t-SNE relies on a gradient based technique.

Many alternative non-parametric techniques proposed in the literature have a very similar structure, as pointed out in [4]: They extract a characteristic of the data points $\mathbf{x}_i$ and try to find projections $\mathbf{y}_i$ such that the corresponding characteristics is as close as possible as measured by some cost function. [4] summarizes some of today's most popular dimensionality reduction methods this way. These techniques do not rely on a parametric form such that they display a rich flexibility to emphasize local nonlinear structures. This makes them much more flexible as compared to linear approaches such as PCA, and it can also give fundamentally different results as compared to GTM or SOM, which are constrained to inherently smooth mappings. This flexibility is payed for by two drawbacks, which make the techniques unsuited for large data sets:

1. The techniques do not provide direct out-of-sample extensions,
2. the techniques display at least quadratic complexity.

Thus, these techniques are not suited for large data sets in its direct form.

Which of the above mentioned approximation techniques can be transferred to non-parametric approaches? Patch processing requires a compression of already seen information in terms of few representatives such as prototypes. Non-parametric approaches do not provide such information. The Nyström approximation relies on the fact that (dis-)similarities are used in terms of matrix operations only, but not referred to individually by means of non-linear operations. This is not the case for nonlinear non-parametric techniques. Hence, approximation possibilities (2) and (3) are ruled out for non-parametric methods.

Method (1), direct subsampling, relies on out-of-sample extensions which is also not directly available for most non-parametric approaches. However, there has been some work in this respect for current non-parametric dimensionality reduction techniques such as e.g. specific methods for spectral methods [1], or a principled approach how to transfer non-parametric approaches into parametric ones [4]. Here, we consider one particularly flexible technique.

Kernel t-SNE

How to extend a non-parametric dimensionality reduction technique such as t-SNE to an explicit mapping? We fix a parametric form $\mathbf{x} \to f_w(\mathbf{x}) = \mathbf{y}$ and optimize the parameters of f_w instead of the projection coordinates. Thereby, it is critical to choose a mapping with sufficient local flexibility to capture local nonlinear structures. In kernel t-SNE, the following form is used [8]:

$$\mathbf{x} \mapsto \mathbf{y}(\mathbf{x}) = \sum_j \alpha_j \cdot \frac{k(\mathbf{x}, \mathbf{x}_j)}{\sum_l k(\mathbf{x}, \mathbf{x}_l)}$$

where $\alpha_j \in Y$ are parameters corresponding to points in the projection space and the data $\mathbf{x}_j$ are taken as a fixed sample. k is the Gaussian kernel parameterized by the bandwidth, as usual. In the limit of small bandwidth, original t-SNE is resembled for the points $\mathbf{x}_i$, α_j corresponding to their projections $\mathbf{y}_j$.

There exist different possibilities to train the parameters α_j of this mapping, several different ones having been compared in [8]. Due to its form as a generalized linear mapping, one very simple and particularly efficient training procedure is possible, which we will use in the following: first a set of example points $\mathbf{x}_i$ and its projections $\mathbf{y}_i$ are determined using standard t-SNE, forming a training set T. The set of support vectors $\mathbf{x}_j$ for the kernel mapping is taken as a subset. Then the parameters α_j are analytically determined as the least squares solution of this

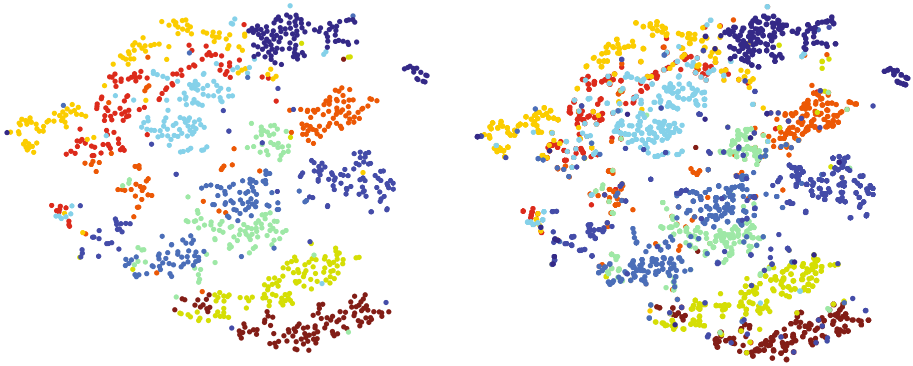


Fig. 1. Kernel t-SNE (left) for 10 % of the USPS data set and out of sample extension for the full data set (right)

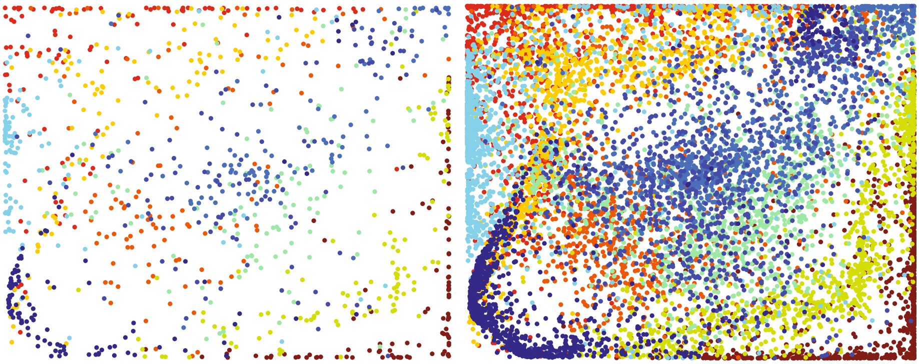


Fig. 2. GTM (left) for 10 % of the USPS data set and out of sample extension for the full data set (right)

training set T. The matrix $\mathbf{A}$ of parameters α_j is explicitly given as $\mathbf{A} = \mathbf{Y} \cdot \mathbf{K}^{-1}$ where $\mathbf{K}$ is the normalized Gram matrix with entries $k(\mathbf{x}_i, \mathbf{x}_j)/\sum_j k(\mathbf{x}_i, \mathbf{x}_j)$. $\mathbf{Y}$ denotes the matrix of projections $\mathbf{y}_i$, and $\mathbf{K}^{-1}$ refers to the pseudo-inverse. The bandwidth can be optimized based on cross-validation.

Assuming a fixed size training set and set of support vectors is used, this method displays linear complexity to map large data sets. Note that it is not restricted to vectors only, rather, any distance measure can be included in the Gaussian kernels. One example of this procedure is shown in Fig.1. In contract, GTM and its out-of-sample extension are displayed in Fig. 2. In both cases, the USPS data set with 1.000 points for the training set and out-of-sample extension to 11.000 points is displayed. Coloring corresponds to the underlying ten classes which represent different handwritten digits. Obviously, both, GTM and t-SNE capture parts of the nonlinear structure underlying the data, but neither method is capable of displaying clear class structures.

3 Discriminative Dimensionality Reduction

Kernel t-SNE enables to map large data sets in linear time by training a mapping on a small subsample only, yielding acceptable results. However, it is often the case that the underlying data structure such as cluster formation is not yet as pronounced based on a small subset only as it would be for the full data set. How can this information gap be closed?

It has been proposed in [11,14,19] to enrich nonlinear dimensionality reduction techniques such as the self-organizing map by auxiliary information in order to enforce the method to display the information which is believed as relevant by an applicant. A particularly intuitive situation is present if data are enriched by accompanying class labels, and the information most relevant for the given classification at hand should be displayed.

Formally, we assume that every data point $\mathbf{x}_i$ is equipped with a class label c_i. Projection points $\mathbf{y}_i$ should be found such that the aspects of $\mathbf{x}_i$ which are relevant for c_i are displayed.

Fisher Kernel t-SNE

Formally, this auxiliary information can be easily integrated into a projection technique by referring to the Fisher information, as recently introduced e.g. in [7] in the context of t-SNE. We consider the Riemannian manifold spanned by the data points $\mathbf{x}_i$ and corresponding tangent spaces equipped with the quadratic form $d_1(\mathbf{x}_i, \mathbf{x}_i + d\mathbf{x}) = (d\mathbf{x})^T \mathbf{J}(\mathbf{x}_i)(d\mathbf{x})$ where $\mathbf{J}(\mathbf{x})$ denotes the Fisher information matrix $\mathbf{J}(\mathbf{x}) = E_{p(c|\mathbf{x})} \left\{ \left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x}) \right) \left(\frac{\partial}{\partial \mathbf{x}} \log p(c|\mathbf{x}) \right)^T \right\}$. A Riemannian metric is induced by minimum path integrals using this quadratic form locally. We refer to this metric as the Fisher metric in the following. This metric measures distances between data points $\mathbf{x}_i$ and $\mathbf{x}_j$ along the manifold, thereby locally transforming the space according to its relevance for the given label information. This auxiliary information can easily be integrated into t-SNE or any other dimensionality reduction technique which relies on distances by simply substituting the Euclidean metric by the Fisher metric.

In practice, the Fisher metric has to be estimated based on the given data only. The conditional probabilities $p(c|\mathbf{x})$ can be estimated from the data using a Parzen nonparametric estimator, for example. The exact formulas for the resulting Fisher matrix estimation as well as different ways to approximate and optimize the resulting path integrals have been discussed in [14], for example.

In [7], it has been proposed to integrate this Fisher information into kernel t-SNE by means of a corresponding kernel. Here, we take an even simpler perspective: we consider a set of data points $\mathbf{x}_i$ which are equipped with the pairwise Fisher metric which is estimated based on their class labels taking simple linear approximations for the path integrals. Using this set, a training set T is obtained with t-SNE which takes the auxiliary label information into account. For this set, we infer a kernel t-SNE mapping as before, which is adapted to the label information due to the information inherent in the training set. Fig. 3 shows one

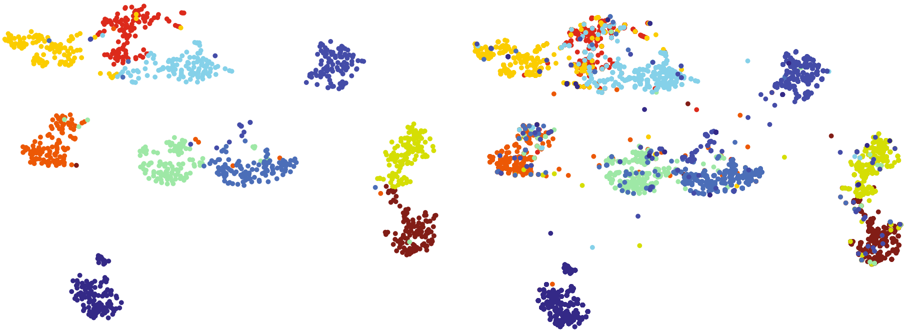


Fig. 3. Fisher kernel t-SNE (left) for 10 % of the USPS data set and out of sample extension for the full data set (right)

example of this procedure for the USPS data set where 1.000 random points are used for training. Obviously, a much clearer class structure as compared to kernel t-SNE is obtained. Note that the estimation of the Fisher metric is necessary for the training set only, such that, again, a linear time technique results.

Supervised GTM

For parameterized approaches such as GTM, there exists the possibility of an even more efficient integration of label information based on the explicit prototypes. Every prototype $\mathbf{w}_j$ can be equipped with a label based on the data contained in its receptive field. Then, the metric tensor can be changed locally at the prototype by integrating an adaptive matrix $M_j = \Omega_j \Omega_j^t$ at the point, which is forced as positive semi-definite via its representation. Note that, for GTM, only the distances of points to the prototypes are computed during training, such that this information is sufficient. This matrix is adapted in order to emphasize the directions which are particularly relevant for the classification induced by the prototypes. Thereby, training the metric parameters can be interleaved with a standard GTM training.

It has been investigated in [5], that a cost function as borrowed from learning vector quantization schemes is particularly suited for the adaptation of the metric. More specifically, the cost function from generalized matrix LVQ is used, which is given by $\sum_{\mathbf{x}_i}(d^+(\mathbf{x}_i) - d^-(\mathbf{x}_i))/(d^+(\mathbf{x}_i) + d^-(\mathbf{x}_i))$ where $d^+(\mathbf{x}_i) = (\mathbf{x}_i - \mathbf{w}_j)^t M_j (\mathbf{x}_i - \mathbf{w}_j)$ denotes the squared distance to the closest prototype $\mathbf{w}_j$ with the same label as $\mathbf{x}_i$ and $d^-(\mathbf{x}_i)$ refers to the corresponding distance to a prototype with different label. A gradient technique allows to adapt the metric parameters this way.

This procedure does not rely on the Fisher metric, rather, a locally discriminative metric tensor is adapted around every prototype to arrive at a maximum margin in its receptive field [15]. Note that the metric adaptation can be done at different levels of granularity, taking full local matrices, global matrices, or even matrices restricted to a diagonal form only, depending on the required flexibility.

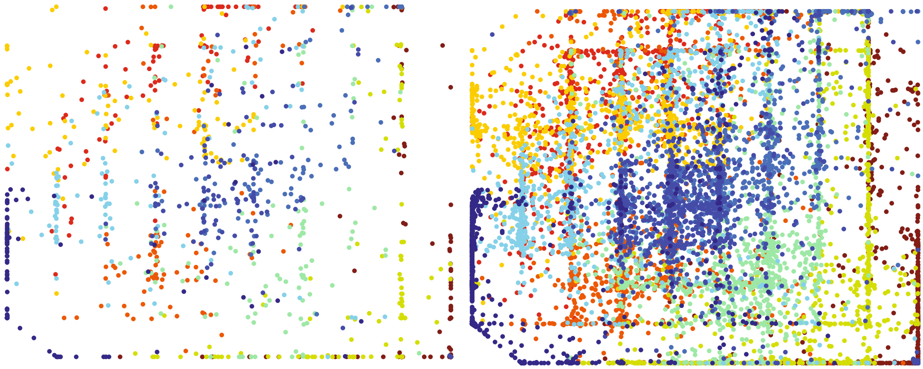


Fig. 4. GTM with local metric adaptation (left) for 10 % of the USPS data set and out of sample extension for the full data set (right)

Fig. 4 displays the result of a GTM trained with locally adapted matrices for a subset of 1.000 points of the USPS data set and its extension to all points. Obviously, a clearer separation of clusters can be achieved this way, albeit class boundaries are less pronounced as compared to Fisher kernel t-SNE.

Visualization of Classifiers

One very interesting application of supervised dimensionality reduction for large data sets is the subject of ongoing work: visualization of classifiers. While visualization of decision boundaries of classifiers is frequently used for low dimensional settings, there does not yet exist an accepted strategy if high dimensional data are dealt with. Discriminative dimensionality reduction as well as a technique similar to kernel t-SNE can offer such a framework.

Assume a classifier such as SVM is given, which maps data $\mathbf{x}_i$ to class labels. In addition, we assume that a real value $r(\mathbf{x}_i)$ is available indicating the distance from the decision boundary or a nonlinear transformation thereof. Now the principled idea is as follows:

1. project the points $\mathbf{x}_i$ to low dimensions $\mathbf{y}_i$ using discriminative Fisher t-SNE based on the class labels c_i,
2. train an inverse mapping $p : Y \to X$ which maps projections $\mathbf{y}_i$ back to the data $\mathbf{x}_i$ similar to kernel t-SNE, thereby taking the label information into account,
3. sample the projection space in the span of these projections $\mathbf{y}_i$ leading to points $\mathbf{z}_i$ and determine its inverse points $\mathbf{a}_i = p(\mathbf{z}_i)$,
4. visualize the projections of training points $\mathbf{y}_i$ together with the contours induced by $\mathbf{z}_i$ with real value $r(\mathbf{a}_i)$ which approximate the decision boundaries.

In this procedure, relying on discriminative visualization, several problems are avoided: the projection focusses on the directions relevant for the class labeling, hence relevant for the classifier. Further, sampling of the class boundary takes

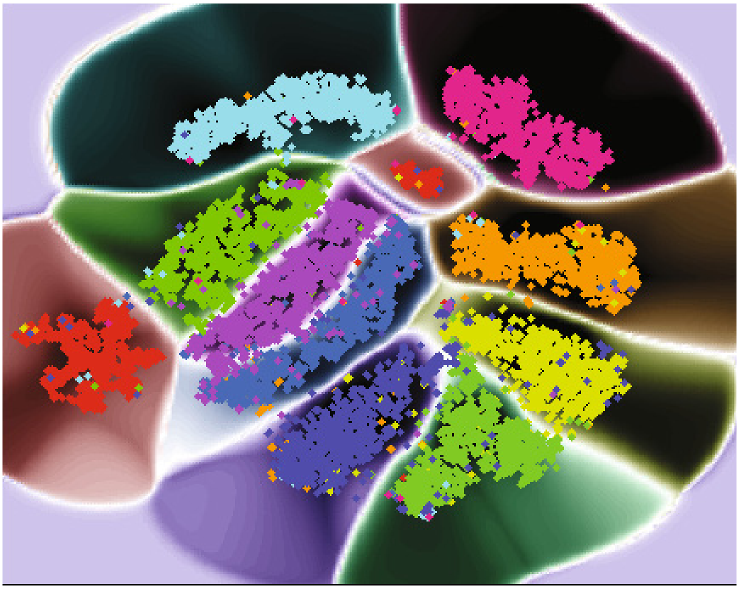


Fig. 5. Visualization of SVM decision boundaries trained for the USPS data set

place in the projection space, avoiding the curse of dimensionality and again, focussing on the directions relevant for the classification. An example of this procedure for an SVM trained for the USPS data set is shown in Fig. 5. This way, it is possible to directly spot relevant characteristics of the classifier such as regions of errors, complex decision boundaries, or the modality of the classes.

4 Discussion

We have discussed recent developments in dimensionality reduction techniques which make the techniques suitable for large data sets. For parametric methods such as SOM or GTM, the inherent availability of out-of-sample extensions and linear complexity allow a direct application also for large size data, making additional approximations necessary only if non-vectorial data are dealt with. In contrast, non-parametric techniques usually scale at least quadratically with the size of the training set such that already a few thousand points put these techniques at the limits of current desk computers. We have discussed linear time approximations based on subsampling only, which require out-of-sample extensions such as as provided by kernel t-SNE, for example. Another focus lies on possibilities to account for the information loss in such cases due to the limited data size, which can be matched by integrating auxiliary information. This way, very promising methods which can be used also for large data sets result.

Acknowledgement. This work has been supported by the DFG under the grants HA 2719/7-1 and HA 2719/4-1 and by the CITEC center of excellence.

References

1. Bengio, Y., Paiement, J.-F., Vincent, P., Delalleau, O., Roux, N.L., Ouimet, M.: Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. In: Advances in Neural Information Processing Systems, pp. 177–184. MIT Press (2004)
2. Biehl, M., Hammer, B., Merényi, E., Sperduti, A., Villmann, T.: Learning in the context of very high dimensional data (Dagstuhl Seminar 11341), vol. 1 (2011)
3. Bishop, C.M., Svensén, M., Williams, C.K.I.: Gtm: The generative topographic mapping. Neural Computation 10, 215–234 (1998)
4. Bunte, K., Biehl, M., Hammer, B.: A general framework for dimensionality reducing data visualization mapping. Neural Computation 24(3), 771–804 (2012)
5. Gisbrecht, A., Hammer, B.: Relevance learning in generative topographic mapping. Neurocomputing 74(9), 1359–1371 (2011)
6. Gisbrecht, A., Hammer, B., Schleif, F.-M., Zhu, X.: Accelerating dissimilarity clustering for biomedical data analysis. In: IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, pp. 154–161 (2011)
7. Gisbrecht, A., Hofmann, D., Hammer, B.: Discriminative Dimensionality Reduction Mappings. In: Hollmén, J., Klawonn, F., Tucker, A. (eds.) IDA 2012. LNCS, vol. 7619, pp. 126–138. Springer, Heidelberg (2012)
8. Gisbrecht, A., Lueks, W., Mokbel, B., Hammer, B.: Out-of-sample kernel extensions for nonparametric dimensionality reduction. In: ESANN 2012, pp. 531–536 (2012)
9. Hammer, B., Gisbrecht, A., Hasenfuss, A., Mokbel, B., Schleif, F.-M., Zhu, X.: Topographic Mapping of Dissimilarity Data. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 1–15. Springer, Heidelberg (2011)
10. Hammer, B., Hasenfuss, A.: Topographic mapping of large dissimilarity datasets. Neural Computation 22(9), 2229–2284 (2010)
11. Kaski, S., Sinkkonen, J., Peltonen, J.: Bankruptcy analysis with self-organizing maps in learning metrics. IEEE Transactions on Neural Networks 12, 936–947 (2001)
12. Kohonen, T.: Self-Organizing Maps. Springer (2000)
13. Lee, J.A., Verleysen, M.: Nonlinear dimensionality redcution. Springer (2007)
14. Peltonen, J., Klami, A., Kaski, S.: Improved learning of riemannian metrics for exploratory analysis. Neural Networks 17, 1087–1100 (2004)
15. Schneider, P., Biehl, M., Hammer, B.: Adaptive relevance matrices in learning vector quantization. Neural Computation 21, 3532–3561 (2009)
16. Tipping, M.E., Bishop, C.M.: Probabilistic principal component analysis. Journal of the Royal Statistical Society, Series B 61, 611–622 (1999)
17. van der Maaten, L., Hinton, G.: Visualizing high-dimensional data using t-sne. Journal of Machine Learning Research 9, 2579–2605 (2008)
18. van der Maaten, L., Postma, E., van den Herik, H.: Dimensionality reduction: A comparative review. Technical report, Tilburg University Technical Report, TiCC-TR 2009-005 (2009)
19. Venna, J., Peltonen, J., Nybo, K., Aidos, H., Kaski, S.: Information retrieval perspective to nonlinear dimensionality reduction for data visualization. Journal of Machine Learning Research 11, 451–490 (2010)
20. Ward, M., Grinstein, G., Keim, D.A.: Interactive Data Visualization: Foundations, Techniques, and Application. A. K. Peters, Ltd. (2010)
21. Yin, H.: On the equivalence between kernel self-organising maps and self-organising mixture density networks. Neural Networks 19(6-7), 780–784 (2006)

On-Line Relational SOM for Dissimilarity Data

Madalina Olteanu, Nathalie Villa-Vialaneix, and Marie Cottrell

SAMM-Université Paris 1 Panthéon Sorbonne
90, rue de Tolbiac, 75013 Paris, France
{madalina.olteanu,nathalie.villa,marie.cottrell}@univ-paris1.fr
http://samm.univ-paris1.fr

Abstract. In some applications and in order to address real world situations better, data may be more complex than simple vectors. In some examples, they can be known through their pairwise dissimilarities only. Several variants of the Self Organizing Map algorithm were introduced to generalize the original algorithm to this framework. Whereas median SOM is based on a rough representation of the prototypes, relational SOM allows representing these prototypes by a virtual combination of all elements in the data set. However, this latter approach suffers from two main drawbacks. First, its complexity can be large. Second, only a batch version of this algorithm has been studied so far and it often provides results having a bad topographic organization. In this article, an on-line version of relational SOM is described and justified. The algorithm is tested on several datasets, including categorical data and graphs, and compared with the batch version and with other SOM algorithms for non vector data.

1 Introduction

In many real-world applications, data cannot be described by a fixed set of numerical attributes. This is the case, for instance, when data are described by categorical variables or by relations between objects (i.e., persons involved in a social network). A common solution to address this kind of issue is to use a measure of resemblance (i.e., a similarity or a dissimilarity) that can handle categorical variables, graphs or focus on specific aspects of the data, designed by expertise knowledge. Many standard methods for data mining have been generalized to non vectorial data, recently including prototype-based clustering. The recent paper [6] provides an overview of several methods that have been proposed to tackle complex data with neural networks.

In particular, several extensions of the Self-Organizing Maps (SOM) algorithm have been proposed. One approach consists in extending SOM to categorical data by using a method similar to Multiple Correspondence Analysis, [5]. Another approach uses the median principle which consists in replacing the standard computation of the prototypes by an approximation in the original dataset. This principle was used to extend SOM to dissimilarity data in [16]. One of the main drawbacks of this approach is that forcing the prototypes to be chosen among the dataset is very restrictive; in order to increase the flexibility of the

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 13–22.
DOI: 10.1007/978-3-642-35230-0_2

representation, [3] propose to represent a class by several prototypes, all chosen among the original dataset. However this method increases the computational time and prototypes still stay restricted to the original dataset, hence reflecting possible sampling or sparsity issues.

An alternative to median-based algorithms relies on a method that is close to the classical algorithm used in the Euclidean case and is based on the idea that prototypes may be expressed as linear combinations of the original dataset. In the kernel SOM framework, this setting is made natural by the use of the kernel that maps the original data into a (large dimensional) Euclidean space (see [17,1] for on-line versions and [2] for the batch version). Many kernels have been designed to handle complex data such as strings, nodes in a graphs or graphs themselves [10].

More generally, when the data are already described by a dissimilarity that is not associated to a kernel, [12,19,11] use a similar idea. They introduce an implicit "convex combination" of the original data to extend the classical batch versions of SOM to dissimilarity data. This approach is known under the name "relational SOM". The purpose of the present paper is to show that the same idea can be used to define on-line relational SOM. Such an approach reduces the computational cost of the algorithm and leads to a better organization of the map. In the remaining of this article, Section 2 describes the methodology and Section 3 illustrates its use on simulated and real-world data.

2 Methodology

In the following, let us suppose that n input data, $x_1, \ldots, x_n$, from an arbitrary input space $\mathcal{G}$ are given. These data are described by a dissimilarity matrix $\mathbf{D} = (\delta_{ij})_{i,j=1,\ldots,n}$ such that D is non negative ($\delta_{ij} \geq 0$), symmetric ($\delta_{ij} = \delta_{ji}$) and null on the diagonal ($\delta_{ii} = 0$). The purpose of the algorithm is to map these data into a low dimensional grid composed of U units which are linked together by a neighborhood relationship $K(u, u')$. A prototype p_u is associated with each unit $u \in \{1, \ldots, U\}$ in the grid. The U prototypes $(p_1, p_2, \ldots, p_U)$ are initialized either randomly among the input data or as random convex combinations of the input data.

In the Euclidean framework, where the input space is equipped with a distance, the matrix D is the distance matrix with entries $\delta_{ij} = \|x_i - x_j\|^2$. In this case, the on-line SOM algorithm iterates

- an *assignment step*: a randomly chosen input x_i is assigned to the closest prototype denoted by $p_{f(x_i)}$ according to shortest distance rule

$$f(x_i) = \arg\min_{u=1,\ldots,U} \|x_i - p_u\|,$$

- a *representation step*: all prototypes are updated

$$p_u^{\text{new}} = p_u^{\text{old}} + \alpha K(f(x_i), u)\,(x_i - p_u)\,,$$

where α is the training parameter.

In the more general framework, where the data are known through pairwise distances only, the assignment step cannot be carried out straightforwardly since the distances between the input data and the prototypes may not be directly computable. The solution introduced in [19] consists in supposing that prototypes are convex combinations of the original data, $p_u = \sum_i \beta_{ui} x_i$ with $\beta_{ui} > 0$ and $\sum_i \beta_{ui} = 1$. If β_u denotes the vector $(\beta_{u1}, \beta_{u2}, \ldots, \beta_{un})$, the distances in the assignment step can be written in terms of D and β_u only:

$$\|x_i - p_u\|^2 = (D\beta_u)_i - \frac{1}{2}\beta_u^T D\beta_u.$$

According to [19], the equation above still holds if the matrix D is no longer a distance matrix, but a general dissimilarity matrix, as long as it is symmetric and null on the diagonal. A generalization of the batch SOM algorithm, called batch relational SOM, which holds for dissimilarity matrices is introduced in [19].

The representation step may also be carried out in this general framework as long as the prototypes are supposed to be convex combinations of the input data. Hence, using the same ideas as [19], we introduce the on-line relational SOM, which generalizes the on-line SOM to dissimilarity data. The proposed algorithm is the following:

Algorithm 1. On-line relational SOM

```
For all $u = 1, \ldots, U$ and $i = 1, \ldots, n$, initialize $\beta^0_{ui}$ randomly in $\mathbb{R}$, such that $\beta^0_{ui} \geq 0$
and $\sum_i^n \beta^0_{ui} = 1$.
for t=1,...,T do
    Randomly chose an input $x_i$
    Assignment : find the unit of the closest prototype
        $f^t(x_i) \leftarrow \arg\min_{u=1,\ldots,U} \left(\beta_u^{t-1}\mathbf{D}\right)_i - \frac{1}{2}\beta_u^{t-1}\mathbf{D}(\beta_u^{t-1})^T$
    Update of the prototypes: $\forall\, u = 1, \ldots, U$,
        $\beta_u^t \leftarrow \beta_u^{t-1} + \alpha^t K^t(f^t(x_i), u)\left(\mathbf{1}_i - \beta_u^{t-1}\right)$
    where $\mathbf{1}_i$ is a vector with a single non null coefficient at the ith position, equal to
    one.
end for
```

In the applications of Section 3, the parameters of the algorithm are chosen according to [4]: the neighborhood K^t decreases in a piecewise linear way, starting from a neighborhood which corresponds to the whole grid up to a neighborhood restricted to the neuron itself; α^t vanishes at the rate of $1/t$. Let us remark that if the dissimilarity matrix is a Euclidean distance matrix, relational on-line SOM is equivalent to the classical on-line SOM algorithm, as long as the n input data contain a basis of the input space $\mathcal{G}$.

As explained in [8], although batch SOM possesses the nice properties of being deterministic and of usually converging in a few iterations, it has several drawbacks such as bad organization, bad visualization, unbalanced classes and strong dependence on the initialization. Moreover, the computational complexity of the online algorithm may be significantly reduced with respect to the batch algorithm. With a naive implementation and for one iteration, the complexity of the batch algorithm is $\mathcal{O}(Un^3 + Un^2)$, while for the online algorithm it is $\mathcal{O}(Un^2 + Un)$. However, since the online algorithm has to scan all input data, the number of iterations has to be significantly larger than in the batch case. To summarize, if T_1 is the number of iterations for batch relational SOM and T_2 is the number of iterations for online relational SOM, the ratio between the two computation times will be $T_1 n / T_2$. For a more efficient implementation of the batch algorithm, the reader may refer to [13].

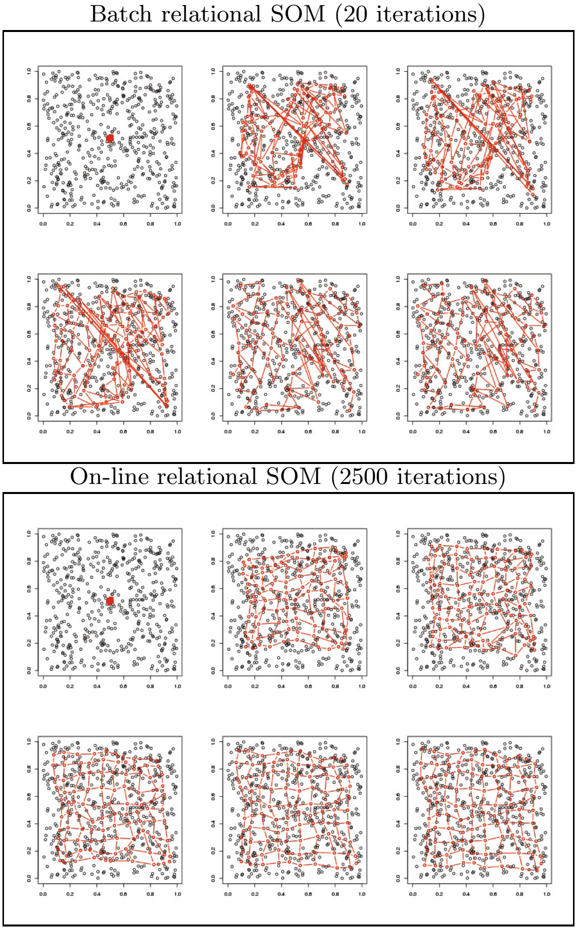
For illustration, let us consider 500 points sampled randomly from the uniform distribution in $[0, 1]^2$. The batch version of relational SOM and the on-line version of relational SOM were performed with identical 10x10 grid structures and identical initializations. Results are available in Figure 1. Batch relational SOM converged quickly, in 20 iterations (the grid organization is represented at iterations 0 (random initialization), 5, 9, 13, 17 and 20), but the map is not well organized. On-line relational SOM converged in less than 2500 iterations (the grid organization is represented at iterations 0 (initialization), 500, 1000, 1500, 2000 and 2500), but the map is now almost perfectly organized. This results was achieved in 40 minutes for the batch version and in 10 minutes for the on-line version on a netpc (with 2 × 1GHz AMD processors and 4Go RAM).

3 Applications

This section presents several applications of the on-line relational SOM on various datasets. Section 3.1 deals with simulated data described by numerical variables, but organized on a non linear surface. Section 3.2 is an application on a real dataset where the individuals are described by categorical variables. Finally, Section 3.3 is an application to the clustering of nodes of a graph.

3.1 Swiss Roll

Let us first use a toy example to illustrate the stochastic version of relational SOM. The simulated data is the popular Swiss roll, a two-dimensional manifold embedded in a three-dimensional space. This example has already been used for illustrating the performances of Isomap [21]. The data has the shape illustrated by Figure 2. 5 000 points were simulated. However, since all methods presented here work with matrices of pairwise distances, the computation times would have been rather heavy for 5 000 points. Hence, we run the different algorithms on 1 000 points uniformly distributed on the manifold. First, the distance matrix was computed using the geodesic distance based on the K-rule with $K = 10$. Then, two types of algorithms were performed: multidimensional scaling and


Batch relational SOM (20 iterations)

On-line relational SOM (2500 iterations)

Fig. 1. Batch and on-line SOM organization for 500 samples from the uniform distribution in $[0, 1]^2$. The same initialization was used for both algorithms.

self-organizing maps. The results obtained with Isomap [21] are available in Figure 2. As expected, both methods succeed in unfolding the Swiss roll and the results are very similar. Next, batch median SOM and on-line relational SOM were applied to the dissimilarity matrix computed with the geodesic distance. As shown in Figure 3, the size of the map plays an important role in unfolding the data. For squared grids, the problem is not completely solved by either of the two algorithms. Nevertheless, on-line relational SOM manages to project the different scrolls of the roll into separate regions on the map. Moreover, some empty cells highlight the roll structure, which is not completely unfolded but rather projected without overlapping. Since squared grids appeared too heavily constrained, we also tested rectangular grids. The results are better for both algorithms which both manage to unfold the data. However, the on-line version clearly outperforms the batch version.

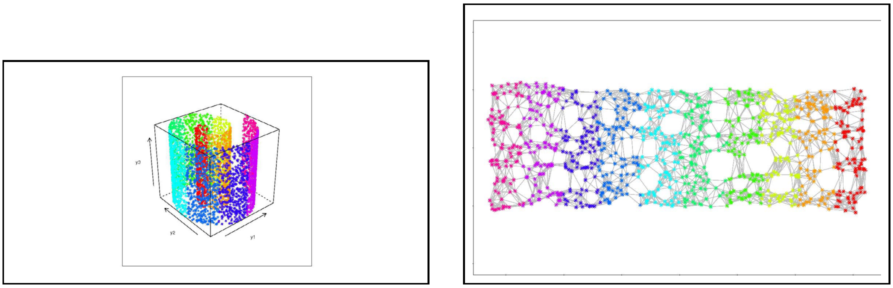
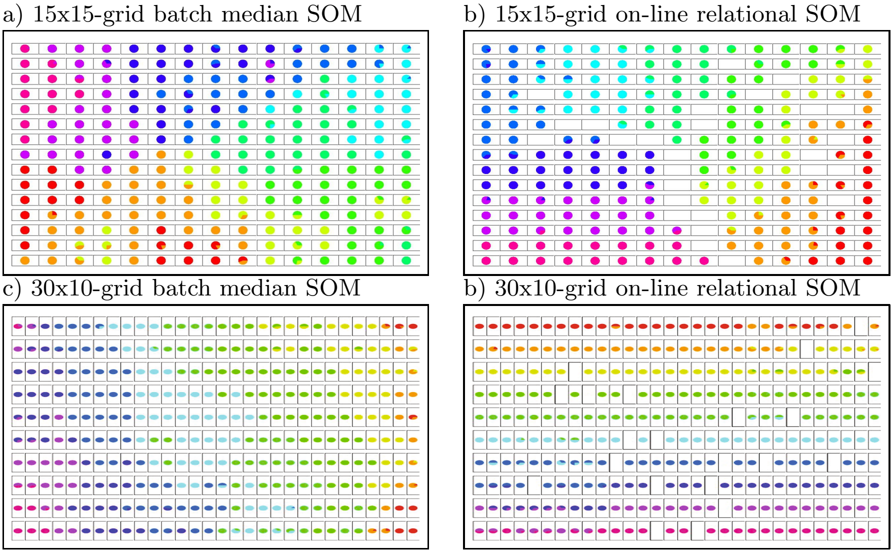


Fig. 2. Unfolding the Swiss roll using Isomap


a) 15x15-grid batch median SOM

b) 15x15-grid on-line relational SOM

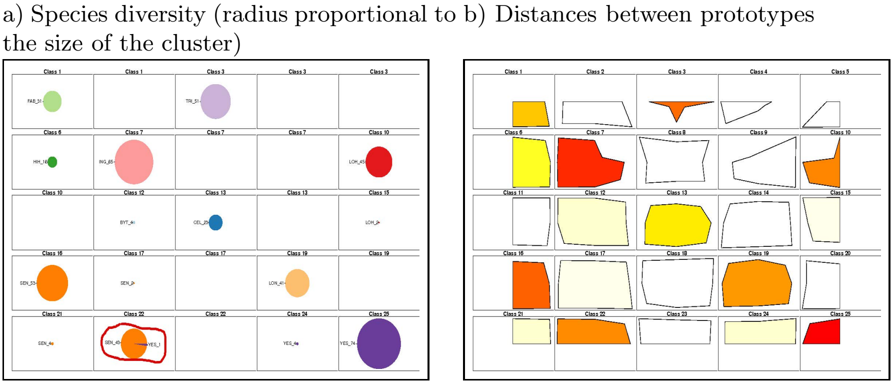
c) 30x10-grid batch median SOM

b) 30x10-grid on-line relational SOM

Fig. 3. Unfolding the Swiss roll using self-organizing maps

3.2 Amazonian Butterflies

This data set contains 465 input data and was previously used by [14] to demonstrate the synergy between DNA barcoding and morphological-diversity studies. The notion of DNA barcoding comprises a wide family of molecular and bioinformatics methods aimed at identifying biological specimens and assigning them to a species. According to the vast literature published during the past years on the topic, two separate tasks emerge for DNA barcoding: on the one hand, assign unknown samples to known species and, on the other hand, discover undescribed species, [7]. The second task is usually approached with the Neighbor Joining algorithm [20] which constructs a tree similar to a dendrogram. When the sample size is large, the trees become rapidly unreadable. Moreover, they are quite sensitive to the order in which the input data are presented. Let us



a) Species diversity (radius proportional to the size of the cluster)

b) Distances between prototypes

Fig. 4. On-line relational SOM for Amazonian butterflies

also mention that unsupervised learning and visualization methods are used to a very limited extent by the DNA barcoding community, although the information they bring may be quite useful. The use of self-organizing maps may be quite helpful in visualizing the data and bringing out clusters or groups of clusters that may correspond to undescribed species.

DNA barcoding data are composed of sequences of nucleotides, i.e. sequences of "a", "c", "g", "t" letters in high dimension (hundreds or thousands of sites). Specific distances and dissimilarities such as the Kimura-2P ([15]) are usually computed. Hence, since the data is not Euclidean, dissimilarity-based methods appear to be more appropriate. Recently, batch median SOM was tested in [18] on several data sets, amongst which the Amazonian butterflies. Although median SOM provided encouraging results, two main drawbacks emerged. First, since the algorithm was run in batch, the organization of the map was generally poor and highly depending on the initialization. Second, since the algorithm calculates a prototype for each cluster among the dataset, it does not allow for empty clusters. Thus, the existence of species or groups of species was difficult to acknowledge. The use of on-line relational SOM overcomes these two issues. As shown in Figure 4, clusters are generally not mixing species, while the empty cells allow detecting the main groups of species. The only mixing class corresponds to a labeling error. Unsupervised clustering may thus be useful in addressing misidentification issues. In Figure 4b, distances with respect to the nearest neighbors were computed for each node. The distance between two nodes/cells is computed as the mean dissimilarity between the observations within each class. A polygon is drawn within each cell with vertices proportional to the distances to its neighbors. If two neighbor prototypes are very close, then the corresponding vertices are very close to the edges of the two cells. If the distance between neighbor prototypes is very large, then the corresponding vertices are far apart, close to the center of the cells.

3.3 Political Books

This application uses a dataset modeled by a graph having 105 nodes. The nodes are books about US politics published around the time of the 2004 presidential election and sold by the on-line bookseller Amazon.com. Edges between two nodes represent frequent co-purchasing of the two books by the same buyers. The graph contains 441 edges and all nodes are labeled according to their political orientation (conservative, liberal or neutral). The graph has been extracted by Valdis Krebs and can be downloaded at `http://www-personal.umich.edu/~mejn/netdata/polbooks.zip`.

On-line relational SOM was used to cluster the nodes of the graph, according to the length of the shortest path between two nodes, which is a standard dissimilarity measure between nodes in a graph. Figures 5 and 6 (left) provide two representations of the "political books" network: the first one is the original graph displayed with a force directed placement algorithm, which is the one described in [9] and colored according to the clusters in which the nodes are classified. The second one is a simplified representation of the graph on the grid, where each node represents a cluster. The colors in the first figure and the density of edges in the second one shows that the clustering has a good organization on the grid, according to the graph structure: groups of nodes that are densely connected are classified in the same or in close clusters whereas groups of nodes that are not connected are classified apart.

Additionally, Figure 6 provides the distribution of the node labels inside each cluster for the obtained clustering (on the right hand part of the figure). Almost all clusters contain books having the same political orientation. Clusters that contain books with multiple political orientations are in the middle of the grid and include neutral books. Hence, this clustering can give a clue on a more subtle

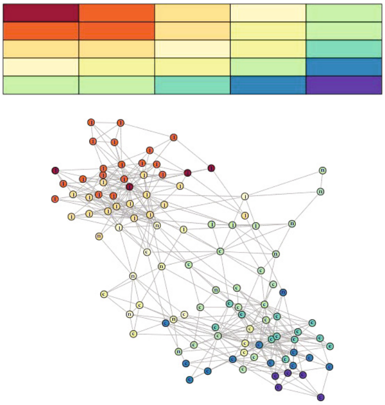


Fig. 5. "Political books" network displayed with a force directed placement algorithm. The nodes are labeled according to their political orientation and are colored according to a gradient that aims at emphasizing the distance between clusters on the grid, as represented at the top the figure.

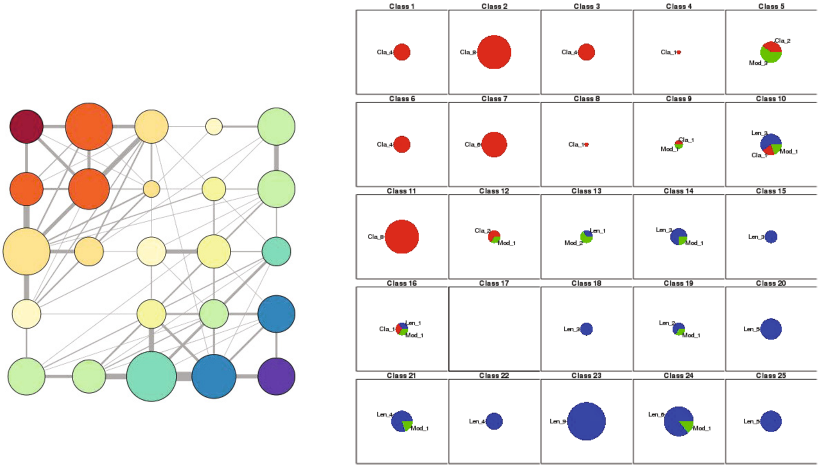


Fig. 6. Left: Simplified representation of the graph on the grid: each node represents a cluster whose area is proportional to the number of nodes included in it and the edges width represents the number of edges between the nodes of the corresponding cluster. Right: Distribution of the node labels for each neuron of the grid for the clustering obtained with the dissimilarity based on the length of the shortest paths. Red is for liberal books, blue for conservative books and green for neutral books.

political orientation than the original labeling: for instance, liberal books from cluster 12 probably have a weaker commitment that those from clusters 1 or 2.

4 Conclusion

An on-line version of relational SOM is introduced in this paper. It combines the standard advantages of the stochastic version of the SOM (better organization and faster computation) with the relational SOM that is able to handle data described by a dissimilarity. The algorithm shows good performances in projecting data described either by numerical variables or by categorical variable, as well as in clustering the nodes of a graph.

References

1. Andras, P.: Kernel-Kohonen networks. International Journal of Neural Systems 12, 117–135 (2002)
2. Boulet, R., Jouve, B., Rossi, F., Villa, N.: Batch kernel SOM and related laplacian methods for social network analysis. Neurocomputing 71(7-9), 1257–1273 (2008)
3. Conan-Guez, B., Rossi, F., El Golli, A.: Fast algorithm and implementation of dissimilarity self-organizing maps. Neural Networks 19(6-7), 855–863 (2006)
4. Cottrell, M., Fort, J.C., Pagès, G.: Theoretical aspects of the SOM algorithm. Neurocomputing 21, 119–138 (1998)

5. Cottrell, M., Letrémy, P.: How to use the Kohonen algorithm to simultaneously analyse individuals in a survey. Neurocomputing 63, 193–207 (2005)
6. Cottrell, M., Olteanu, M., Rossi, F., Rynkiewicz, J., Villa-Vialaneix, N.: Neural networks for complex data. Künstliche Intelligenz 26(2), 1–8 (2012)
7. DeSalle, R., Egan, M., Siddal, M.: The unholy trinity: taxonomy, species delimitation and dna barcoding. Philosophical Transactions of the Royal Society B-Biological Sciences 360, 1905–1916 (2005)
8. Fort, J.C., Letremy, P., Cottrell, M.: Advantages and drawbacks of the batch kohonen algorithm. In: ESANN 2002, pp. 223–230 (2002)
9. Fruchterman, T., Reingold, B.: Graph drawing by force-directed placement. Software-Practice and Experience 21, 1129–1164 (1991)
10. Gärtner, T.: Kernel for Structured Data. World Scientific (2008)
11. Hammer, B., Gisbrecht, A., Hasenfuss, A., Mokbel, B., Schleif, F.-M., Zhu, X.: Topographic Mapping of Dissimilarity Data. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 1–15. Springer, Heidelberg (2011)
12. Hammer, B., Hasenfuss, A., Strickert, M., Rossi, F.: Topographic processing of relational data. In: Proceedings of the 6th Workshop on Self-Organizing Maps (WSOM 2007), Bielefeld, Germany (September 2007) (to be published)
13. Hammer, B., Rossi, F., Hasenfuss, A.: Accelerating relational clustering algorithms with sparse prototype representation. In: Proceedings of the 6th Workshop on Self-Organizing Maps, WSOM 2007 (2007)
14. Hebert, P.D.N., Penton, E.H., Burns, J.M., Janzen, D.H., Hallwachs, W.: Ten species in one: DNA barcoding reveals cryptic species in the neotropical skipper butterfly astraptes fulgerator. Genetic Analysis (2004)
15. Kimura, M.: A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. Journal of Molecular Evolution 16, 111–120 (1980)
16. Kohohen, T., Somervuo, P.: Self-Organizing maps of symbol strings. Neurocomputing 21, 19–30 (1998)
17. Mac Donald, D., Fyfe, C.: The kernel self organising map. In: Proceedings of 4th International Conference on Knowledge-Based Intelligence Engineering Systems and Applied Technologies, pp. 317–320 (2000)
18. Olteanu, M., Nicolas, V., Schaeffer, B., Denys, C., Kennis, J., Colyn, M., Missoup, A.D., Laredo, C.: On the use of self-organizing maps for visualizing and studying barcode data. application to two data sets (preprint submitted for publication, 2012)
19. Rossi, F., Hasenfuss, A., Hammer, B.: Accelerating relational clustering algorithms with sparse prototype representation. In: 6th International Workshop on Self-Organizing Maps (WSOM). Neuroinformatics Group. Bielefield University, Bielefield (2007)
20. Saitou, N., Nei, M.: The neighbor-joining method: a new method for reconstructing phylogenetic trees. Molecular Biology and Evolution 4(4), 406–425 (1987), http://mbe.oxfordjournals.org/content/4/4/406.abstract
21. Tenenbaum, J.B., Silva, V., Langford, J.C.: A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science 290(5500), 2319–2323 (2000)

Non-Euclidean Principal Component Analysis and Oja's Learning Rule – Theoretical Aspects

Michael Biehl[1], Marika Kästner[2], Mandy Lange[2], and Thomas Villmann[2,⋆]

[1] Johann Bernoulli Institute, Intelligent Systems Group,
University Groningen, 9700 AK Groningen, The Netherlands
[2] Computational Intelligence Group,
University of Applied Sciences Mittweida, 09648 Mittweida, Germany
villmann@hs-mittweida.de

Abstract. Principal component analysis based on Hebbian learning is originally designed for data processing in Euclidean spaces. We present in this contribution an extension of Oja's online learning approach for non-Euclidean spaces. First we review the kernel principal component approach. We show that for differentiable kernel this approach can be formulated as an online learning scheme. Hence, PCA can be explicitly carried out in the data space but now equipped with a non-Euclidean metric. Moreover, the theoretical framework can be extended to principal component learning in Banach spaces based on semi-inner products. This becomes particularly important when learning in l_p-norm spaces with $p \neq 2$ is considered. In this contribution we focus on the mathematics and theoretical justification of the approach.

1 Introduction

Principal component analysis (PCA) constructs a basis of a multi-dimensional feature space, reflecting the variability observed in a given data set. It determines the linear projection of largest variance as well as orthogonal directions which are ranked according to decreasing variance [9]. Algebraic approaches to PCA, which determine directly the eigenvectors of the empirical covariance matrix, are sensitive to outliers, frequently. Iterative PCA based on Hebbian learning offers a more robust alternative as established in the pioneering work of E. Oja [15,16]. Several modifications and improvements of the basic idea have been proposed: while, for instance, Oja's subspace algorithm determines an arbitrary basis for the span of the leading eigenvectors [15,16], Sanger presented an extension which yields the eigenvectors ordered according to their eigenvalue, i.e. the observed empirical variance of projections [17].

A number of nonlinear extensions to the concept of PCA have been proposed in the literature. Kernel Hebbian learning was established by Kim et al. [11,12] based on the general concept of kernel PCA (KPCA) and reproducing kernel Hilbert spaces (RKHS) [8,20], which offer the possibility to capture non-linear

[⋆] Corresponding author.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 23–33.
DOI: 10.1007/978-3-642-35230-0_3

data structures while applying PCA. This approach was further improved by S. Günther et al. who introduced an accelerating gain parameter [4]. Hebbian PCA for functional data using Sobolev metrics was proposed in [23]. Other approaches for iterative PCA can be found in, for instance, [6].

The aim of this paper is to unify and generalize these approaches. In particular, we will revisit KPCA under the specific aspect of *differentiable kernels*. This approach delivers new aspects relating to the topological structure of the corresponding RKHS and offers new interpretations of Hebbian KPCA. Moreover, we extend this concept to so-called reproducing kernel Banach spaces (RKBS,[25]), which assume a kernel defining a semi-inner product (SIP,[13,3]) in a reflexive Banach space. As a result, PCA can be explicitly carried out in the data space but now equipped with a non-Euclidean metric. This allows visualization of data in these space. This problem becomes important, when data classification is processed in non-Euclidean spaces, as it is of great interest for better classification accuracy [10,5,19]. In particular, prototype based learning in kernel metric spaces using differentiable kernel is of great interest, because the prototypes are adjusted in the data space, here equipped with a *differentiable kernel metric* [22]. Thus, PCA projections are needed in just this kernel metric data space for visualization.

2 Hebbian Learning of Principal Components in Finite-Dimensional Vector Spaces

In this section we discuss Hebbian learning for PCA in finite-dimensional Euclidean, Hilbert and Banach spaces, subsequently.

2.1 Hebbian Learning in the Euclidean Space - Oja's and Sanger's Rule

We consider n-dimensional data vectors $\mathbf{v} \in V \subseteq \mathbb{R}^n$. Hebbian principal component learning is carried out by a stochastic iteration according to

$$\triangle \mathbf{w} = \varepsilon \cdot O \cdot (\mathbf{v} - O \cdot \mathbf{w}) \tag{1}$$

where

$$O = \langle \mathbf{v}, \mathbf{w} \rangle \tag{2}$$

is the Euclidean inner product of the current vector $\mathbf{w}$ and a randomly selected data vector $\mathbf{v}$. The inner product (2) is frequently referred to as *Hebb-output* or *Hebb-response* in this context. The parameter $0 < \varepsilon \ll 1$ is the so-called learning rate. The update scheme (1) is known as *Oja's rule* in the literature [15,16]. Under the assumption of a slowly changing weight vector $\mathbf{w}$, the stationary state $\triangle \mathbf{w} = 0$ of Oja's rule corresponds to the eigenvalue equation

$$\mathbf{C}\mathbf{w} = \langle \mathbf{w}, \mathbf{C}\mathbf{w} \rangle \, \mathbf{w}. \tag{3}$$

The stability analysis by E. Oja shows that the adaptation process (1) converges to the eigenvector corresponding to the maximum eigenvalue of the covariance matrix $\mathbf{C} = E\left[\mathbf{v}\mathbf{v}^\top\right]$ defined by the expectation operator $E[\cdot]$ [15,16]. Moreover, this learning scheme can be seen as a stochastic gradient descent on the cost function $J(\mathbf{w}) = \mathbf{w}^\top \mathbf{C}\mathbf{w}$.

The basic scheme can be extended to learn all principal components. To this end, Sanger considered n weight vectors $\mathbf{w}_i$ with Hebbian responses $O_i = \langle \mathbf{v}, \mathbf{w}_i \rangle$ and introduced the modified adaptation rule (1)

$$\triangle \mathbf{w}_i = \varepsilon \cdot O_i \cdot \left(\mathbf{v} - \sum_{j=1}^{i} O_j \cdot \mathbf{w}_j \right). \tag{4}$$

Note that for $i = 1$ the update is equivalent to (1), Sanger's algorithm yields the eigenvectors of $\mathbf{C}$ in decreasing order with respect to the corresponding eigenvalues [17].

2.2 Hebbian Learning in Finite-Dimensional Hilbert Spaces

Obviously, the Euclidean space $\mathbb{R}^n$ is a Hilbert space. We consider data $\mathbf{v} = (v_1, \ldots, v_n)^\top$ in an n-dimensional Hilbert space $\mathbb{H}^n$ with the inner product $\langle \cdot, \cdot \rangle_{\mathbb{H}^n}$ defining the norm $\|\cdot\|_{\mathbb{H}^n}$. Because each n-dimensional Hilbert space $\mathbb{H}^n$ is isomorph to the Euclidean space $\mathbb{R}^n$ there exist an isomorphism $\Theta : \mathbb{R}^n \to \mathbb{H}^n$, and linear operators are matrices $\mathbf{A}$. Application of the operator to a vector then is defined by

$$\mathbf{A}[\mathbf{v}] = \left(\langle \mathbf{a}_1, \mathbf{v} \rangle_{\mathbb{H}^n}, \ldots, \langle \mathbf{a}_n, \mathbf{v} \rangle_{\mathbb{H}^n} \right)^\top \tag{5}$$

where $\mathbf{a}_i$ are the row vectors of $\mathbf{A}$. Hence, we can replace in the Hebb-output (2) the Euclidean inner product by the inner product $\langle \cdot, \cdot \rangle_{\mathbb{H}^n}$ of the Hilbert space. For the resulting update rule we obtain immediately the stationary condition

$$\mathbf{C}_\Theta[\mathbf{w}] = \langle \mathbf{w}, \mathbf{C}_\Theta[\mathbf{w}] \rangle_{\mathbb{H}^n} \mathbf{w} \tag{6}$$

where $\mathbf{C}_\Theta = E\left[\mathbf{v}\mathbf{v}^\top\right]$ is the covariance matrix in the Hilbert space $\mathbb{H}^n$. The stability analysis follows immediately from the isomorphism. The extension to the Sanger-algorithm is obvious.

2.3 Hebbian Learning in Finite-Dimensional Banach Spaces

Here we consider n-dimensional Banch spaces $\mathbb{B}^n$ with the norm $\|\cdot\|_{\mathbb{B}^n}$. Banach spaces have gained popularity in machine learning, recently [2,7,24,25]. Prominent n-dimensional examples are the l_p-spaces with the Minkowski-norm

$$\|\mathbf{v}\|_p = \sqrt[p]{\sum_{i=1}^{n} |v_i|^p} \tag{7}$$

for $1 < p \leq \infty$. Generally, for a norm $\|\cdot\|_{\mathbb{B}^n}$ of a Banach space, a semi-inner product (SIP) $[\cdot,\cdot]_{\mathbb{B}^n}$ exists such that $\|\mathbf{v}\|_{\mathbb{B}^n} . = \sqrt{[\mathbf{v},\mathbf{v}]_{\mathbb{B}^n}}$ is valid [13,3]. Thereby, a SIP fulfills the properties of a usual inner product except the sesqui-linearity. The SIP is unique if it is Gâteaux-differentiable [25]. For the l_p-space the (unique) SIP is

$$[\mathbf{v},\mathbf{w}]_p = \frac{1}{\|\mathbf{w}\|_p^{p-2}} \sum_{i=1}^{n} v_i \left|w_i\right|^{p-1} sgn\left(w_i\right) \tag{8}$$

where $sgn\left(x\right)$ is the sign function. The application of a linear operator $\mathbf{A}$ is now defined via the SIP as

$$\mathbf{A}\left[\mathbf{v}\right] = \left(\left[\mathbf{a}_1,\mathbf{v}\right]_{\mathbb{B}^n}, \ldots, \left[\mathbf{a}_n,\mathbf{v}\right]_{\mathbb{B}^n}\right)^{\top}. \tag{9}$$

Remark 1. Consider two vectors $\mathbf{v}$ and $\mathbf{w}$ in a Banach space $\mathcal{B}$. The vector $\mathbf{v}$ is *normal* to the vector $\mathbf{w}$ and the vector $\mathbf{w}$ is *transversal* to the vector $\mathbf{v}$ iff $[\mathbf{v},\mathbf{w}]_{\mathcal{B}} = 0$, i.e. the orthogonality relation is not symmetric.

Obviously, the SIP can be plugged into the Hebb-output (2), which leads to the corresponding stationary state equation

$$\mathbf{C}\left[\mathbf{w}\right] = \left[\mathbf{w},\mathbf{C}\left[\mathbf{w}\right]\right]_{\mathbb{B}^n}\mathbf{w} \tag{10}$$

with the covariance matrix $\mathbf{C} = E\left[\mathbf{v}\mathbf{v}^{\top}\right]$. The stability analysis of conventional Oja-learning does not rely on the sesqui-linearity of the inner product [15,16]. Hence, it is applicable also for semi-inner products and, therefore, the update yields the eigenvector corresponding to the largest eigenvalue also in the case of a Banach-space. Again, the extension to the Sanger-approach is straightforward.

Remark 2. It is well-known that l_p-spaces are closely related to the function Banach spaces $\mathcal{L}_p$. The respective SIP

$$[f,g]_p = \frac{1}{\left(\|g\|_p\right)^{p-2}} \int f \cdot |g|^{p-1} sgn\left(g\right) dt \tag{11}$$

is uniformly continuous and, hence, unique [3]. Let $D^{\alpha} = \frac{\partial^{|\alpha|}}{\partial\alpha_1 \ldots \partial\alpha_{|\alpha|}}$ be the differential operator. Then the Banach space $\mathcal{W}_p^K = \{f | D^{\alpha}f \in \mathcal{L}_p, |\alpha| \leq K\}$ is the *Sobolev-space* with the norm

$$\|f\|_{K,p} = \left[\sum_{|\alpha|\leq K}\left(\|D^{\alpha}f\|_p\right)^p\right]^{\frac{1}{p}} = \left[\sum_{|\alpha|\leq K}\int |D^{\alpha}f|^p \, dx\right]^{\frac{1}{p}} \tag{12}$$

and the SIP

$$[f,g]_{K,p} = \frac{1}{\|g\|_{K,p}^{p-2}} \sum_{|\alpha|\leq K} \int f^{(\alpha)} \cdot \left|g^{(\alpha)}\right|^{p-1} sgn\left(g^{(\alpha)}\right) dt \tag{13}$$

with $f^{(\alpha)} = D^{\alpha}f$. Because of lack of space we omit the proof of the latter statement. For $p = 2$, $\mathcal{W}_p^K$ and $\mathcal{L}_p$ are obviously Hilbert spaces.

3 Hebbian Learning for PCA in Reproducing Kernel Spaces

After revisiting properties of kernel spaces including both Hilbert and Banach spaces for reproducing kernel spaces, we explain how the idea of iterative Hebbian learning can be transferred to kernelized problems.

3.1 Kernel Spaces

In the following we assume a compact metric space (V, d_V) with the vector space V equipped with a metric d_V. A function κ on V is a kernel $\kappa_\Phi : V \times V \to \mathbb{C}$ if there exists a Hilbert space $\mathcal{H}$ and a map

$$\Phi : V \ni \mathbf{v} \longmapsto \Phi(\mathbf{v}) \in \mathcal{H} \tag{14}$$

with

$$\kappa_\Phi(\mathbf{v}, \mathbf{w}) = \langle \Phi(\mathbf{v}), \Phi(\mathbf{w}) \rangle_{\mathcal{H}} \tag{15}$$

for all $\mathbf{v}, \mathbf{w} \in V$ and $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ is the inner product of the Hilbert space $\mathcal{H}$. The mapping Φ is called feature map and $\mathcal{H}$ the feature space of V. Without further restrictions on the kernel κ_Φ, both, $\mathcal{H}$ and Φ are not unique. Positive kernels are of special interest because they *uniquely* correspond to a reproducing kernel Hilbert spaces (RKHS) $\mathcal{H}$ in a canonical manner [1,14]. The kernel κ_Φ is said to be positive definite if for all finite subsets $V_m \subseteq V$ with cardinality $\#V_m = m$, the Gram-Matrix

$$\mathbf{G}_m = [\kappa(\mathbf{v}_i, \mathbf{v}_j) : i, j = 1 \ldots m] \tag{16}$$

is positive semi-definite [1]. The norm $\|\Phi(\mathbf{v})\|_{\mathcal{H}} = \sqrt{\kappa_\Phi(\mathbf{\Phi}(\mathbf{v}), \mathbf{\Phi}(\mathbf{v}))}$ of this RKHS induces a metric

$$d_{\mathcal{H}}(\Phi(\mathbf{v}), \Phi(\mathbf{w})) = \sqrt{\kappa_\Phi(\mathbf{v}, \mathbf{v}) - 2\kappa_\Phi(\mathbf{v}, \mathbf{w}) + \kappa_\Phi(\mathbf{w}, \mathbf{w})} \tag{17}$$

based on the kernel κ_Φ [18]. Steinwart has shown that continuous, universal kernels induce the continuity and separability of the corresponding feature map Φ and the image $\mathcal{I}_{\kappa_\Phi} = \Phi(V)$ is a subspace of $\mathcal{H}$ [21]. It was further shown in this paper that continuous, universal kernels also imply the continuity and injectivity of the map

$$\Psi : (V, d_V) \longrightarrow (V, d_{\kappa_\Phi}) \tag{18}$$

with $d_{\kappa_\Phi}(\mathbf{v}, \mathbf{w}) = d_{\mathcal{H}}(\Phi(\mathbf{v}), \Phi(\mathbf{w}))$ and (V, d_{κ_Φ}) is the compact vector space V with the kernel induced metric d_{κ_Φ}. It was shown in [22] that (V, d_{κ_Φ}) is isometric and isomorph to $\mathcal{I}_{\kappa_\Phi}$.

An analogous theory can be obtained if the mapping space has weaker assumptions: Zhang et al. consider reflexive Banach spaces as mapping spaces [25]. As above for the Hilbert space $\mathcal{H}$, the Banach space is also assumed to be a function space, here. Consider such a reflexive function Banach space $\mathcal{B}$ over the compact metric space (V, d_V) with the SIP $[h, g]_{\mathcal{B}}$, which additionally has a reproducing property for Banach spaces (Reproducing Kernel Banach space, RKBS). If the RKBS is Fréchet-differentiable, it is called a SIP-RKBS. Again, we consider the feature map $\varPhi : V \longrightarrow \mathcal{B}$. For a SIP-RKBS $\mathcal{B}$ an unique correspondence exists between a so-called SIP-kernel $\gamma_{\varPhi}$ and the map $\varPhi$ with

$$\gamma_{\varPhi}(\mathbf{v},\mathbf{w}) = [\varPhi(\mathbf{v}), \varPhi(\mathbf{w})]_{\mathcal{B}} \tag{19}$$

based on a Banach space representation theorem [25]. If the the map $\varPhi$ is continuous then also $\gamma_{\varPhi}$ is. Moreover, one can show that (weakly) universal SIP-kernels correspond to bijective mappings $\varPhi$ [22]. Further, it turns out that the map

$$\Psi : (V, d_V) \longrightarrow (V, d_{\mathcal{B}}) \tag{20}$$

is also continuous and, therefore, bijective iff the SIP-kernel is (weakly) universal and continuous. In consequence, the subspace $\mathcal{I}_{\gamma_{\varPhi}} = \varPhi(V) \subseteq \mathcal{B}$ is isomorphic to $(V, d_{\mathcal{B}})$. These results are proofed in [22].

3.2 Kernel Principal Component Analysis

We start this subsection considering a RKHS $\mathcal{H}$ as a mapping space by a map $\varPhi$ from a data vector space V and the corresponding kernel $\kappa_{\varPhi}$. We assume centralized kernels, i.e. $E[\varPhi(\mathbf{v})] = \mathbf{0}$, which can be always be achieved for arbitrary positive kernels and finite data sets [18]. We define $\mathbf{C}_{\varPhi} = E\left[\varPhi(\mathbf{v}) \cdot (\varPhi(\mathbf{v}))^{\top}\right]$. In case of infinite-dimensional $\mathcal{H}$, we have to interpret $\varPhi(\mathbf{v}) \cdot (\varPhi(\mathbf{v}))^{\top}$ as a linear operator $\varOmega_{\mathcal{H}}$ on $\mathcal{H}$

$$\varOmega_{\mathcal{H}}[\mathbf{h}] = \varPhi(\mathbf{v}) \cdot \langle \varPhi(\mathbf{v}), \mathbf{h} \rangle_{\mathcal{H}}. \tag{21}$$

Following Schölkopf et al. in [20] the respective eigen-problem $\mathbf{C}_{\varPhi}\mathbf{g} = \lambda\mathbf{g}$ can be solved using the observation that for all $\mathbf{v} \in V$ the equation $\lambda\langle \varPhi(\mathbf{v}), \mathbf{g} \rangle_{\mathcal{H}} = \langle \varPhi(\mathbf{v}), \mathbf{C}_{\varPhi}\mathbf{g} \rangle_{\mathcal{H}}$ has to be fulfilled. For a data set $D \subset V$ with m data vectors $\mathbf{v}_k$ there exists a dual representation of the eigenvectors $\mathbf{g} = \sum_{j=1}^{m} \alpha_j \varPhi(\mathbf{v}_j)$ such that in this case $\mathbf{C}_{\varPhi}$ becomes the Gram-matrix $\mathbf{G}_m$ from (16). Then the original eigen-problem can be replaced by the dual problem

$$m\lambda\alpha = \mathbf{G}_m\alpha \tag{22}$$

where α is the column vector of the values α_i. According to Zhang et al., this eigen-decomposition can also be seen as an eigen-problem for a linear operator determined by

$$\langle T\mathbf{c}, \mathbf{h} \rangle_{\mathcal{H}} = \frac{1}{m} \sum_{j=1}^{m} \langle \varPhi(\mathbf{v}_j), \mathbf{c} \rangle_{\mathcal{H}} \langle \varPhi(\mathbf{v}_j), \mathbf{h} \rangle_{\mathcal{H}} \tag{23}$$

using the kernel properties [25].

It is possible to extend the RKHS approach to RKBS [25]: Consider an RKBS $\mathcal{B}$ as a mapping space by a map $\varPhi$ from a data vector space V and the corresponding (centralized) SIP-kernel $\gamma_{\varPhi}$. We consider again a data set $D \subset V$ with m data vectors $\mathbf{v}_k$. Let us define for an arbitrary $\mathbf{v} \in \mathcal{B}$ the complex m-dimensional vector

$$\tilde{\Phi}_{\mathcal{B}}(\mathbf{v}) = \left([\Phi(\mathbf{v}), \Phi(\mathbf{v}_1)]_{\mathcal{B}}, \ldots, [\Phi(\mathbf{v}), \Phi(\mathbf{v}_m)]_{\mathcal{B}}\right) \tag{24}$$

such that a linear operator T on $\mathbb{C}^m$ can be defined by

$$T\mathbf{c} = \frac{1}{m}\sum_{j=1}^{m}\left(\tilde{\Phi}^*_{\mathcal{B}}(\mathbf{v}_j)\,\mathbf{c}\right)\tilde{\Phi}_{\mathcal{B}}(\mathbf{v}_j) \tag{25}$$

where $\tilde{\Phi}^*_{\mathcal{B}}(\mathbf{v}_j)$ is the conjugate transpose of $\tilde{\Phi}_{\mathcal{B}}(\mathbf{v}_j)$, which corrresponds to $T\mathbf{c} = \mathbf{M}_m\mathbf{c}$ with

$$\mathbf{M}_m = \frac{1}{m}\left(\mathbf{K}^*_m \cdot \mathbf{K}_m\right)^\top \tag{26}$$

and

$$\mathbf{K}_m = [\gamma_\Phi(\mathbf{v}_i, \mathbf{v}_j) : i, j = 1 \ldots m] \tag{27}$$

is the Gram-matrix of the SIP-kernel γ_Φ. Hence, here the dual problem is

$$\mathbf{M}_m\alpha = \lambda\alpha \tag{28}$$

with the basis representation according to

$$\left\langle \tilde{\Phi}_{\mathcal{B}}(\mathbf{v}), \alpha \right\rangle_{\mathbb{C}^m} = \sum_{j=1}^{m}\overline{\alpha_j}\gamma_\Phi(\mathbf{v}, \mathbf{v}_j)\,. \tag{29}$$

3.3 Kernel PCA and Hebbian Learning

Kernel Hebbian learning based on the Oja-lerning rule (1) was proposed in [12]. It is carried out implicitly in the Hilbert space $\mathcal{H}$ such that the coefficient vector α in (22) is iteratively determined using the Gram-matrix $\mathbf{G}_m$ from (16). This approach can be transferred to the kernel Banach space problem in a straightforwad manner by replacing, in the terms containing $\mathbf{G}_m$, the respective parts by $\mathbf{M}_m$ from (26). Due to the lack of space we drop the explicit formulation and follow a different route: We consider the mapping Ψ for RKHS and RKBS in the following.

3.3.1 Hebbian PCA Learning in $(V, d_{\mathcal{H}})$

Now, we process PCA in the space $(V, d_{\mathcal{H}})$ from (18) using its ismorphy to the image space $\mathcal{I}_{\kappa_\Phi} \subseteq \mathcal{H}$ of the kernel mapping Φ such that the data remain the original ones but are equipped with the kernel metric. Furthermore, we assume centralized kernels such that $E[\Psi(\mathbf{v})] = \mathbf{0}$. Now Oja's learning rule (1) in $(V, d_{\mathcal{H}})$ for given $\mathbf{v} \in (V, d_V)$ is given as

$$\triangle\mathbf{w} = O \cdot (\Psi(\mathbf{v}_k) - O \cdot \mathbf{w}) \tag{30}$$

where

$$O = \kappa_\Phi(\Psi(\mathbf{v}_k), \mathbf{w}) \tag{31}$$

is the new non-Euclidean Hebbian response instead of the Euclidean inner product used in the original Oja's learning rule [15]. Substituting this in (30) we get

$$\triangle \mathbf{w} = \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \cdot \Psi\left(\mathbf{v}_k\right) - \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \cdot \mathbf{w}, \tag{32}$$

which can be rewritten to

$$\triangle \mathbf{w} = \Omega\left[\mathbf{w}\right] - \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \cdot \mathbf{w} \tag{33}$$

using the linear operator $\Omega = \Psi\left(\mathbf{v}_k\right) \cdot \left(\Psi\left(\mathbf{v}_k\right)\right)^{\top}$ with

$$\Omega\left[\mathbf{w}\right] = \Psi\left(\mathbf{v}_k\right) \cdot \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right), \tag{34}$$

which is comparable to in (21). Under the usual assumption that the prototype $\mathbf{w}$ changes slowly compared to the number of presented inputs we get

$$\triangle \mathbf{w} = \mathbf{C}_{\Psi}\left[\mathbf{w}\right] - \lambda \mathbf{w} \tag{35}$$

with

$$\mathbf{C}_{\Psi} = E\left[\Omega\right] \tag{36}$$

defining the covariance in $(V, d_{\mathcal{H}})$, which reduces to

$$\mathbf{C}_{\Psi} = \frac{1}{n}\sum_{j=1}^{n} \Psi\left(\mathbf{v}_j\right) \cdot \left(\Psi\left(\mathbf{v}_j\right)\right)^{\top} \tag{37}$$

for a finite number of samples $V = \{\mathbf{v}_k | k = 1 \ldots n\}$.

The value λ in eq. (35) is the expectation

$$\lambda = E\left[\kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \cdot \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right)\right] \tag{38}$$

of the squared non-Euclidean Hebbian response O from (31). Thus, we obtain in the stationary state $\triangle \mathbf{w} = 0$ an eigenvalue equation $\mathbf{C}_{\Psi}\left[\mathbf{w}\right] = \lambda \mathbf{w}$ for the operator $\mathbf{C}_{\Psi}$ for an eigenvector $\mathbf{w} \neq \mathbf{0}$ and eigenvalues $\lambda > 0$. The last inequality stems from the positive definiteness of the kernel.

Because $\mathbf{w} \in (V, d_{\mathcal{H}})$, we may conclude that $\mathbf{w} \in \operatorname{span}\left\{\Psi\left(\mathbf{v}_j\right) | j = 1 \ldots n\right\}$ holds. Hence, the relation

$$\lambda \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) = \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{C}_{\Psi}\left[\mathbf{w}\right]\right) \tag{39}$$

must be valid for all $k = 1 \ldots n$. Moreover, $\mathbf{w}$ can be expressed as a linear combination

$$\mathbf{w} = \sum_{j=1}^{n} \alpha_j \Psi\left(\mathbf{v}_j\right)$$

of the images $\Psi\left(\mathbf{v}_k\right)$ of the original data vectors. Putting together the last statement with (39) we get

$$\lambda \sum_{j=1}^{n} \alpha_j \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \Psi\left(\mathbf{v}_j\right)\right) = \frac{1}{n}\sum_{j=1}^{n} \alpha_j \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \sum_{i=1}^{n} \Psi\left(\mathbf{v}_i\right) \cdot \kappa_{\Phi}\left(\Psi\left(\mathbf{v}_i\right), \Psi\left(\mathbf{v}_j\right)\right)\right). \tag{40}$$

Here we have used the linearity of the kernel, interpreted as a real inner product, and the definition of $\mathbf{C}_{\Psi}$ in (36). If we now take into account the definition of the Gram-matrix $\mathbf{G}_n$ in (16), we immediately obtain

$$n\lambda\mathbf{G}_n\alpha = \mathbf{G}_n^2\alpha \tag{41}$$

where $\alpha = (\alpha_1, \ldots, \alpha_n)^{\top}$, which corresponds to the solution of the so-called dual eigen-problem (22) in [18], and, hence, the stability analysis can be taken from [12], which also delivers the extension to the full eigen-problem and the respective Sanger-algorithm.

3.3.2 Hebbian PCA Learning in $(V, d_{\mathcal{B}})$

Here we consider the space $(V, d_{\mathcal{B}})$ from (20) and exploit its isomorphism to the image space $\mathcal{I}_{\gamma_{\Phi}} \subseteq \mathcal{B}$ of the kernel mapping Φ for a SIP-RKBS $\mathcal{B}$. Again, we assume centralized kernels satisfying $E\left[\Psi\left(\mathbf{v}\right)\right] = \mathbf{0}$. Further, we assume that the kernel γ_{Φ} takes only real values. Hence, $\mathbf{K}_m^* = \mathbf{K}_m^{\top}$ is valid in (26) which results in $\mathbf{M}_m = \frac{1}{m}\left(\mathbf{K}_m^{\top} \cdot \mathbf{K}_m\right)$ being symmetric and positive definite. The non-Euclidean Hebb-response becomes

$$O = \gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \tag{42}$$

Substituting this in (30) we get in complete analogy

$$\triangle\mathbf{w} = \gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \cdot \Psi\left(\mathbf{v}_k\right) - \gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right)\gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \cdot \mathbf{w}, \tag{43}$$

which reads as

$$\triangle\mathbf{w} = \mathbf{C}_{\Psi}^{\mathcal{B}}\left[\mathbf{w}\right] - \lambda\mathbf{w} \tag{44}$$

but here with $\Omega_{\mathcal{B}}\left[\mathbf{w}\right] = \Psi\left(\mathbf{v}_k\right) \cdot \gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right)$ and $\mathbf{C}_{\Psi}^{\mathcal{B}} = E\left[\Omega_{\mathcal{B}}\right]$. However, because $\Psi\left(\mathbf{v}_k\right) = \mathbf{v}_k$ despite the changing metric we have $\mathbf{C}_{\Psi}^{\mathcal{B}} = \mathbf{C}$ and $\mathbf{C}$ being the covariance of the data in D.[1] The value λ in eq. (44) is the expectation

$$\lambda = E\left[\gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) \cdot \gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right)\right] \tag{45}$$

of the squared non-Euclidean Hebbian response O from Eq. (42). The stationary state $\triangle\mathbf{w} = 0$ corresponds to the eigen equation $\mathbf{C}_{\Psi}^{\mathcal{B}}\left[\mathbf{w}\right] = \mathbf{C}\mathbf{w} = \lambda\mathbf{w}$ for the operator $\mathbf{C}_{\Psi}^{\mathcal{B}}\left[\mathbf{w}\right]$ with eigenvector $\mathbf{w} \neq \mathbf{0}$ and eigenvalue $\lambda \neq 0$.

Because $\mathbf{w} \in (V, d_{\mathcal{H}})$, we may conclude that $\mathbf{w} \in \operatorname{span}\left\{\Psi\left(\mathbf{v}_j\right) | j = 1 \ldots n\right\}$ holds, because $\mathcal{B}$ is a SIP-RKBS. Hence, the relation

$$\lambda\gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{w}\right) = \gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \mathbf{C}_{\Psi}^{\mathcal{B}}\left[\mathbf{w}\right]\right) \tag{46}$$

must be valid for all $k = 1 \ldots n$. Moreover, $\mathbf{w}$ can be expressed again as a linear combination $\mathbf{w} = \sum_{j=1}^{n}\beta_j\Psi\left(\mathbf{v}_j\right)$ of the images $\Psi\left(\mathbf{v}_k\right)$ of the original data vectors. Putting together the last statement together with (46) we get

$$\lambda\sum_{j=1}^{n}\beta_j\gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \Psi\left(\mathbf{v}_j\right)\right) = \frac{1}{n}\sum_{j=1}^{n}\beta_j\gamma_{\Phi}\left(\Psi\left(\mathbf{v}_k\right), \sum_{i=1}^{n}\Psi\left(\mathbf{v}_i\right) \cdot \gamma_{\Phi}\left(\Psi\left(\mathbf{v}_i\right), \Psi\left(\mathbf{v}_j\right)\right)\right) \tag{47}$$

[1] We emphasize at this point that, $\Psi\left(\mathbf{v}_k\right) = \mathbf{v}_k$ is valid only numerically. Yet, $\mathbf{v}_k$ and its image $\Psi\left(\mathbf{v}_k\right)$ are objects in *different metric spaces*. Therefore, we will still use the notation $\Psi\left(\mathbf{v}_k\right)$ for the image to indicate this difference.

using the linearity of the SIP-kernel in its first argument, interpreted as a real semi-inner product, and the definition of $\mathbf{C}_{\Psi}^{\mathcal{B}}$ as expectation. If we now take into account the definition of the Gram-matrix $\mathbf{K}_n$ in (27), we immediately conclude

$$n\lambda\mathbf{K}_n\beta = \mathbf{K}_n^2\beta \tag{48}$$

where $\beta = (\beta_1, \ldots, \beta_n)^{\top}$ plays the same role as β in (41). Moreover, it relates via the operator eigen-problems for RKHS (23) and RKBS (25) to the dual problem in case of RKBS (28).

As it was shown for the RKHS in [12], the stability analysis for RKBS follows analogously keeping also in mind that the original stability analysis in [15] does not require the sesqui-linearity of the inner product. Again, the extension to full PCA according to Sanger [17] is obvious.

4 Conclusion

In this paper we tackle the problem of PCA in non-Euclidean spaces. This is an important task if data have to be visualized and the dissimilarity measure is non-Euclidean, as it is the case in classification problems with metric adaptation, for example. We provide the theoretical framework for non-Euclidean PCA for Hebbian learning by Oja's learning rule. We mathematically proof that adaptive PCA by Hebbian learning can be done for general finite-dimensional Banach and Hilbert spaces and also in the context of kernel metrics with underlying RKHS and RKBS. Thus, we close the gap between kernel based learning and adequate data visualization if kernel learning is done using *differentiable* kernels, which allow prototype based learning in the data space but equipped with a differentiable kernel metric.

References

[1] Aronszajn, N.: Theory of reproducing kernels. Transactions of the American Mathematical Society 68, 337–404 (1950)

[2] Der, R., Lee, D.: Large-margin classification in Banach spaces. In: JMLR Workshop and Conference Proceedings. AISTATS, vol. 2, pp. 91–98 (2007)

[3] Giles, J.: Classes of semi-inner-product spaces. Transactions of the American Mathematical Society 129, 436–446 (1967)

[4] Günter, S., Schraudolph, N., Vishwanathan, S.: Fast iterative kernel principal component analysis. Journal of Machine Learning Research 8, 1893–1918 (2007)

[5] Hammer, B., Villmann, T.: Generalized relevance learning vector quantization. Neural Networks 15(8-9), 1059–1068 (2002)

[6] Haykin, S.: Neural Networks. A Comprehensive Foundation. Macmillan, New York (1994)

[7] Hein, M., Bousquet, O., Schölkopf, B.: Maximal margin classification for metric spaces. Journal of Computer Systems Sciences 71, 333–359 (2005)

[8] Hoffmann, T., Schölkopf, B., Smola, A.: Kernel methods in machine learning. The Annals of Statistics 36(3), 1171–1220 (2008)

[9] Joliffe, I.: Principal Component Analysis, 2nd edn. Springer (2002)

[10] Kästner, M., Hammer, B., Biehl, M., Villmann, T.: Functional relevance learning in generalized learning vector quantization. Neurocomputing 90(9), 85–95 (2012)

[11] Kim, K., Franz, M., Schölkopf, B.: Kernel hebbian algorithm for iterative kernel principal component analysis. Technical Report 109, Max-Planck-Institute for Biological Cybernetics (June 2003)

[12] Kim, K., Franz, M., Schölkopf, B.: Iterative kernel principal component analysis for image modelling. IEEE Transactions on Pat. 27(9), 1351–1366 (2005)

[13] Lumer, G.: Semi-inner-product spaces. Transactions of the American Mathematical Society 100, 29–43 (1961)

[14] Mercer, J.: Functions of positive and negative type and their connection with the theory of integral equations. Philosophical Transactions of the Royal Society, London, A 209, 415–446 (1909)

[15] Oja, E.: Neural networks, principle components and suspaces. International Journal of Neural Systems 1, 61–68 (1989)

[16] Oja, E.: Nonlinear pca: Algorithms and applications. In: Proc. of the World Congress on Neural Networks, Portland, pp. 396–400 (1993)

[17] Sanger, T.: Optimal unsupervised learning in a single-layer linear feedforward neural network. Neural Networks 12, 459–473 (1989)

[18] Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press (2002)

[19] Schneider, P., Hammer, B., Biehl, M.: Adaptive relevance matrices in learning vector quantization. Neural Computation 21, 3532–3561 (2009)

[20] Schölkopf, B., Smola, A., Müller, K.-R.: Nonlinear component analysis as a kernel eigenvalue problem. Neur. 14(7), 1299–1319 (1998)

[21] Steinwart, I.: On the influence of the kernel on the consistency of support vector machines. Journal of Machine Learning Research 2, 67–93 (2001)

[22] Villmann, T., Haase, S.: A note on gradient based learning in vector quantization using differentiable kernels for Hilbert and Banach spaces. Machine Learning Reports 6(MLR-02-2012), 1–29 (2012) ISSN:1865-3960, http://www.techfak.uni-bielefeld.de/fschleif/mlr/mlr$_$02$_$2012.pdf

[23] Villmann, T., Hammer, B.: Functional Principal Component Learning Using Oja's Method and Sobolev Norms. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 325–333. Springer, Heidelberg (2009)

[24] von Luxburg, U., Bousquet, O.: Distance-based classification with Lipschitz functions. Journal of Machine Learning Research 5, 669–695 (2004)

[25] Zhang, H., Xu, Y., Zhang, J.: Reproducing kernel banach spaces for machine learning. Journal of Machine Learning Research 10, 2741–2775 (2009)

Classification of Chain-Link and Other Data with Spherical SOM

Masaaki Ohkita[1], Heizo Tokutaka[1], Makoto Ohki[2], Matashige Oyabu[3], and Kikuo Fujimura[2]

[1] SOM Japan Inc., Tottori, Japan
[2] Tottori University, Tottori, Japan
[3] Kanazawa Institute of Technology, Nonoichi, Ishikawa, Japan
tokutaka@somj.com

Abstract. A new classification method is proposed based on the spherical SOM that has been developed earlier for visualizing multidimensional data sets. Phase distances between labeled data on the spherical surface are computed. With these distances, a dendrogram can be constructed. Then, using the constructed dendrogram, a classification of each cluster group on the spherical surface, based on the label data was carried out. This method can be applied to various data sets. Here, the method was applied to the chain-link problem which can be considered as a particularly difficult one from a representational standpoint, and to the problem of separating parallel random number planes.

Keywords: Spherical SOM, colored classification, chain-link problem, the separation of the random number planes.

1 Introduction

A clustering method based on the visualization of a multidimensional data set on a sphere was recently proposed [1]. The phase distance between labeled data represented on the spherical surface was computed and a dendrogram constructed from the phase distance calculation. In the current paper, we perform the classification of the found clusters by applying labeled data to the dendrogram.

This new classification method can be applied to various data sets. Here, we apply it to the chain-link problem [2] considered to be notoriously difficult, and to the problem of the separation of stacked random number planes [3].

2 Application to the Chain-Link Problem

2.1 The Comparison of the Results Obtained with the XOM Method and the Spherical SOM for a Benchmark Problem

The benchmark problem consists of two rings (links) (Fig. 1(a)) with a distance d from each other and intersecting vertically in the three-dimensional space. With the

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 35–44.
DOI: 10.1007/978-3-642-35230-0_4

XOM method, the rings become projected into a two dimensional space [2], hiding the three-dimensional structure (Fig. 1(b)). In Fig. 1(c), the U-matrix can be seen as gray shades by the spherical SOM. The form resembles very well the seam of a regular baseball. The rings don't intersect. An expression at Glyph value 0.5 and a figure of the corresponding color boundary are shown in Fig. 2 (b) and (c), respectively. As a result, it is possible to clearly identify, without intersection, the boundary between rings 1 and 2 of Fig. 1(a) and Fig. 2 (a). Characteristic positions in the chain-link are shown in Figs. 2-3.

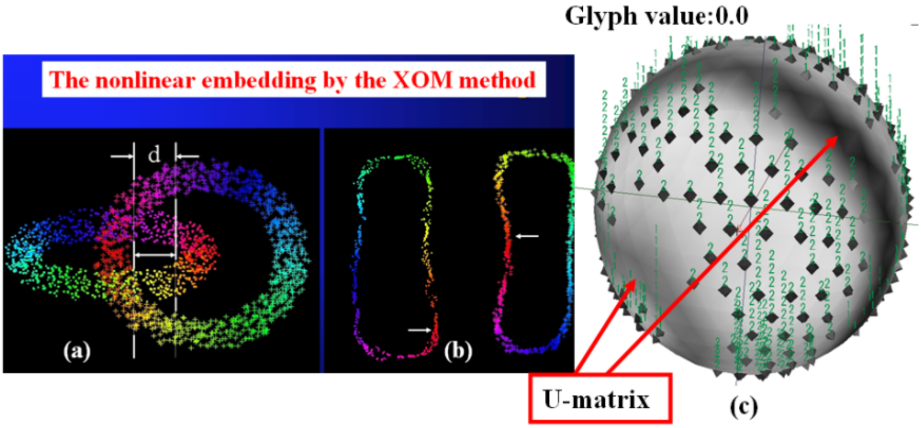


Fig. 1. (a) the input 3-dimensional data (b) the 2 dimensional result obtained with the XOM method [2] (c) The representation obtained with spherical SOM (blossom) [1, 4, 5]

2.2 Representation Using the Spherical SOM of Positional Information in the Chain-Link

The typical positions of chain-link of Figs. 1 and 2 are shown in the following Fig. 2-3. Then, a representation is developed using a spherical SOM.

The representations in terms of coordinates on the ring A and B are shown in Fig. 3(a) and (b), respectively.

In Fig. 5(a), the first ring starts in A(-0,1,-z) and ends in A(+0,1,+z). In Fig. 5(b), the second ring ends in B(+x,0,+0) and starts in B(-x,0,-0) . In Fig. 5, A(-0,1,-z) corresponds to B(-x,0,-0) (blue circled) and A(+0,1,+z) corresponds to B(+x,0,+0) (black circled), since their distances are nearer compared to any other combination. Here, -0 signifies a negative value that is near 0, and +0 a positive value that is near to 0. -z signifies a negative z value, -x a negative x value, + z, and + x, are defined in the same way.

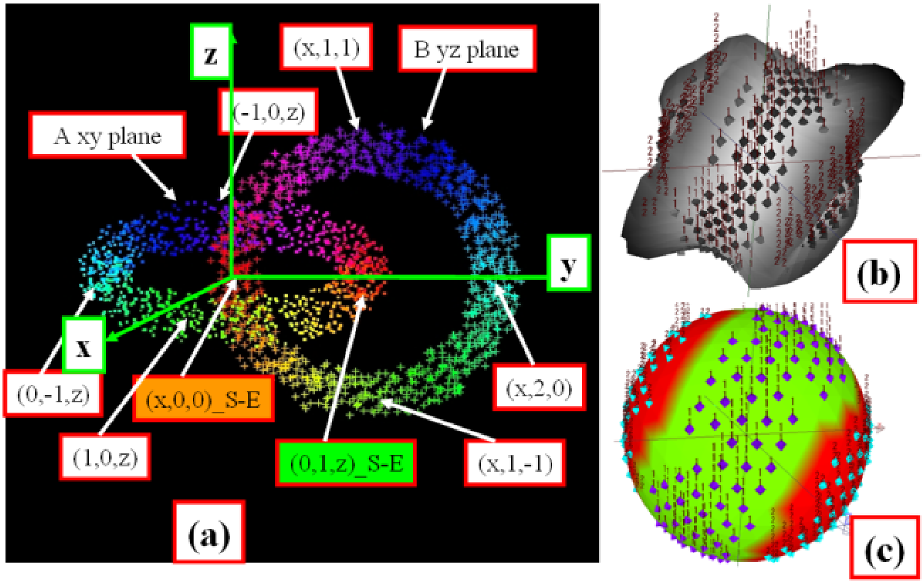


Fig. 2. (a) the coordinates inside ring A and B. (b) the figure with the Gryph value 0.5 of Fig. 1(c). (c) The corresponding boundary is shown by coloring.

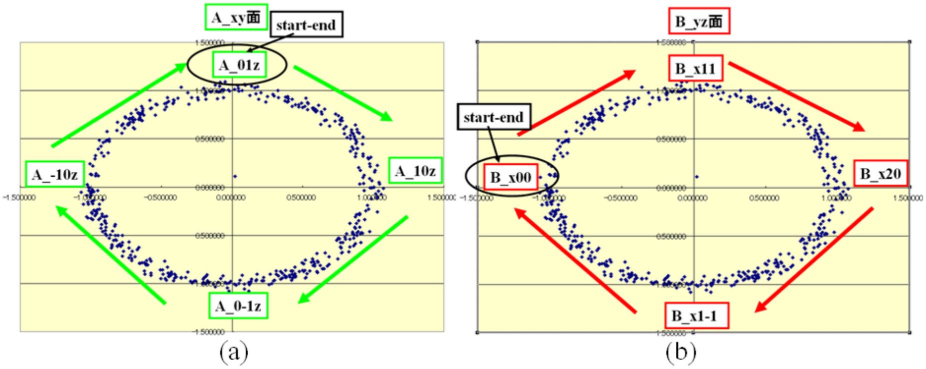


Fig. 3. (a) Each of the coordinates (Fig. 2) of the A ring are on the spherical surface, and the arrow starts and ends in A(0,1,z). (b) each of the coordinates (Fig. 2) on the B ring are on the spherical surface, and the arrow starts and ends in B(x,0,0).

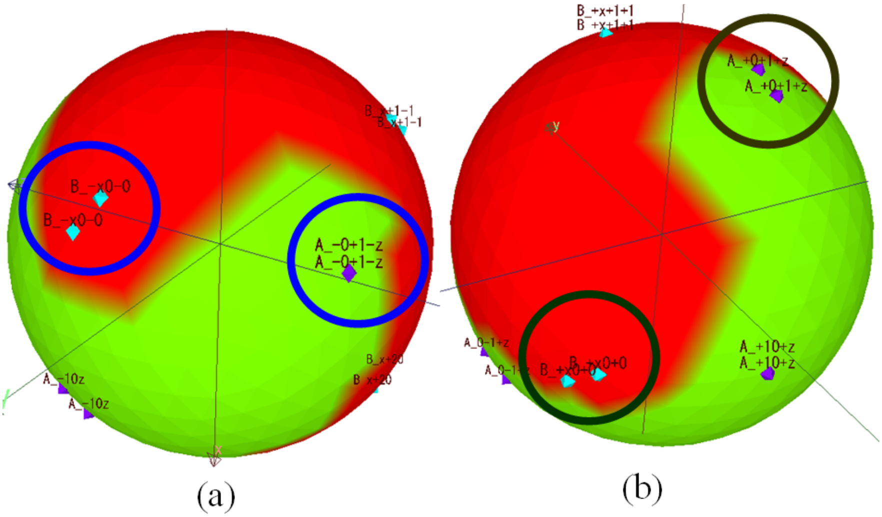
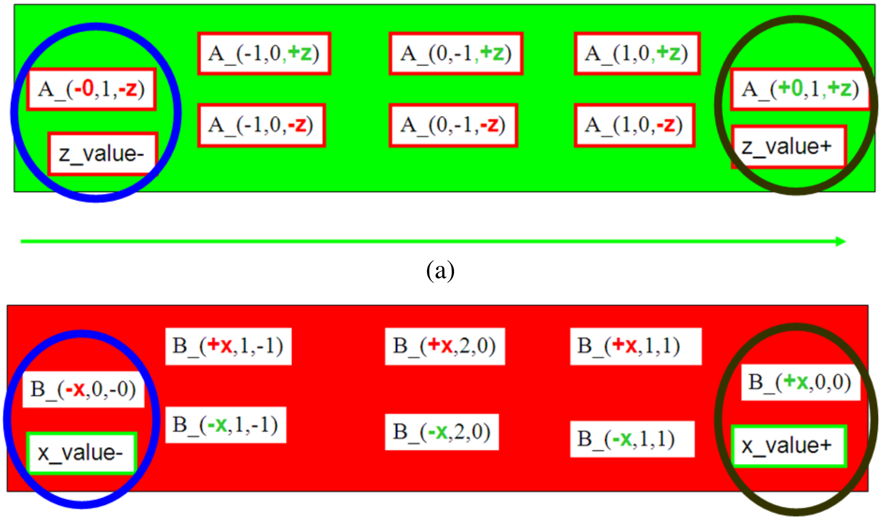


Fig. 4. (a) the starting (blue) and (b) the end (black) position in Fig. 3



(a)

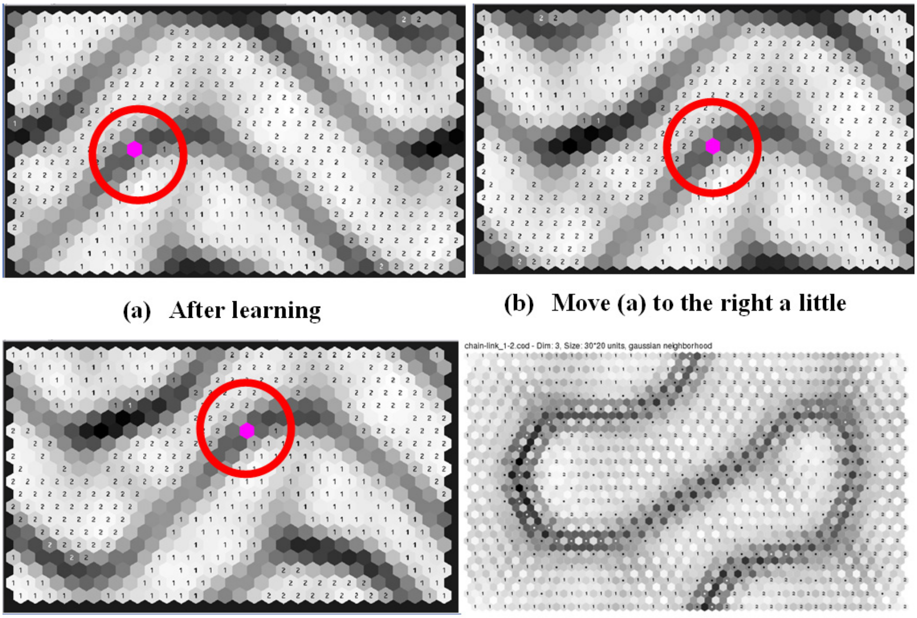
(b)

Fig. 5. Representation obtained when Fig. 4 is stretched to one dimension. In panels (a) and (b), the labels A and B, are the corresponding positions of start (left blue circled) and end (right black circled).

2.3 The Results Obtained with Other SOMs (Torus-SOM and SOM_PAK)

Other SOM methods were also tried, namely, Torus-SOM [5] and SOM_PAK [5, 6]. Torus-SOM is a kind of SOM for which either side of the map, including the top and bottom sides are designed to be continuous (i.e., to form a torus). SOM_PAK is free software of the regular SOM, developed at the Helsinki institute of technology in Finland [6].

The result of the Torus-SOM is shown in Fig. 6. Let's pay attention to the Cyan mark (red circled) on the U-matrix in the central left of the figure. The mark in Fig. 6(a) was moved about 10 steps to the right in Fig. 6(b), as the map is continuous in either side. Similarly, in Fig. 6(c), it is moved up by about 5 steps compared with Fig. 6(b).



(a) After learning

(b) Move (a) to the right a little

(c) Move (b) a little to the upward

(d) The result by SOM_PAK

Fig. 6. (a) After Torus-SOM learning, (b) the map (a) is slightly moved to the right, and (c) moved upwards**.** (d) The result obtained with SOM_PAK learning.

Finally, the result obtained after learning, using the SOM_PAK is shown in Fig. 6(d). In the case of the Torus-SOM of Figs. 6, the map is continuous on all sides. The SOM_PAK is a usual SOM. However, this SOM does not continue at all sides. As a result, after learning with SOM_PAK, the continuity of the representation provided by the spherical SOM (blossom) and Torus-SOM cannot be observed. As shown in Fig. 6(d), the map is divided only into two areas.

3 Application to the Case of Layered Random Number Planes

3.1 Preparation of the Layered Random Number Planes

A set of parallel random number planes [3] as shown in Fig. 7 are prepared. The layers are separated by the distance d, for which we explore the values 0.1, 1, 5, and 10.

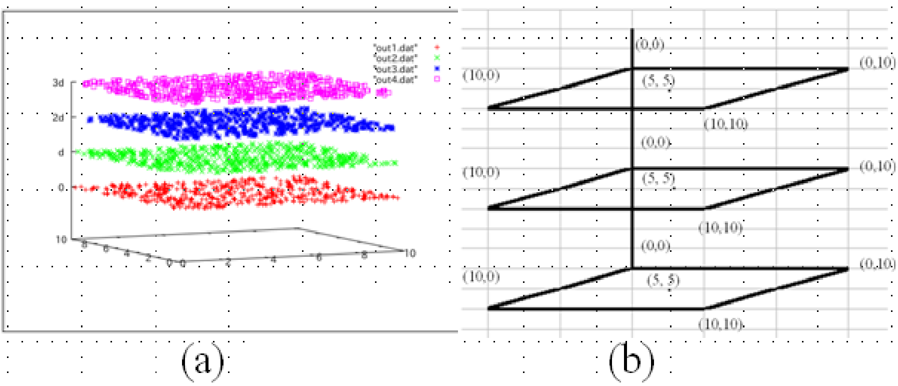


Fig. 7. (a) A stack of 4 random number layers is constructed sized 10×10 with every layer containing 500 random points. The layers are separated by a distance d. (b) Typical case of 4 corners and 1 center position in each layer; the spacing d is 10.

3.2 Analysis with Spherical SOM

The cases of 2 layers, 3 layers and 4 layers were analyzed by a spherical SOM. One example of the obtained results is shown in Fig. 8. The distance d is mentioned in the

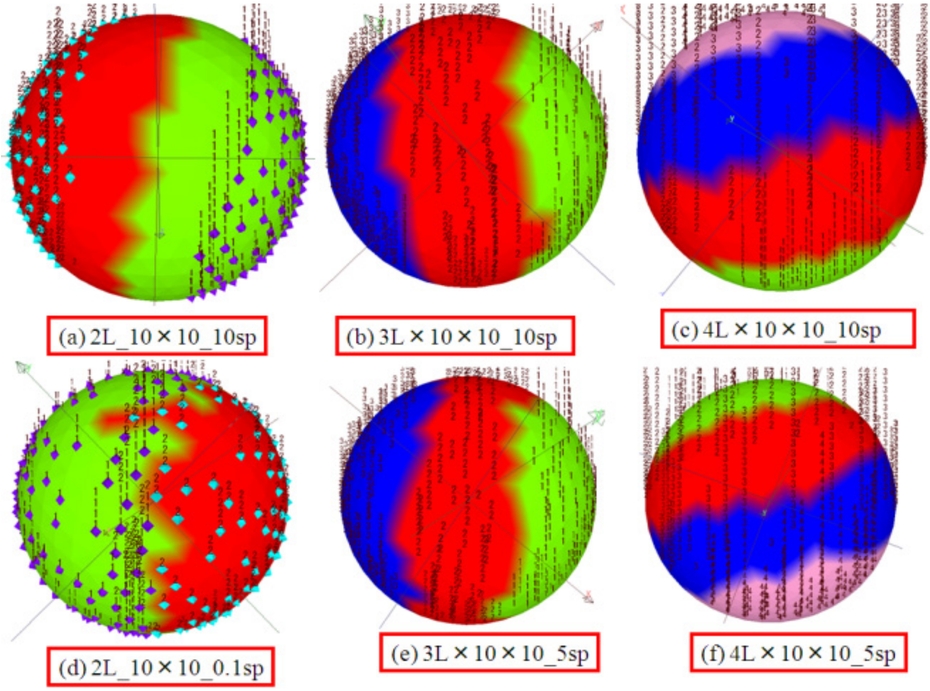
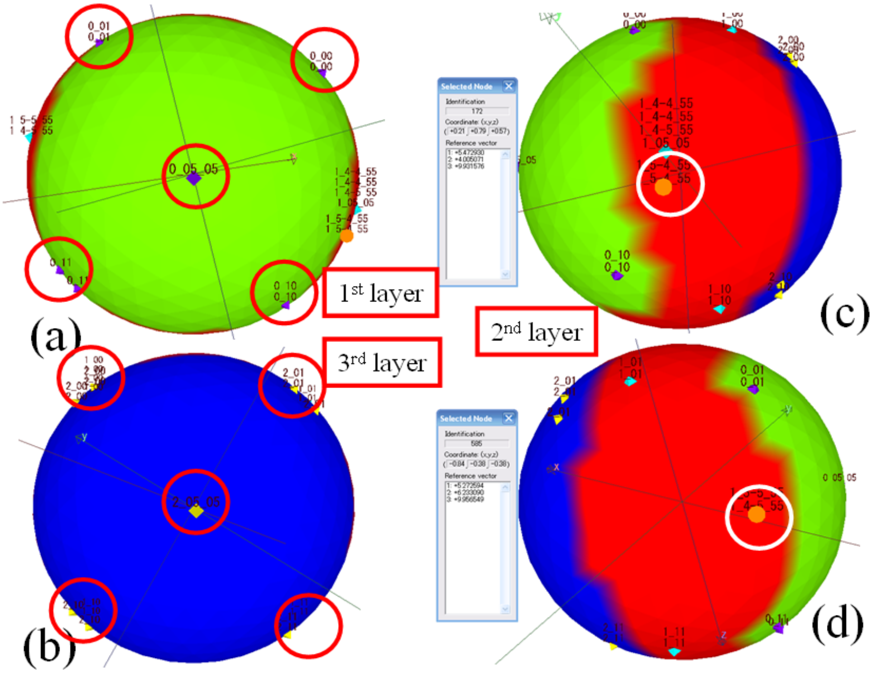


Fig. 8. The spherical SOM results for the case of (a) 2 layers, d=10, (b) 3 layers, d=10, (c) 4 layers, d=10, (d) 2 layers, d=0.1, (e) 3 layers, d=5, and (f) 4 layers, d=5

caption of Fig. 8. In the case of 2 layers, Fig. 8(d) shows that the boundary is mixed and not smoothed when d is taken as 0.1. However, in the case of 3 layers and 4 layers, the boundaries could not be separated with a d equal to 1. Therefore, the cases of d=5, and 10 where the boundaries could be separated are shown respectively in (b), (e), and (c), (f).

As shown in Fig. 8, in the case of 2 layers, the resolution is good with d=0.1. The resolution fails for 3 layers, and 4 layers. There, the layers are fully mixed for d=1 the layers could not be separated. Next, d was set to 10 in all cases. The resolution results for 2 layers, 3 layers, and 4 layers were good as shown in Fig. 8 (a), (b), and (c).

In the case of 3 layers, let us consider the representation of the central coordinates (5, 5) (Fig. 7(b)). As shown in Fig. 9(a) and (b), the coordinates are represented in the center of layers 1 and 3. However, for the 2nd layer, when the y coordinate is smaller than 5, it is represented as shown in Fig. 9(c) on the right-hand side, seen from the 1st layer. When y is larger than 5, it is represented as shown in Fig. 9(d) on the left-hand



Selected Node
Identification
172
Coordinate: (x,y,z)
(+0.21 , +0.79 , +0.57)
Reference vector
1: +5.472930
2: +4.005071
3: +9.931576

Selected Node
Identification
585
Coordinate: (x,y,z)
(-0.84 , -0.38 , -0.38)
Reference vector
1: +5.272594
2: +6.233090
3: +9.966549

Fig. 9. The 3 layer structure is learned by the spherical SOM. 4 corners and 1 center in the 1st and the 3rd layers are learned as shown in panels (a) and (b). However, the center of the 2nd layer is at the right hand side of the 1st layer as shown in panel (c) and at the left hand side of the 1st layer as shown in panel (d) where the position in (c) is (5.47, 4.00, 9.97) and in (d) is (5.27, 6.23, 9.97).

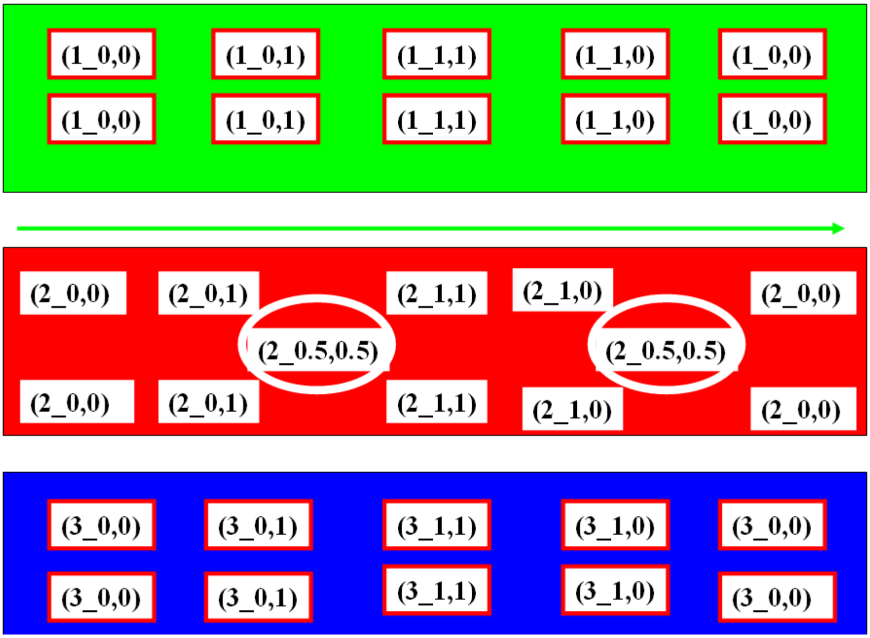


Fig. 10. The result obtained when the map of Fig. 9 is stretched into the one dimension. The result is shown starting from the top panel for the 1st layer, then the 2nd layer and 3rd layer, respectively. In the top and bottom panels, the 4 corners are represented. The center position can't be shown in one dimension. However, the center position of the 2nd layer can be shown in one dimension (middle panel, white circled).

side. Considering the 2 center positions (5.47, 4.00, 9.97) and (5.27, 6.23, 9.97) in the 2nd layer, the difference between the x coordinates 5.47 and 5.27 is only 0.20. However, the difference between the y coordinates 4.00 and 6.23 is 2.23. Due to the larger difference between the y coordinates than the x coordinates, the projected positions of the centers in Figs 9(c) and (d) can be understood.

3.3 Analysis by Torus-SOM and Ordinary SOM

Next, the result obtained with an ordinary SOM is shown in Fig. 11. SOM_PAK is used for the analysis. The cases of 2 layers and 3 layers are shown in Fig. 11. Here, the interval d equals 10.

As evidenced by Fig. 11, the boundaries between the layers are clearly shown by the U-matrix. Next, the learning is repeated for the cases of 2 layers and 3 layers with the Torus-SOM. When the 2 layers case is analyzed with the Torus-SOM, a boundary is expressed by the U matrix as in Fig.11. Moreover, when the case of 3 layers is analyzed, the result is divided into 3 layers. The boundaries among the layers are not distinguishable when they are observed by the U-matrix.

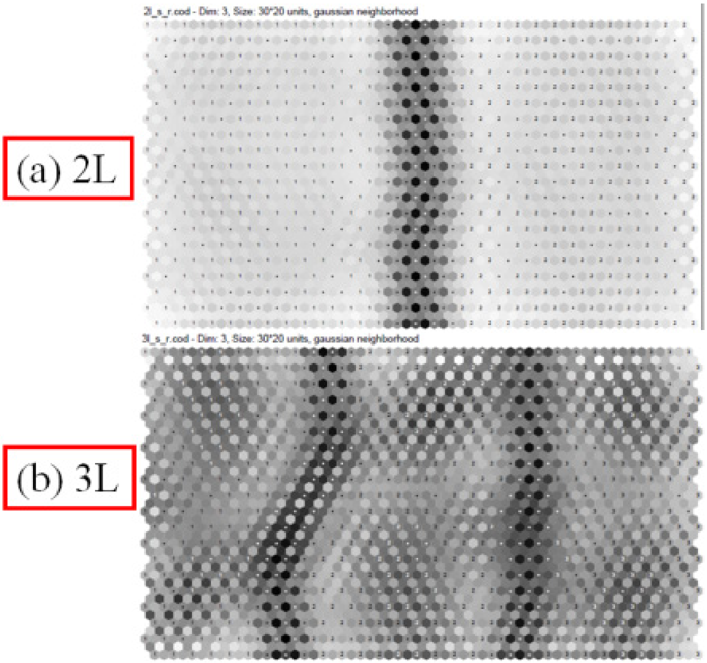

2l_s_r.cod - Dim: 3, Size: 30*20 units, gaussian neighborhood
(a) 2L
3l_s_r.cod - Dim: 3, Size: 30*20 units, gaussian neighborhood
(b) 3L

Fig. 11. The analysis with SOM_PAK for (a) the case of 2 layers and (b) 3 layers, where the distance d among the layers is 10

4 Conclusion

The chain-link benchmark consisting of two interlocked rings without crossing, and perpendicular in three dimensions was examined with different SOM methods. The chain-link becomes problematic to represent when projecting on a planar surface. The conventional result shows that the two rings are separated but the crossing is not learned. The results are summarized as follows: methods.

1. Two areas are distinguished by SOM_PAK. However, the two rings and their crossing are not shown.
2. As for the Torus-SOM, the result approaches a ring by virtue of the continuity of the map (i.e., a torus).
3. When a cluster spherical SOM (blossom) is used, a boundary could be observed after coloring the clusters. When the boundary of rings 1(A) and 2(B) is traced on the map, they don't cross, from which we conclude that the configuration of Fig. 1 (a) is properly learnt.
4. When Ring_1 (A) and Ring_2 (B) are represented on the spherical surface, the boundary that is expressed from the U-matrix appears on the spherical surface. The form is a shaped like the seam of a baseball. When the Glyph value is changed to 0.5, the boundary is clearly expressed in the form of an edge.

5. When 8 points in total are represented on the spherical surface, each 4 connected points can be represented in one dimension. The starting point and the end point in one dimension are the points (0,1,z) on the A ring and (x,0,0) on the B ring where the A and B rings most closely separated. When the boundaries are stretched into one dimension, positive values in the z, x directions, and negative ones correspond to the points (0,1,z) on the A ring and (x,0,0) on the B ring, respectively (refer to Figs. 4 and 5).

In addition to the chain-link benchmark, layered random number planes case was also analyzed. For a spherical SOM, the arrangement of the layered structure is well shown and is superior to that of the Torus-SOM and the planar SOM.

Finally, the one more important result will be added. As shown in Fig. 3(b) by E-SOM [2], the iris data [2,7,8] of the benchmark problem where three kinds of each 50 stocks were classified. As the result, setosa are fully separated. Three virginicas in the virgicolor group and 3 virgicolors in the virginica group are falsely classified respectively. In our present proposal method, three kinds of setosa, virgicolor, and virginica groups are fully classified with color [1].

Thus, it can be understood that a cluster spherical SOM (blossom) can solve high degree advanced benchmark problems like the separation of the two ring configuration (chain-link problem), the random number plane layers and additionally iris data [2,7,8]. Finally, the authors are very obliged to Prof. M. V. Hulle of K.U.Leuven Belgium, for his kind reading and correcting our manuscript.

References

[1] Tokutaka, H., Ohkita, M., Hai, Y., Fujimura, K., Oyabu, M.: Classification Using Topologically Preserving Spherical Self-Organizing Maps. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 308–317. Springer, Heidelberg (2011)

[2] Herrmann, L., Ultsch, A.: Clustering with Swarm Algorithms Compared to Emergent SOM. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 80–88. Springer, Heidelberg (2009)

[3] Hammer, B., Biehl, M., Bunte, K., Mokbel, B.: A General Framework for Dimensionality Reduction for Large Data Sets. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 277–287. Springer, Heidelberg (2011)

[4] http://www.somj.com

[5] Ohkita, M., Tokutaka, H., Fujimura, K., Gonda, E.: Self-Organizing Maps and the tool. Springer Japan Inc. (2008) (in Japanese)

[6] Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, Berlin (2001)

[7] Fisher, R.A.: The Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics 7, 179–188 (1936)

[8] http://www.ics.uci.edu/~mlearn/databases/

Unsupervised Weight-Based Cluster Labeling for Self-Organizing Maps

Willem S. van Heerden and Andries P. Engelbrecht

Computational Intelligence Research Group,
Department of Computer Science,
University of Pretoria, Pretoria, South Africa
{wvheerden,engel}@cs.up.ac.za

Abstract. Self-organizing maps (SOMs) have been applied for practical data analysis, in the contexts of exploratory data analysis (EDA) and data mining (DM). Many SOM-based EDA and DM techniques require that descriptive labels be applied to a SOM's neurons. Several techniques exist for labeling SOM neurons in a supervised fashion, using classification information associated with a set of labeling data examples. However, classification information is often unavailable, necessitating the use of unsupervised labeling approaches that do not require pre-classified labeling data. This paper surveys existing unsupervised neuron labeling techniques. A novel unsupervised labeling algorithm, namely unsupervised weight-based cluster labeling, is described and critically discussed. The proposed method labels emergent neuron clusters using sub-labels built from statistically significant weights. Visualizations of the labelings produced by a prototype of the proposed approach are presented.

Keywords: Artificial neural networks, self-organizing maps, exploratory data analysis, data mining, neuron labeling, visualization.

1 Introduction

A self-organizing map (SOM) is an unsupervised neural network [14]. Much research exists on SOMs [10,17,18], and the approach has been applied to many practical problems, ranging from industrial [1] to financial [7] applications.

This paper views exploratory data analysis (EDA) and data mining (DM) as distinct concepts, both of which extract knowledge from sets of data [24]:

- EDA is a human-centered approach to extracting knowledge from data. Artificial intelligence algorithms often act in a role that supports expert human analysts. Data visualization techniques [5] are often a key part of EDA.
- Data mining (DM) [9] uses one or more artificial intelligence techniques as the primary analyzing mechanism. These algorithms are used as "black-box" extractors of knowledge from data sets, and usually produce rule sets.

Several approaches have been proposed for attaching descriptive (usually textual) labels to the neurons making up a SOM. Neuron labels are an important

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 45–54.
DOI: 10.1007/978-3-642-35230-0_5

part of many SOM-based EDA methods and all DM algorithms [24]. The most common labeling methods are supervised [13,20], relying on classification information in a labeling data set. However, since classification information is often unavailable, or may bias the labeling if it is available, unsupervised labeling (which requires no data classifications) has been investigated [4,6,19,21].

This paper presents a novel unsupervised labeling method, called unsupervised weight-based cluster labeling. The algorithm selects statistically significant weights for emergent neuron clusters, constructs sub-labels by linking values with these weights, and labels each cluster's neurons using the sub-labels.

The remainder of this paper is organized as follows: Section 2 gives a brief overview of SOMs. Section 3 overviews neuron labeling methods for SOMs. Section 4 describes the novel unsupervised weight-based cluster labeling algorithm, and critically discusses the method. Section 5 presents a prototype of the algorithm, as well as some visualizations of sample results. Finally, Sect. 6 presents this paper's conclusions, and outlines possible related work for the future.

2 Self-Organizing Maps

The SOM is an unsupervised neural network, developed by Teuvo Kohonen in 1982 [12]. The approach's inspiration was the self-organizing nature of the human cerebral cortex and associative memory. SOMs are unsupervised because no training data classification information is explicitly used during training.

Fig. 1 (a) illustrates the basic architecture of a SOM. The map's training data set, denoted as $\mathcal{D}_T = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{P_T}\}$, holds P_T training examples. Each training vector, $\boldsymbol{z}_p = (z_{p1}, z_{p2}, \ldots, z_{pI})$, consists of I attribute values, such that each $z_{pi} \in \mathbb{R}$. The SOM has a $K \times J$ neuron grid, where each neuron at row k and column j has an I-dimensional weight vector, $\boldsymbol{w}_{kj} = (w_{kj1}, w_{kj2}, \ldots, w_{kjI})$. Each weight $w_{kji} \in \mathbb{R}$, and correlates to attribute i in $\mathcal{D}_T$. Each weight vector acts as a model representation of a subset of the training vectors in $\mathcal{D}_T$.

After defining a SOM's initial structure, the weight vector values must be adjusted, by means of a *training algorithm*. The objective of any SOM training algorithm is to update the position of each neuron's weight vector, so that:

- Weight vectors "drift" towards denser groupings of data examples, causing the SOM to model the probability density function of the input space.
- The map is topologically structured, since input examples that are close to one another in the input space are also close to each other on the map.

Several algorithms exist to optimize the weights of a SOM's map structure, including the original stochastic training algorithm [12], and a batch training variant [14]. Since unsupervised weight-based cluster labeling is applicable to any SOM that has the aforementioned architecture and mapping characteristics, the exact nature of the training algorithm is unimportant to this discussion.

Fig. 1 (b) illustrates the result of training on a small part of a hypothetical SOM trained on two-dimensional data. Gray circles paired with dashed lines, and black circles paired with solid lines, show the weight vector positions and neighboring neuron connections upon initialization and after training, respectively.

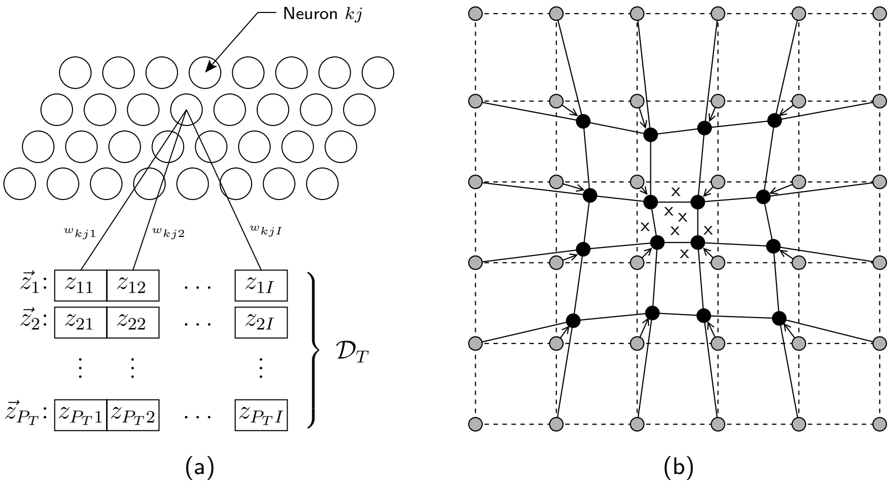

Neuron kj
w_{kj1}
w_{kj2}
w_{kjI}
$\vec{z}_1$: z_{11} z_{12} … z_{1I}
$\vec{z}_2$: z_{21} z_{22} … z_{2I}
$\vec{z}_{P_T}$: z_{P_T1} z_{P_T2} … z_{P_TI}
$\mathcal{D}_T$
(a)
(b)

Fig. 1. The basic structure and operation of a SOM: (a) shows the SOM's architecture; (b) shows the local effect of map training in a hypothetical two-dimensional case

Crosses denote the input space positions of training data vectors. The weight vectors of the neurons tend to cluster around dense areas within the input data, while similar training vectors are represented by neighboring neurons.

3 Neuron Labeling for Self-Organizing Maps

The process of neuron labeling entails the association of descriptive labels, which are typically textual in nature, with map neurons. Neuron labeling is important, since such labels are required by many EDA methods, and most DM algorithms (such as the SIG* algorithm [23] and the HybridSOM framework [25]).

Two general categories of neuron labeling approaches [24] can be identified, both of which are also illustrated in terms of broad overviews within Fig. 2:

- *Supervised labeling* refers to a family of techniques that derive labels for neurons from classification information that is associated with each example in a labeling data set. These methods either map labeling examples to the SOM structure [13,20], or map neurons to labeling examples [13].
- *Unsupervised labeling* uses no data classification information to build labels. These techniques either guide a human analyst [6] in manually assigning labels, derive labels directly from the map structure [21], or use an unclassified data set and the map structure to build labels [4,19].

The proposed algorithm falls into the latter category, and supervised labeling is thus disregarded. The proposed method, and the other unsupervised techniques this section discusses, apply sub-labels to neurons using no human assistance.

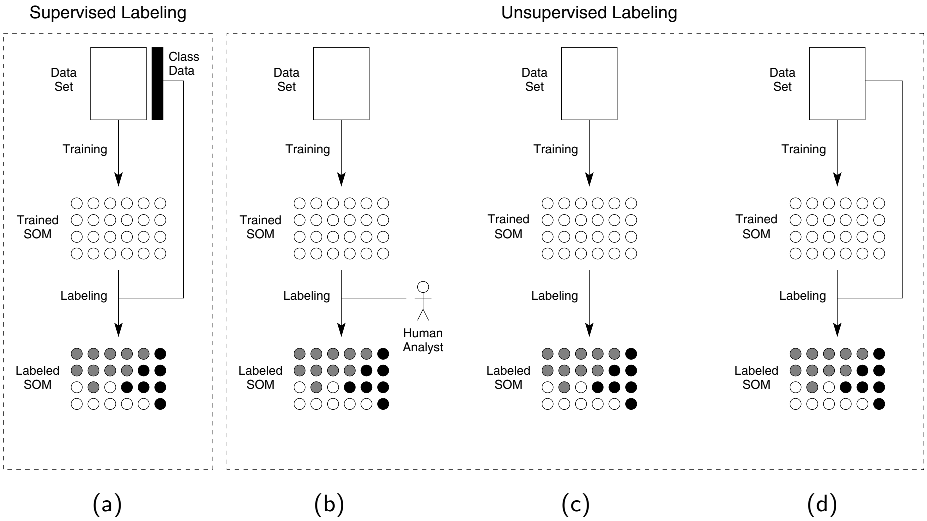


Fig. 2. Supervised and unsupervised neuron labeling methods: (a) shows supervised labeling; (b) shows exploratory labeling; (c) shows unsupervised labeling using map weights; (d) shows unsupervised labeling using map weights and unclassified data

Serrano-Cinca [21] proposes forming each neuron's sub-labels from attribute-weight pairs, which are chosen based on the neuron's weights. Labels are guaranteed for all neurons, but are unlikely to be exactly the same for similar neurons. This non-uniform labeling may be difficult for human analysts to interpret.

LabelSOM [19] maps each example in a labeling set to the example's closest matching neuron. Based on each neuron's mapped examples, attributes are chosen and combined with the attribute's mean value over the mapped examples, to form sub-labels. Neurons with no mapped examples are not labeled, which becomes problematic when labeling sets are sparse and maps are large.

Azcarraga et al. [4] propose a method similar to LabelSOM, but label clusters of neurons uniformly using attribute-value pairs selected from labeling examples that map to these clusters. The technique largely overcomes LabelSOM's unlabeled neuron problem. The method's uniformly defined neuron groups are often well-suited to EDA, and are required by many SOM-based DM algorithms.

If a very small labeling data set is used, both LabelSOM and the approach of Azcarraga et al. are also likely to produce inaccurate labels. In such a case, attributes will be selected from very small sets of mapped data examples, which are less likely to be statistically representative of data subset characteristics.

4 Unsupervised Weight-Based Cluster Labeling

This section describes the novel unsupervised weight-based cluster labeling algorithm, which falls into the category depicted in Fig. 2 (c), and critically discusses the feasibility of the approach within the context of practical EDA or DM.

4.1 The Labeling Algorithm

This section describes the steps performed by weight-based cluster labeling. Fig. 3 presents the algorithm's pseudocode, which is inspired by aspects of the unsupervised labeling methods of Azcarraga et al. [4] and Serrano-Cinca [21].

Step 1: Discover Emergent Clusters: Unsupervised weight-based cluster labeling requires the discovery of a set, $\mathcal{C} = \{S_1, S_2, \ldots, S_m\}$, of m meta-level clusters of weight vectors, called emergent clusters. A cluster contains weight vectors with similar characteristics, which represent similar data examples.

Many emergent weight vector discovery methods are available, including exploratory methods [26] and algorithms such as SOM-Ward [16] and k-means [15] clustering. The specifics of these techniques are not this paper's focus.

Step 2: Compute Significance Measures: Significance values are computed for each cluster's attributes. The significance value, $sig(A_l, S_i)$, is the significance of attribute A_l in cluster S_i, calculated using the weight vectors in S_i.

A variety of sensible $sig(A_l, S_i)$ measures are possible. This work identifies only three possible measures. All of the measures require attributes that are normalized to the same range (either training set attributes are normalized prior to training, or weights are normalized after training), and categorical attributes that are binary encoded into several separate continuous attributes:

- Absolute weight value significance is similar to a statistic used by Serrano-Cinca's method [21], and assumes that very large and very small weight values denote important attributes. The measure requires that all the map's weight values share a range centered on zero, and is computed as follows:

$$sig(A_l, S_i) = |mean(w_{kjl}, S_i)| \ , \tag{1}$$

  where $mean(w_{kjl}, S_i)$ denotes the mean of weight l over S_i. A high $sig(A_l, S_i)$ value indicates that attribute A_l is considered to be more significant.
- Variance-based significance gives higher value to low variance weights, since these values are very characteristic within a cluster, and is defined as:

$$sig(A_l, S_i) = -\left(\frac{1}{o_i - 1} \cdot \sum_{\boldsymbol{w}_{kj} \in S_i} \big(w_{kjl} - mean(w_{kjl}, S_i)\big)^2 \right) \ , \tag{2}$$

  where o_i is the number of constituent weight vectors that make up S_i. A higher $sig(A_l, S_i)$ value indicates an attribute with higher significance.
- A measure based on the Kolmogorov-Smirnov (K-S) statistic is possible. The set, $out(S_i)$, is the union of all emergent clusters on the map, excluding cluster S_i. The K-S statistic [11] computes to what degree the cumulative distributions of w_{kjl} over S_i and $out(S_i)$ differ. Higher values denote distributions that differ more, which indicate attributes in S_i that are significant [2].

These types of significance measures have previously been applied outside the broader domain of neuron labeling that is discussed in this paper, for applications such as the construction of rules to describe emergent neuron clusters [22,23].

```
Create and initialize a SOM, denoted $map$, consisting of $K \times J$ neurons
Train $map$ on an $I$-attribute training set, denoted $\mathcal{D}_T$, until convergence
Derive a discrete set of clusters, $\mathcal{C} = \{S_1, S_2, \ldots, S_m\}$, of all $\boldsymbol{w}_{kj} \in map$
for all clusters $S_i \in \mathcal{C}$ do
    for all attributes $A_l$ represented by a weight in $\boldsymbol{w}_{kj}$ do
        Use weight $w_{kjl}$ for all $\boldsymbol{w}_{kj} \in S_i$, to compute a significance value
        Associate the computed significance value with $A_l$ in $S_i$
    end for
end for
for all clusters $S_i \in \mathcal{C}$ do
    for all sufficiently significant attributes $A_l$ with corresponding $w_{kjl}$ do
        Build a sub-label using the name of $A_l$ and value of $w_{kjl}$ over $S_i$
        Add the new sub-label to the label of each $n_{kj} \in S_i$
    end for
end for
```

Fig. 3. Pseudocode of the unsupervised weight-based cluster labeling algorithm

Step 3: Select Descriptive Attributes: Using $sig(A_l, S_i)$, a subset of attribute names are chosen as cluster sub-labels. Too many sub-labels are complex and unreadable, while too few reduce label accuracy (a problem common to all unsupervised labeling algorithms). Four selection methods are possible:

- Selecting only the n most significant attributes for each cluster. This technique is simple, but likely to produce rough and inaccurate labels.
- Choosing attributes with a pre-defined minimum significance. To label all neurons, selection of the most significant attribute can be enforced.
- Choosing attributes that are significant relative to other attributes in the same cluster by, for example, selecting attributes with $sig(A_l, S_i)$ values of more than a standard deviation from the mean of all $sig(A_l, S_i)$ in S_i.
- Selecting attributes with significances that are distinctive over the whole map, for instance, choosing attributes with $sig(A_l, S_i)$ values of more than a standard deviation from the mean significance of A_l over all clusters.

Step 4: Associating Values with Weights: Attribute names alone are usually insufficiently descriptive sub-labels. Values representative of chosen attributes in each cluster are thus usually added to sub-labels. Simple value choices for A_l in S_i are the mean or median of all $w_{kjl} \in S_i$. If label values are normalized, these values should be de-normalized to the original data's ranges.

To facilitate easy analysis, it may be desirable to replace raw attribute values with wider ranges, such as "high" or "low" [21]. The selection of good thresholds for such ranges is important, and can make use of data binning techniques [8].

Step 5: Label Neurons: Finally, the algorithm applies labels to neurons. For each emergent cluster, the labels built in step 4 are used to label every neuron within the cluster. Labels are typically listed in decreasing order of significance.

4.2 A Critical Discussion

The proposed algorithm has important advantages in the following situations:

- Since clusters are labeled uniformly, the proposed algorithm is preferred to Serrano-Cinca's [21] method when characterizing broad data classes. This is often useful for EDA, and required by many SOM-based DM algorithms.
- The proposed method is useful when labeling data is scarce, since only the map's weight vectors are used. Lack of data adversely affects the accuracy of labels produced by either the method of Azcarraga et al. [4] or LabelSOM.

However, certain factors may have a negative impact on performance:

- Label quality depends on map quality, and a map's quality can be difficult to assess. The clustering method, significance measure, and attribute selection method must also be well chosen. This decision is often not obvious.
- The algorithm is more time complex than both Serrano-Cinca's method and LabelSOM, because an initial cluster discovery step is required. This effect is especially detrimental to performance when labeling very large maps.
- Inseparable clusters represent two or more classes [24]. All the neurons in an inseparable cluster will be labeled uniformly, while Serrano-Cinca's method and LabelSOM might be able to distinguish between inseparable classes.

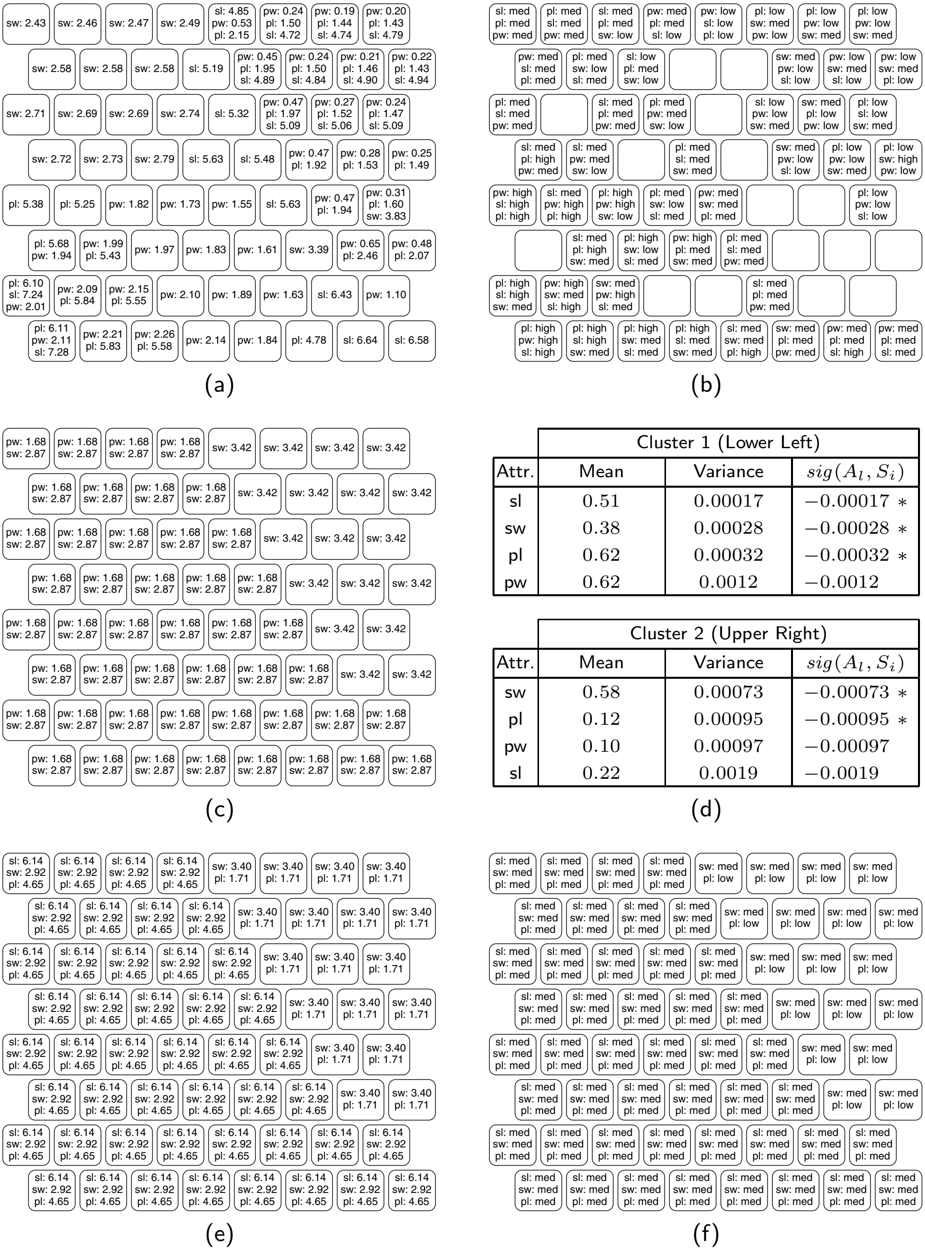
5 Algorithm Prototype and Results

Since unsupervised label quality is subjective, empirical analysis is difficult. Rather than use performance measures, labelings are usually qualitatively analyzed [19,21], or behavior similar to other statistical or rule extraction methods is assumed to show good performance [4]. This section thus focuses on comparative examples of results, but defers a detailed cross-analysis to future work.

The examples use the Iris data set from the UCI Machine Learning Repository [3]. The data set contains 150 examples, divided equally between three classes (Iris setosa, Iris versicolor, and Iris virginica). Each example is described by four continuous attribute values (sepal length, sepal width, petal length, and petal width), which were all scaled to the same range before SOM training.

One SOM, trained using the original stochastic algorithm [12], was used for all examples. The cluster-based methods used SOM-Ward clustering [16], which found two clusters. A cluster in the map's upper right represents the Iris setosa class, while the rest of the map is an inseparable cluster for Iris versicolor and virginica. The latter cluster illustrates several technique characteristics that are this section's focus, and justifies the data set's use. Future work will analyze other real-world data sets. Fig. 4 visualizes the labelings, where sl, sw, pl, and pw respectively represent the attributes sepal length, sepal width, petal length, and petal width. Sub-label values were re-scaled to the training set's ranges.

Fig. 4 (a) shows the outcome of Serrano-Cinca's method on the example SOM. Selected attributes all have a significance value of more than 50%. Each neuron is uniquely labeled, which may confuse a data analyst and is useless for DM


sw: 2.43 | sw: 2.46 | sw: 2.47 | sw: 2.49 | sl: 4.85 pw: 0.53 pl: 2.15 | pw: 0.24 pl: 1.50 sl: 4.72 | pw: 0.19 pl: 1.44 sl: 4.74 | pw: 0.20 pl: 1.43 sl: 4.79
sw: 2.58 | sw: 2.58 | sw: 2.58 | sl: 5.19 | pw: 0.45 pl: 1.95 sl: 4.89 | pw: 0.24 pl: 1.50 sl: 4.84 | pw: 0.21 pl: 1.46 sl: 4.90 | pw: 0.22 pl: 1.43 sl: 4.94
sw: 2.71 | sw: 2.69 | sw: 2.69 | sw: 2.74 | sl: 5.32 | pw: 0.47 pl: 1.97 sl: 5.09 | pw: 0.27 pl: 1.52 sl: 5.06 | pw: 0.24 pl: 1.47 sl: 5.09
sw: 2.72 | sw: 2.73 | sw: 2.79 | sl: 5.63 | sl: 5.48 | pw: 0.47 pl: 1.92 | pw: 0.28 pl: 1.53 | pw: 0.25 pl: 1.49
pl: 5.38 | pl: 5.25 | pw: 1.82 | pw: 1.73 | pw: 1.55 | sl: 5.63 | pw: 0.47 pl: 1.94 | pw: 0.31 pl: 1.60 sw: 3.83
pl: 5.68 pw: 1.94 | pw: 1.99 pl: 5.43 | pw: 1.97 | pw: 1.83 | pw: 1.61 | sw: 3.39 | pw: 0.65 pl: 2.46 | pw: 0.48 pl: 2.07
pl: 6.10 sl: 7.24 pw: 2.01 | pw: 2.09 pl: 5.84 | pw: 2.15 pl: 5.55 | pw: 2.10 | pw: 1.89 | pw: 1.63 | sl: 6.43 | pw: 1.10
pl: 6.11 pw: 2.11 sl: 7.28 | pw: 2.21 pl: 5.83 | pw: 2.26 pl: 5.58 | pw: 2.14 | pw: 1.84 | pl: 4.78 | sl: 6.64 | sl: 6.58
(a)
sl: med pl: med pw: med | pl: med sl: med pw: med | sl: med pl: med sw: low | pw: med pl: med sl: low | pw: low sl: low pl: low | pl: low sw: med pw: low | pl: low pw: low sw: med | pl: low pw: low sw: med
pw: med sl: med pl: med | pl: med sw: low sl: med | sl: low pl: med sw: low | | | sw: med pw: low sl: low | pw: low sw: med sl: low | pw: low sw: med pl: low
pl: med sl: med pw: med | | sl: med pl: med pw: med | pl: med pw: med sw: low | | sl: low pw: low sw: med | sw: med pl: low pw: low | pl: low sl: low sw: med
sl: med pl: high pw: med | sl: med pw: med sw: low | | pl: med sl: med sw: med | | sw: med pw: low sl: low | pl: low pw: low sw: med | pl: low sw: high pw: low
pw: high sl: high pl: high | sl: med pw: high pl: high | pl: high pw: high sw: low | pl: med sw: low sl: med | pw: med sw: med pl: med | | | pl: low pw: low sl: low
| sl: med pl: high sw: med | pl: high sw: low sl: med | pw: high pl: med sw: med | pl: med sl: med pw: med | | |
pl: high sl: high sw: med | pw: high sw: med sl: high | sw: med pw: high sl: med | | | sl: med pl: med pw: med | |
pl: high pw: high sl: high | pl: high sl: high sw: med | pl: high sw: med sl: med | pl: high sl: med sw: med | sl: med sw: med pl: high | sw: med pl: med pw: med | pw: med pl: med sl: high | pw: med pl: med sl: med
(b)
pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | sw: 3.42 | sw: 3.42 | sw: 3.42 | sw: 3.42
pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | sw: 3.42 | sw: 3.42 | sw: 3.42 | sw: 3.42
pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | sw: 3.42 | sw: 3.42 | sw: 3.42
pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | sw: 3.42 | sw: 3.42 | sw: 3.42
pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | sw: 3.42 | sw: 3.42
pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | sw: 3.42 | sw: 3.42
pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87
pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87 | pw: 1.68 sw: 2.87
(c)
(d)
sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71
sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71
sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71
sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71
sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71
sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sw: 3.40 pl: 1.71 | sw: 3.40 pl: 1.71
sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65
sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65 | sl: 6.14 sw: 2.92 pl: 4.65
(e)
sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sw: med pl: low | sw: med pl: low | sw: med pl: low | sw: med pl: low
sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sw: med pl: low | sw: med pl: low | sw: med pl: low | sw: med pl: low
sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sw: med pl: low | sw: med pl: low | sw: med pl: low
sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sw: med pl: low | sw: med pl: low | sw: med pl: low
sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sw: med pl: low | sw: med pl: low
sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sw: med pl: low | sw: med pl: low
sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med
sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med | sl: med sw: med pl: med
(f)

	Cluster 1 (Lower Left)		
Attr.	Mean	Variance	$sig(A_l, S_i)$
sl	0.51	0.00017	-0.00017 *
sw	0.38	0.00028	-0.00028 *
pl	0.62	0.00032	-0.00032 *
pw	0.62	0.0012	-0.0012

	Cluster 2 (Upper Right)		
Attr.	Mean	Variance	$sig(A_l, S_i)$
sw	0.58	0.00073	-0.00073 *
pl	0.12	0.00095	-0.00095 *
pw	0.10	0.00097	-0.00097
sl	0.22	0.0019	-0.0019

Fig. 4. Neuron labeling of a SOM trained on the Iris data set: (a) uses Serrano-Cinca's method, where values are neuron weights; (b) uses LabelSOM, where values are attribute means for mapped examples; (c) uses Azcarraga's method, where values are attribute means for mapped examples; (d) shows weight-based cluster labeling statistics; (e) uses weight-based cluster labeling, with variance-based significance and mean weight values over clusters; (f) uses weight-based cluster labeling with threshold values.

algorithms that require broad classes. The neurons constituting the inseparable cluster are, however, somewhat differentiated from one another.

Fig. 4 (b) shows LabelSOM's results, where the entire training set was used for labeling. Sub-labels are the three most significant attributes per neuron. Of the map's neurons, 25% are unlabeled, 18.75% have only one mapped example, and an average of only 3.125 examples map to each neuron. The statistical soundness of the sub-labels is thus clearly questionable. Neurons have unique labels, with similar advantages and drawbacks to Serrano-Cinca's method.

Fig. 4 (c) shows results for Azcarraga's method, using all training data. Attributes with significances more than a standard deviation from a cluster's mean significance are ordered by decreasing absolute significance. LabelSOM's sparse example mapping problem is overcome (small clusters will cause the same issue), but the inseparable cluster is undifferentiated. Larger areas of uniform labels often ease human interpretation and are suitable for many DM methods.

Fig. 4 (d) shows statistics used for weight-based cluster labeling. Equation (2) was used for $sig(A_l, S_i)$. Attributes with a minimum $sig(A_l, S_i)$ of -0.00095 are marked as sub-labels, using asterisks. Sub-label values are attribute means.

Fig. 4 (e) shows weight-based cluster labeling, using the computed statistics. Like Azcarraga's method, large areas are labeled uniformly, while inseparable clusters are undifferentiated. However, no labeling data is relied upon.

Fig. 4 (f) shows the same labeling as Fig. 4 (e) does, but uses thresholds instead of raw attribute values. The values `low`, `med` and `high` respectively denote mean values in the lower, middle and upper third of each attribute range. Threshold values are easier to interpret than raw values, at the expense of detail.

6 Conclusions and Future Work

This paper presented an overview of SOM neuron labeling. A novel unsupervised weight-based cluster labeling algorithm, which labels neuron clusters using significant attributes and values (thus labeling all map neurons), was proposed and critiqued. A prototype was presented, with visualizations of some results.

Future work will comparatively analyze the proposed method on several more complex, higher-dimensional data sets. An analysis of the method's time complexity and scalability is also planned. The authors hope to develop a method to empirically evaluate the algorithm's label quality, allowing analyses of the performances of different significance measures and attribute selection schemes.

References

1. Alhoniemi, E.: Analysis of pulping data using the self-organizing map. Tappi Journal 83(7), 66–75 (2000)
2. Alhoniemi, E., Simula, O.: Interpretation and comparison of multidimensional data partitions. In: Proceedings of ESANN, pp. 277–282 (2001)
3. Asuncion, A., Frank, A.: UCI repository of machine learning databases. University of California, Irvine (2010), http://archive.ics.uci.edu/ml

4. Azcarraga, A., Hsieh, M.H., Pan, S.L., Setiono, R.: Improved SOM labeling methodology for data mining applications. In: Soft Computing for Knowledge Discovery and Data Mining, pp. 45–75. Springer (2008)
5. Card, S.K., Mackinlay, J.D., Shneiderman, B. (eds.): Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann (1999)
6. Corradini, A., Gross, H.M.: A hybrid stochastic-connectionist architecture for gesture recognition. In: Proceedings of ICIIS, pp. 336–341 (1999)
7. Deboeck, G., Kohonen, T. (eds.): Visual Explorations in Finance with Self-Organizing Maps. Springer (1998)
8. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann (2012)
9. Holsheimer, M., Siebes, A.P.J.M.: Data mining: The search for knowledge in databases. Tech. Rep. CS-R9406, Centrum voor Wiskunde en Informatica (1994)
10. Kaski, S., Kangas, J., Kohonen, T.: Bibliography of self-organizing map (SOM) papers: 1981–1997. Neural Computing Surveys 1, 102–350 (1998)
11. Knuth, D.E.: The Art of Computer Programming, 3rd edn., vol. 2, pp. 48–58. Addison-Wesley (1997)
12. Kohonen, T.: Self-organizing formation of topologically correct feature maps. Biological Cybernetics 43(1), 59–69 (1982)
13. Kohonen, T.: Self-Organization and Associative Memory, 3rd edn. Springer (1989)
14. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer (2001)
15. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
16. Murtagh, F.: Interpreting the Kohonen self-organizing feature map using contiguity-constrained clustering. Pattern Recognition Letters 16(4), 399–408 (1995)
17. Oja, M., Kaski, S., Kohonen, T.: Bibliography of self-organizing map (SOM) papers: 1998–2001 addendum. Neural Computing Surveys 3, 1–156 (2003)
18. Pöllä, M., Honkela, T., Kohonen, T.: Bibliography of self-organizing map (SOM) papers: 2002-2005 addendum. Tech. Rep. TKK-ICS-R23, Helsinki University of Technology (2009)
19. Rauber, A., Merkl, D.: Automatic Labeling of Self-Organizing Maps: Making a Treasure-Map Reveal Its Secrets. In: Zhong, N., Zhou, L. (eds.) PAKDD 1999. LNCS (LNAI), vol. 1574, pp. 228–237. Springer, Heidelberg (1999)
20. Samarasinghe, S.: Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition. Auerbach Publications (2007)
21. Serrano-Cinca, C.: Self organizing neural networks for financial diagnosis. Decision Support Systems 17(3), 227–238 (1996)
22. Siponen, M., Vesanto, J., Simula, O., Vasara, P.: An approach to automated interpretation of SOM. In: Proceedings of WSOM (2001)
23. Ultsch, A.: Konnektionistische Modelle und ihre Integration mit wissensbasierten Systemen. Report 396, University of Dortmund (1991)
24. Van Heerden, W.S., Engelbrecht, A.P.: A comparison of map neuron labeling approaches for unsupervised self-organizing feature maps. In: Proceedings of IJCNN, pp. 2140–2147 (2008)
25. Van Heerden, W.S., Engelbrecht, A.P.: HybridSOM: A generic rule extraction framework for self-organizing feature maps. In: Proceedings of CIDM, pp. 17–24 (2009)
26. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11(3), 586–600 (2000)

Controlling Self-Organization and Handling Missing Values in SOM and GTM

Tommi Vatanen, Ilari T. Nieminen, Timo Honkela,
Tapani Raiko, and Krista Lagus

Aalto University School of Science,
Department of Information and Computer Science,
P.O. Box 15400, FI-00076 Aalto, Espoo, Finland
first.last@aalto.fi

Abstract. In this paper, we study fundamental properties of the Self-Organizing Map (SOM) and the Generative Topographic Mapping (GTM), ramifications of the initialization of the algorithms and properties of the algorithms in presence of missing data. We show that the commonly used principal component analysis (PCA) initialization of the GTM does not guarantee good learning results with complex, high-dimensional data. We propose initializing the GTM with SOM and demonstrate usefulness of this improvement using the ISOLET data set. We also propose a revision to the batch SOM algorithm called the Imputation SOM and show that the new algorithm is more robust in presence of missing data. We compare the performance of the algorithms in the missing value imputation task. We also announce a revised version of the SOM Toolbox for Matlab with added GTM functionality.

1 Introduction

Topographic mappings, such as the Self-Organizing Map (SOM) [15,16] and the Generative Topographic Mapping (GTM) [2], are useful tools in inspecting and visualizing high-dimensional data. The SOM was originally inspired by neuroscientific research on cortical organization and the algorithm models the basic principles of the organization process at a general level. In practice, SOM has proved to be a robust approach tested in thousands of different applications. The GTM was inspired by the SOM algorithm, while operating in the probabilistic framework which provides well-founded regularization and model comparison. In this paper, we show that the both methods have their own strengths over the other and the methods may even benefit each other.

This paper is organized as follows. Sections 2 and 3 introduce the SOM and the GTM models, respectively. In Section 4, self-organization and convergence of the algorithms are discussed and using the SOM for initializing the GTM is shown to improve the learning results. Section 5 explains the treatment of missing values in the GTM and adapts the same principled way into SOM. Performance of the algorithms is compared in a missing value imputation task. Finally, the results and prossible future work are discussed in Section 6.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 55–64.
DOI: 10.1007/978-3-642-35230-0_6

In all experiments, SOM Toolbox [23] and Netlab [1] software packages are used. The GTM scripts in Netlab are revised to handle data with missing values and sequential training algorithm is contributed. Finally, we announce a revised version of the SOM Toolbox which incorporates GTM functionality. An up-to-date version of the SOM Toolbox is available at

`http://research.ics.aalto.fi/software/somtoolbox`

2 Self-Organizing Map

The Self-Organizing Map (SOM) [16] discovers some underlying structure in data using K map units, prototypes or reference vectors $\{\boldsymbol{m}_i\}$. For the prototypes, explicit neighborhood relations have been defined. The classical sequential SOM algorithm proceeds by processing one data point $\boldsymbol{x}(t)$ at a time. Euclidean, or any other suitable distance measure is used to find the best-matching unit given by $\boldsymbol{m}_{c(\boldsymbol{x}(t))} = \arg\min_i \|\boldsymbol{x}(t) - \boldsymbol{m}_i\|$. The reference vectors are then updated using the update rule $\boldsymbol{m}_i(t+1) = \boldsymbol{m}_i(t) + h_{ci}(t)\left(\boldsymbol{x}(t) - \boldsymbol{m}_i(t)\right)$, where an explicit neighborhood function $h_{ci} = \alpha(t) \cdot \exp\left\{-\|r_c - r_i\|^2/2\sigma^2(t)\right\}$ is used in order to obtain topological mapping. In the neighborhood function, $\|r_c - r_i\|$ is the distance between the best-matching unit r_c and unit i in the array, $0 < \alpha(t) < 1$ is scalar-valued learning-rate factor and $\sigma(t)$ is the width of the neighborhood kernel.

2.1 Batch SOM

In the Batch SOM, the reference vectors are updated using all data (or a mini-batch, a part of the data) at once and weighted accordingly. The batch update rule is

$$\boldsymbol{m}_i = \frac{\sum_n h_{ni} \boldsymbol{x}_n}{\sum_j h_{ni}}, \tag{1}$$

where the index n runs over the data vectors whose best-matching units satisfy $h_{ni} > 0$, that is, all data points up to the range of the neighborhood function are taken into account.

2.2 Quality and Size of SOM

Selecting the size of the array of map units in the SOM is a subtle task. Previously many solutions, such as hierarchical [17] and growing maps [10,5], have been proposed to tackle this issue. The question of the size can be approached from the point of view of different quality measures. Two most commonly used error measures are the *quantization error* and the *topological error* [16]. The former measures the mean of the reconstruction errors $\|\boldsymbol{x} - \boldsymbol{m}_c\|$ when each data point used in learning is replaced by its best-matching unit. The latter measures the proportion of data points for which the two nearest map units are not neighbors in the array topology. As the number of map units increases, quantization error decreases and topological error tends to increase. Hence, there is no straightforward way of choosing the number of map units based on the measures above. Topographic preservation has been studied in detail, e.g., in [24,22,26]. In this work, we use an error measure proposed in [13]. This *combined error* is a sum of the quantization error and the distance from the best-matching unit to the second-best-matching unit of each data vector along the shortest path following the neighborhood relations. We have added this feature in the SOM Toolbox `som_quality` function and demonstrate its use in the experiments.

3 Generative Topographic Mapping

The Generative Topographic Mapping (GTM) [2,3] is a nonlinear latent variable model which was proposed as a probabilistic alternative to the SOM. Loosely speaking, it extends the SOM in a similar manner as Gaussian mixture model extends k-means clustering. This is achieved by working in a probabilistic framework where data vectors have posterior probabilities given a map unit. Hence, instead of possessing only one best-matching unit, each data vector contributes to many reference vectors directly.

The GTM can be seen consisting of three parts: 1) discrete set of points in usually one or two-dimensional latent space, 2) nonlinear mapping, usually radial basis function (RBF) network, between the latent space and the data space, and 3) a Gaussian noise model in the data space such that the resulting model is a constrained mixture of Gaussians. In this paper, latent points $\{\boldsymbol{u}_i\}$, which are arranged in a regular grid, are mapped to the data space using M fixed radial basis functions $\boldsymbol{\phi}(\boldsymbol{u}_i) = \{\phi_j(\boldsymbol{u}_i)\}$, where $\phi_j(\boldsymbol{u}_i) = \exp\left\{-\|\boldsymbol{c}_j - \boldsymbol{u}_i\|/\sigma^2\right\}$, σ is the width parameter of the RBFs, $\{\boldsymbol{c}_j\}$ are the RBF centers and $j = 1, \ldots, M$. The number of RBFs, M, is a free parameter which has to be chosen by the experimenter. The radius of the RBFs is chosen according to $\sigma = d_{\max}/\sqrt{M}$, where $d_{\max}$ is the maximum distance between two RBF centers (see, e.g. [11]). The node locations in latent space, $\boldsymbol{u}_i$, define a corresponding set of reference vectors $\boldsymbol{m}_i = \boldsymbol{W}\boldsymbol{\phi}(\boldsymbol{u}_i)$ in the data space, where $\boldsymbol{W}$ is a weight matrix defining the mapping from the latent space to the data space. In this work, each reference vector $\boldsymbol{m}_i$ serves as a center of an isotropic Gaussian distribution in the data space

$$p(\boldsymbol{x}|\boldsymbol{m}_i) = \left(\frac{\beta}{2\pi}\right)^{D/2} \exp\left\{-\frac{\beta}{2}\|\boldsymbol{m}_i - \boldsymbol{x}\|^2\right\}, \tag{2}$$

where β is the precision or inverse variance. The Gaussian distribution above also represents a noise model accounting for the fact that the data will not be confined precisely to the lower-dimensional manifold in the data space. More general noise models have been proposed [3].

The probability density function of the GTM is obtained by summing over the Gaussian components yielding

$$p(\boldsymbol{x}|\boldsymbol{W},\beta) = \sum_{i=1}^{K} P(\boldsymbol{m}_i)p(\boldsymbol{x}|\boldsymbol{m}_i) = \sum_{i=1}^{K} \frac{1}{K}\left(\frac{\beta}{2\pi}\right)^{D/2} \exp\left\{-\frac{\beta}{2}\|\boldsymbol{m}_i - \boldsymbol{x}\|^2\right\}, \quad (3)$$

where K is the total number grid points in the latent space, or map units in the SOM terminology, and the prior probabilities $P(\boldsymbol{m}_i)$ are given equal probabilities $1/K$.

The GTM represents a parametric probability density model, with parameters $\boldsymbol{W}$ and β, and it can be fitted to a data set $\{\boldsymbol{x}_n\}$ by maximum likelihood. The log-likelihood function of the GTM is given by

$$\log(\mathcal{L}(\boldsymbol{W},\beta)) = \sum_{n=1}^{N} \ln p(\boldsymbol{x}_n|\boldsymbol{W},\beta), \quad (4)$$

where $p(\boldsymbol{x}_n|\boldsymbol{W},\beta)$ is given by (3) and independent, identically distributed (iid) data is assumed. The log-likelihood can be maximized using the EM algorithm or alternatively any standard non-linear optimization techniques.

4 Self-Organization and Convergence

Both the GTM and the batch SOM require careful initialization in order to self-organize [14,8]. For both algorithms, the common choice is to initialize according to the plane spanned by the two main principal components of the data. In the batch SOM, the neighborhood is annealed during the learning which decreases the rigidness of the map. The most important advantages of the batch SOM when compared to classical sequential SOM are quick convergence and computational simplicity [8].

As we will show, initializing the GTM using PCA does not always lead to appropriate results. Instead, we propose using the batch SOM for initializing the GTM. In the SOM initialization, using few epochs of ’rough training’ with wide neighborhood will suffice. Next, $\boldsymbol{W}$ can be determined by minimizing the error function,

$$E = \frac{1}{2}\sum_{i}\|\boldsymbol{W}\boldsymbol{\phi}(\boldsymbol{u}_i) - \boldsymbol{m}_i^{\mathrm{SOM}}\|, \quad (5)$$

where $\boldsymbol{m}_i^{\mathrm{SOM}}$ are the reference vectors of the initializing SOM.

Efficacy of the SOM initialization is demonstrated using the ISOLET data set from the UCI machine learning repository [9]. The data contains 7797 spoken samples of the letters of the alphabet. The 617 features are described in [6] and include, e.g., spectral coefficients, contour features and sonorant features. The class labels, i.e., the letter identifiers, were not used in training of the maps.

The appropriate model complexity for the GTM, i.e, the number of RBFs and latent points, can be chosen, e.g., by cross-validating the negative log-likelihood. Using cross-validation for the ISOLET data, a suitable number of RBFs was found to be 400 (20×20) and a suitable number of map units 4004 ($77{\times}52$).

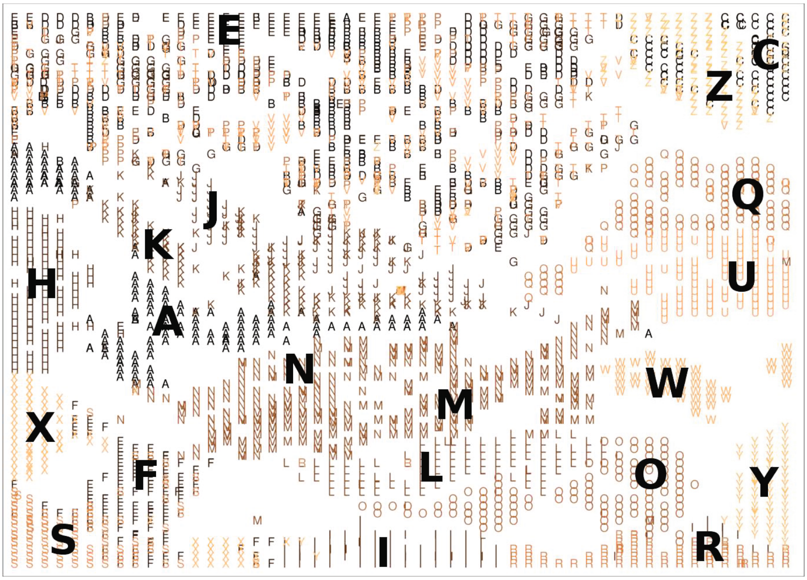


Fig. 1. SOM initialization: A GTM with 4004 (77×52) map units and 400 (20×20) RBFs trained using the ISOLET data. The GTM was initialized using SOM. Most of the letters of the alphabet are clustered as is shown with the manually added bold-face characters.

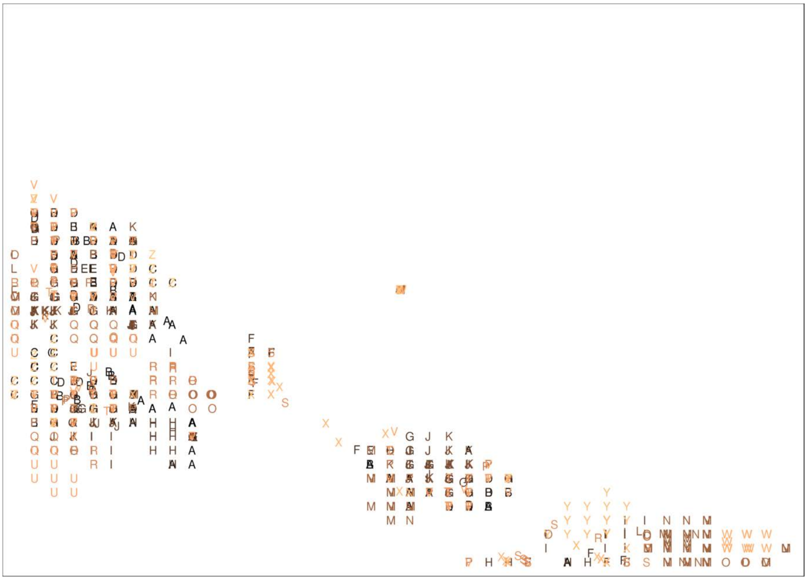


Fig. 2. PCA initialization: A GTM of the ISOLET data with 4004 map units and 400 RBFs. The map was initialized using PCA. The GTM fails finding reasonable structure in the data, that is, to self-organize.

Figures 1 and 2 show two GTM visualization of the ISOLET data. In Figure 1, the batch SOM initialization was used, whereas in Figure 2 the GTM was initialized using PCA. The map initialized using SOM has a clear cluster structure where most of the letters form distinct clusters. Furthermore, similar sounding letters are mapped close to each other. In the top center area and top left corner of the map, the data is more ambiguous and different letters, such as B, D, E, P, and V, are mixed together. Obviously, the GTM in Figure 2 does not provide useful representation of the ISOLET data. We did not manage to obtain learning results comparable to Figure 1 using the PCA initialization. The GTM with SOM initialization converges to higher log-likelihood-per-sample value (-573.2 vs. -675.1). Thus, the GTM seems to benefit from the SOM initialization when complex, high-dimensional data is used. The fact that relatively complex model with 400 RBFs and 4004 map units was required in order to obtain the mapping in Figure 1 suggests that the linear PCA initialization is too simple to allow the GTM to learn any interesting structure in the data.

5 Missing Values

In this section, we discuss the behavior of topographic mappings in presence of missing values. We start by showing how missing values are treated in the GTM and develop the same idea for the SOM. The section is concluded by an experimental study.

In all what follows, missing-at-random (MAR) data is assumed. This means that the probability of missingness is independent of missing values given the observed data. Even though this assumption can be questioned in many real-life scenarios, this is usually a reasonable assumption given that only a small proportion of the data is missing.

5.1 GTM and Missing Values

The GTM offers a robust framework for dealing with missing values, noted already in [2]. As with any method operating in the probabilistic framework, missing values can be handled by integrating them out. If the missing values are MAR, this does not introduce any bias. Hence, the maximum-likelihood estimation of the model parameters θ reduces to maximizing $\mathcal{L}(\theta|\boldsymbol{X}_{\mathrm{obs}}) = p(\boldsymbol{X}_{\mathrm{obs}}|\theta)$, where $\boldsymbol{X}_{\mathrm{obs}}$ denotes the observed data. For the GTM, the likelihood function is given by

$$\mathcal{L}(\boldsymbol{W}, \beta|\boldsymbol{X}_{\mathrm{obs}}) = p(\boldsymbol{X}_{\mathrm{obs}}|\boldsymbol{W}, \beta) = \int p(\boldsymbol{X}_{\mathrm{obs}}|\boldsymbol{X}_{\mathrm{mis}}, \boldsymbol{W}, \beta) d\boldsymbol{X}_{\mathrm{mis}}, \tag{6}$$

where $\boldsymbol{X}_{\mathrm{mis}}$ denotes the missing or unobserved data. This integration can be performed analytically for the standard GTM with an isotropic noise model.

The handling of missing data can be incorporated in the EM algorithm in a straightforward manner. In the E-step, where posterior probabilities of data vectors given the map units are calculated, missing values are simply omitted.

That is, the distance between the map units and a data vector with missing value(s) is evaluated only in the dimensions observed for the corresponding data vector. In the M-step, the expected values of the missing data and other sufficient statistics are used. The details of learning the GTM with missing values using the EM algorithm can be found in [21].

After the training, there are at least two possibilities to perform imputation in the GTM. One may use the expected values $\mathbb{E}(\boldsymbol{X}_{\text{mis}}|\boldsymbol{X}_{\text{obs}}, \boldsymbol{W}, \beta)$ or impute using the maximum-a-posteriori (MAP) estimates $p_{\text{MAP}}(\boldsymbol{X}_{\text{mis}}|\boldsymbol{X}_{\text{obs}}, \boldsymbol{W}, \beta)$ which takes the missing values from the most similar map unit. Additionally, multiple imputations can be conducted by sampling the posterior distribution $p(\boldsymbol{X}_{\text{mis}}|\boldsymbol{X}_{\text{obs}}, \boldsymbol{W}, \beta)$.

5.2 SOM and Missing Values

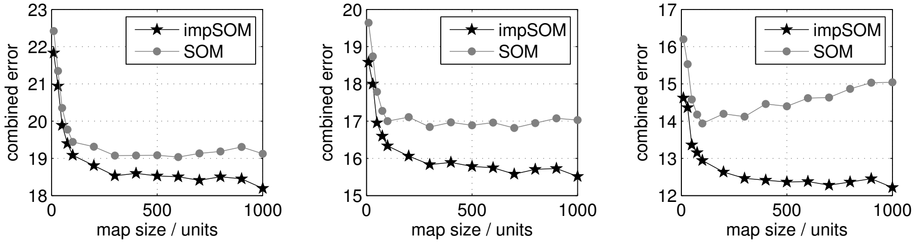
The SOM has been used for missing value imputation with many kinds of data, such as survey data [7,25], socio-economic data [4], industrial data [19,18] and climate data [20]. In most of the SOM literature, the missing values are treated as was proposed in [4]. The best-matching units for the data vectors with missing values are computed by omitting the missing values. This is consistent with the procedure in the probabilistic setting. The missing values are ignored also while updating the reference vectors. This approach is implemented in the widely used SOM Toolbox [23]. After the training, missing values can be filled according to the best-matching units of the corresponding data vectors.

Imputation SOM. A novel approach, named the *Imputation SOM (impSOM)*, stems from the way missing values are treated while using the GTM with an isotropic noise model (see above). The distances between data points and reference vectors are evaluated as described above, since this already corresponds to the statistical approach. While updating the reference vectors, instead of ignoring the missing data their expected values

$$\hat{\boldsymbol{x}}_{ni,\text{mis}} = \mathbb{E}\left[\boldsymbol{x}_{n,\text{mis}}|\boldsymbol{m}_i\right] = \boldsymbol{m}_i \tag{7}$$

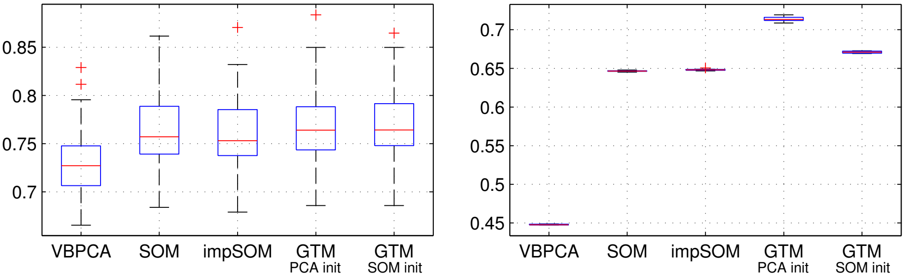
are used. Above, expectation is used in an informal sense, since the SOM is not a statistical model. This results in an update rule, where the reference vectors are updated according to (1) such that for each unobserved component of $\boldsymbol{x}_n$ the current value $\boldsymbol{m}_i$ is used. Thus, the data with missing values contribute by restraining the reference vectors in the dimensions corresponding to the missing values.

Figure 3 illustrates the difference between the Imputation SOM and the traditional way of treating missing values in SOM. The three subfigures plot the combined error (explained in Section 2.2) with respect to the number of map units with 30, 50 and 70 % of missing data. Clearly, the higher the proportion of missing data the larger the improvement of the Imputation SOM algorithm.



(a) 30 % missing data (b) 50 % missing data (c) 70 % missing data

Fig. 3. A comparison between the traditional SOM and Imputation SOM algorithms with 30, 50 and 70 % of MAR missing data in ISOLET data. The larger the missingness ratio the larger the difference between the algorithms.



(a) wine data (b) ISOLET data

Fig. 4. Box plots of RMS imputation errors of wine and ISOLET data sets using SOM, the Imputation SOM (impSOM), the GTM with PCA and SOM initialization and the Variational Bayesian PCA (VBPCA). Randomly generated data sets with 20 % missing data are used and the imputation is repeated 100 times for the wine data and 10 times for the ISOLET data.

5.3 Imputation Experiments

We compare the methods in the presence of missing data using two data sets: the wine and ISOLET data sets from the UCI machine learning repository [9]. Variational Bayesian PCA (VBPCA) [12][1] is used as a comparison method to evaluate the general usability of topographic mappings in missing value imputation tasks.

Figure 4 shows box plots of the imputations results. Randomly generated data sets with 20 % missing data were used and the imputation was repeated 100 times for the wine data and 10 times for the ISOLET data. The results are reported on normalized data. With 13-dimensional wine data, there are no significant differences in the RMS imputation errors between the topographic methods. The variance of the results is high due to only 150 samples which

[1] http://users.ics.tkk.fi/alexilin/software/

do not provide enough information for efficient imputation. The best results among the SOM and GTM based methods are obtained using the Imputation SOM. Better results may be obtained using VBPCA without a need of model selection. The results obtained using the GTM are slightly worse compared to the SOM and the initialization of the GTM does not effect the results.

The results for the more complex ISOLET data reveal more substantial differences between the methods. The results obtained using SOM are better compared to the GTM and it is obvious that the learning results of the GTM are poor if PCA initialization is used. When SOM initialization was used, the results were closer to SOM. However, comparison with VBPCA shows that topographic mappings are not particularly suitable for missing value imputation since using VBPCA provides significantly better results. This suggests that it might be beneficial to impute the data with any robust imputation method before the SOM or GTM visualization. In VBPCA, number of components is determined by automatic relevance determination and more than two components (latent dimensions) are used. Hence, the prediction of missing values becomes more accurate. A more comprehensive comparison of the imputation capabilities of SOM and GTM was conducted in [21].

6 Conclusions and Discussion

In this paper, we have studied topographic properties of the SOM and GTM and proposed initializing the GTM with SOM. We showed that SOM initialization enables learning complex and high-dimensional data with the GTM—a task that may fail using the conventional PCA initialization. We have also proposed a novel way of treating missing values in SOM training called the Imputation SOM and showed that this revision makes SOM more robust in terms of combined error when missing values are present.

In the future, it might be interesting to study whether the self-organization of the GTM benefits from sequential training. In our initial experiments, we have found that mini-batch training speeds up the convergence, as proposed by [3]. Additionally, the improvements developed to enhance the self-organization of the batch SOM may be applied for the GTM, as well. The number of RBFs, M, roughly corresponds to the width of the neighborhood function in the SOM. The smaller M, i.e. less RBFs, the more rigid the mapping. Thus, the effect of annealed neighborhood may be achieved by increasing the number of RBFs during the learning. It is also possible to use regularization, as was shown in [21], in order to control the rigidness of the GTM.

References

1. NETLAB: algorithms for pattern recognition. Springer-Verlag New York, Inc., New York (2002)
2. Bishop, C.M., Svensén, M., Williams, C.K.I.: GTM: The generative topographic mapping. Neural Computation 10 (1998)

3. Bishop, C.M., Williams, C.K.I.: Developments of the Generative Topographic Mapping. Neurocomputing 21, 203–224 (1998)
4. Cottrell, M., Letrémy, P.: Missing values: processing with the Kohonen algorithm. In: Applied Stochastic Models and Data Analysis, pp. 489–496 (2005)
5. Dittenbach, M., Merkl, D., Rauber, A.: The growing hierarchical self-organizing map. In: IJCNN, pp. 15–19 (2000)
6. Fanty, M.A., Cole, R.A.: Spoken letter recognition. In: NIPS, pp. 220–226 (1990)
7. Fessant, F., Midenet, S.: Self-organising map for data imputation and correction in surveys. Neural Computing and Applications 10(4), 300–310 (2002)
8. Fort, J.C., Letrémy, P., Cottrell, M.: Advantages and drawbacks of the Batch Kohonen algorithm. In: ESANN, pp. 223–230 (2002)
9. Frank, A., Asuncion, A.: UCI machine learning repository (2010), http://archive.ics.uci.edu/ml
10. Fritzke, B.: Grorwing cell structures–a self-organizing network for unsupervised and supervised learning. Neural Networks 7(9), 1441–1460 (1994)
11. Haykin, S.: Neural Networks and Learning Machines, 3rd edn. Prentice Hall (2008)
12. Ilin, A., Raiko, T.: Practical approaches to principal component analysis in the presence of missing values. Journal of Machine Learning Research 99, 1957–2000 (2010)
13. Kaski, S., Lagus, K.: Comparing Self-organizing Maps. In: Vorbrüggen, J.C., von Seelen, W., Sendhoff, B. (eds.) ICANN 1996. LNCS, vol. 1112, pp. 809–814. Springer, Heidelberg (1996)
14. Kiviluoto, K., Oja, E.: S-Map: A Network with a Simple Self-Organization Algorithm for Generative Topographic Mappings. In: Advances in Neural Information Processing Systems, pp. 549–555. Morgan Kaufmann Publishers (1998)
15. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics 43, 59–69 (1982)
16. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer-Verlag New York, Inc., Secaucus (2001)
17. Koikkalainen, P., Oja, E.: Self-organizing hierarchical feature maps. In: IJCNN 1990, pp. 279–285 (1990)
18. Merlin, P., Sorjamaa, A., Maillet, B., Lendasse, A.: X-SOM and L-SOM: a double classification approach for missing value imputation. Neurocomputing 73(7-9), 1103–1108 (2010)
19. Rustum, R., Adeloye, A.J.: Replacing outliers and missing values from activated sludge data using Kohonen Self-Organizing Map. Journal of Environmental Engineering 133(9), 909–916 (2007)
20. Sorjamaa, A.: Methodologies for Time Series Prediction and Missing Value Imputation. Ph.D. thesis, Aalto University School of Science and Technology (2010)
21. Vatanen, T.: Missing Value Imputation Using Subspace Methods with Applications on Survey Data. Master's thesis, Aalto University, Espoo, Finland (2012)
22. Venna, J., Kaski, S.: Local multidimensional scaling. Neural Networks 19(6-7), 889–899 (2006)
23. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: Self-organizing map in MATLAB: the SOM Toolbox. In: The Matlab DSP Conference, pp. 35–40 (2000)
24. Villmann, T., Der, R., Herrmann, J.M., Martinetz, T.: Topology preservation in self-organizing feature maps: exact definition and measurement. IEEE Trans. Neural Netw. Learning Syst. 8(2), 256–266 (1997)
25. Wang, S.: Application of Self-Organising Maps for data mining with incomplete data sets. Neural Computing and Applications 12, 42–48 (2003)
26. Zhang, L., Merényi, E.: Weighted differential topographic function: a refinement of topographic function. In: ESANN, pp. 13–18 (2006)

Opposite Maps for Hard Margin Support Vector Machines

Ajalmar R. da Rocha Neto[1] and Guilherme A. Barreto[2]

[1] Federal Institute of Ceará, Maracanaú Campus, Teleinformatics Department,
Av. Contorno Norte, S/N, Maracanaú, Ceará, Brazil
ajalmar@ifce.edu.br

[2] Federal University of Ceará, Department of Teleinformatics Engineering
Center of Technology, Campus of Pici, Fortaleza, Ceará, Brazil
guilherme@deti.ufc.br

Abstract. This paper introduces a new approach to building hard margin classifiers based on Opposite Maps (OM). OM is a Self-Organizing Map-based method used for obtaining reduced-set classifiers in the sense of soft margin. As originally proposed, Opposite Maps was used for reducing the training data set and obtaining soft margin reduced-set SVM and LSSVM classifiers. In our new proposal we use Opposite Maps in order to obtain a set of patterns in the overlapping area between positive and negative classes and, a posteriori, to remove them from the default training data set. This process can transform a non-linear problem into a linear one in which a hard-margin classifier like Huller SVM can be applied. This approach assure to get resulting classifiers from a training process without needing to set up the cost parameter C that controls the trade off between allowing training errors and margin maximization. Besides that, but differently from soft-margin classifiers, these obtained classifiers leave the patterns at wrong side of the hyperplane out of the set of support vectors and, therefore, reduced-set hard-margin classifiers come out with few support vectors.

Keywords: Opposite Maps, Hard-Margin Support Vector Machines, Reduced-Set Classifiers, Huller SVM, Self-Organizing Map.

1 Introduction

Pattern classification algorithms aims at providing functions that defines the relation between input vectors and their class labels. Differently from Artificial Neural Networks and Decision Trees training process, the training process of Large Margin Classifiers like Support Vector Machines (SVM) is based on minimizing the empirical and structural risk [18]. Minimizing the empirical risk means to reducing the classification error on training data set and minimizing the structural risk is related to reducing the classification error on unseen patterns. This is an advantage of SVM over other classifiers, however, the SVM classifier

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 65–74.
DOI: 10.1007/978-3-642-35230-0_7

has an important drawback: the run time complexity can be considerably higher because of the large number of support vectors [3,12].

To handle this issue, several Reduced Set (RS) methods have been proposed to alleviate this problem, either by eliminating less important SVs or by constructing a new (smaller) set of training examples, often with minimal impact on performance [4,6,8,10,17]. An alternative to standard SVM formulation is the Least Squares Support Vector Machine (LS-SVM) [16], which leads to solving linear KKT systems[1] in a least square sense. The solution follows directly from solving a linear equation system, instead of a quadratic programming optimization problem. As we know, it is in general easier and less computationally intensive to solve a linear system than a QP problem. On the other hand, the introduced modifications also result in loss of sparseness of the induced SVM. It is common to have all examples of the training data set belonging to the set of the SVs. To mitigate this drawback, several pruning methods have been proposed in order to improve the sparseness of the LS-SVM solution [7,11,15].

The flexibility added to SVM classifiers with the aim of allowing some training errors by making the margin soft causes the increase in the quantity of support vectors due to Lagrange multipliers related to the patterns placed at wrong side of the hyperplane have non-zero value. As we know, this flexibility of soft margin SVM classifiers has been succeeding in non-linear problems. Nevertheless, a hard margin SVM classifier differently considers only the patterns on the margin as support vectors and, therefore, this kind of classifier has less support vectors than soft margin classifiers. In this point of view, an interesting strategy for reducing the number of support vectors is to apply some existing algorithm in order to transform a non linear problem into a linear one in which a hard margin classifier can be applied.

In this paper, we introduce a new proposal called Opposite Maps Hard Support Vector Machines (OM-HSVM) that comes from applying the Opposite Maps in order to detect the overlapping area between positive and negative classes and, a posteriori, to remove them from the default training data set and then from using a hard margin support vector machine as Huller SVM to classifying unseen patterns. This new approach assures to get resulting classifiers from a training process without needing to set up the cost parameter C that controls the trade off between allowing training errors and margin maximization. Besides that, but differently from soft-margin classifiers, these obtained classifiers leave the patterns at wrong side of the hyperplane out of the set of support vectors and, therefore, reduced-set hard-margin classifiers come out with few support vectors.

This paper is organized as follows. In Section 2 we review the fundamentals of the soft margin SVM classifiers and SMO algorithm. In Section 3 we describe the theory of hard margin and Huller SVM classifiers. In Section 4 we present the Opposite Maps algorithm and then in Section 5 we describe our proposal. Simulations and results are shown in Section 6. The paper is concluded in Section 7.

[1] Karush-Kuhn-Tucker systems.

2 Soft Margin Support Vector Machines

Consider a training data set $\{(\mathbf{x}_i, y_i)\}_{i=1}^{L}$, so that $\mathbf{x}_i \in \mathbb{R}^p$ is an input vector and $y_i \in \{-1, +1\}$ are the corresponding class labels. For soft margin classification, the SVM primal problem is defined as

$$\begin{aligned} &\min_{\mathbf{w},\boldsymbol{\xi}} \left\{\tfrac{1}{2}\mathbf{w}^T\mathbf{w} + C\textstyle\sum_{i=1}^{L}\xi_i\right\} \\ \text{subject to} \quad &y_i[(\mathbf{w}^T\mathbf{x}_i) + b] \geq 1 - \xi_i, \quad \xi_i \geq 0 \end{aligned} \tag{1}$$

where $\{\xi_i\}_{i=1}^{L}$ are slack variables and $C \in \mathbb{R}$ is a cost parameter that controls the trade-off between allowing training errors and forcing rigid margins.

The solution of the problem in Eq. (1) is the saddle point of the following Lagrangian function:

$$L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{L}\xi_i - \sum_{i=1}^{L}[\alpha_i(y_i(\mathbf{x}_i^T\mathbf{w} + b) - 1 + \xi_i) + \beta_i\xi_i], \tag{2}$$

where $\boldsymbol{\alpha} = \{\alpha_i\}_{i=1}^{L}$ and $\boldsymbol{\beta} = \{\beta_i\}_{i=1}^{L}$ are Lagrange multipliers. This Language must be minimized with respect to $\mathbf{w}$, b and ξ_i, as well as maximized with respect to α_i and β_i . For this purpose, we need to compute the following differentiations:

$$\frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta_i})}{\partial \mathbf{w}} = \mathbf{0}, \qquad \frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta_i})}{\partial b} = 0 \quad \text{and} \quad \frac{\partial L(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta_i})}{\partial \xi_i} = 0,$$

resulting on $\mathbf{w} = \sum_{i=1}^{L}\alpha_i y_i \mathbf{x}_i$, $\sum_{i=1}^{L}\alpha_i y_i = 0$ and $C = \alpha_i + \beta_i$, respectively. Introducing these expressions into Eq.(2), we present the SVM dual problem as

$$\begin{aligned} \max J(\boldsymbol{\alpha}) &= \sum_{i=1}^{L}\alpha_i - \frac{1}{2}\sum_{i=1}^{L}\sum_{j=1}^{L}\alpha_i\alpha_j y_i d_j \mathbf{x}_i^T\mathbf{x}_j, \\ \text{subject to} \quad &\sum_{i=1}^{N}\alpha_i y_i = 0 \text{ and } 0 \leq \alpha_i \leq C. \end{aligned} \tag{3}$$

Once we have the values of the Lagrange multipliers, the output can be calculated based on the classification function described as

$$f(\mathbf{x}) = \operatorname{sign}\left(\sum_{i=1}^{L}\alpha_i y_i \mathbf{x}^T\mathbf{x}_i + b\right). \tag{4}$$

It is straightforward to use the *kernel trick* to generate non-linear versions of the standard linear SVM classifier. This procedure works by replacing the dot product $\mathbf{x}^T\mathbf{x}_i$ with the kernel function $k(\mathbf{x}, \mathbf{x}_i)$. A symmetric function $k(\mathbf{x}, \mathbf{x}_i)$ is a kernel if it fulfills Mercer's condition, i.e. the function $K(\cdot, \cdot)$ is (semi) positive definite. In this case, there is a mapping $\phi(\cdot)$ such that it is possible to write $K(\mathbf{x}, \mathbf{x}_i) = \phi(\mathbf{x}) \cdot \phi(\mathbf{x}_i)$. The kernel represents a dot product on feature space into which the original vectors are mapped.

2.1 Sequential Minimal Optimization

Sequential Minimal Optimization (SMO) is an iterative algorithm for solving the dual Soft Margin SVM optimization problem presented in Eq. (3) . The algorithm selects two parameters, α_p and α_q, from the set of the Lagrange multipliers, $\{\alpha_i\}_{i=1}^{l}$, and optimizes the objective value jointly for both these α values. At the end of the algorithm it adjusts the bias (b parameter) based on the new parameter set. This process as described bellow is repeated until the set of the Lagrange multipliers convergence.

```
1. Initialize $\alpha_i \leftarrow 0$ and $b \leftarrow 0$;
2. Let $f^{'}(\mathbf{x}) = \sum_{i=1}^{l} y_i \alpha_i k(\mathbf{x}, \mathbf{x}_i) + b$;
3. Let $E_i = f^{'}(\mathbf{x}_i) - y_i$;
4. Let $\lambda$ be the tolerance;
5. Principal loop
   (a) Use heuristics to choose two Lagrange multipliers, $\alpha_p$ and $\alpha_q$, from $\{\alpha_i\}_{i=1}^{l}$ to jointly optimize;
   (b) if can not be found such examples then exit principal loop;
   (c) Compute $\mu$, such that $\mu \leftarrow \frac{E_q - E_p}{k(\mathbf{x}_p,\mathbf{x}_p) - 2k(\mathbf{x}_p,\mathbf{x}_q) + k(\mathbf{x}_q,\mathbf{x}_q)}$;
   (d) Update $\alpha_q^{new} \leftarrow \alpha_q + y_q \mu$;
   (e) Verify the bounds applied to $\alpha_q$;
   (f) Update $\alpha_p^{new} \leftarrow \alpha_p - y_p \mu$;
6. Update $b$ such that
   (a) $b_p \leftarrow E_p + y_p(\alpha_p^{new} - \alpha_p)k(\mathbf{x}_p, \mathbf{x}_p) + y_q(\alpha_q^{new} - \alpha_q)k(\mathbf{x}_q, \mathbf{x}_q) + b$;
   (b) $b_q \leftarrow E_q + y_p(\alpha_p^{new} - \alpha_p)k(\mathbf{x}_p, \mathbf{x}_p) + y_q(\alpha_q^{new} - \alpha_q)k(\mathbf{x}_q, \mathbf{x}_q) + b$;
   (c) $b = (b_p + b_q)/2$;
```

The algorithm described above owns some simplifications. In this work, we have implemented the algorithm as presented in Platt's work [13].

3 Hard Margin Support Vector Machines

The geometrical formulation of Hard Margin Support Vector Machines is presented in [1,5]. For a separable training data, the convex hulls formed by the positive and negative examples are disjoint. In this context, consider two points $\boldsymbol{X}_p$ and $\boldsymbol{X}_n$ belonging to each convex hull and then make them as close as possible without allowing them to leave their respective convex hulls. The median hyperplane of these two points is the maximum margin separating hyperplane and it can be obtained by solving

$$\min_{\boldsymbol{\alpha}} ||\boldsymbol{X}_p - \boldsymbol{X}_n||^2 \tag{5}$$

subject to

$$\boldsymbol{X}_p = \textstyle\sum_{i \in P} \alpha_i \boldsymbol{x}_i \qquad \sum_{i \in P} \alpha_i = 1 \quad \alpha_i \geq 0$$

and

$$\boldsymbol{X}_n = \textstyle\sum_{j \in N} \alpha_j \boldsymbol{x}_j \qquad \sum_{j \in N} \alpha_j = 1 \quad \alpha_j \geq 0 \ ,$$

where the sets $P = \{i|(\mathbf{x}_i, y_i) \wedge y_i = +1\}$ and $N = \{j|(\mathbf{x}_j, y_j) \wedge y_j = -1\}$ respectively contain the indexes of the positive and negative patterns. The optimal hyperplane is then represented by the following linear discriminant function

$$f(\boldsymbol{x}) = \mathrm{sign}\left((\boldsymbol{X}_p - \boldsymbol{X}_n)\boldsymbol{x} + (\boldsymbol{X}_n\boldsymbol{X}_n - \boldsymbol{X}_p\boldsymbol{X}_p)/2\right). \tag{6}$$

Note that $\boldsymbol{X}_p$ and $\boldsymbol{X}_n$ are described as linear combination of the training patterns, so both the discriminant function and the optimization criterion can be expressed using dot products between patterns in order to use kernel functions and obtaining non-linear classifiers.

3.1 Huller SVM

We now describe the algorithm Huller that can be viewed as a simplification of the nearest point algorithms discussed in [1,5] and improved by the additions presented in [2]. The algorithm uses the parametrization $\boldsymbol{X}_p = \sum_{i \in P} \alpha_i \boldsymbol{x}_i$ and $\boldsymbol{X}_n = \sum_{j \in N} \alpha_j \boldsymbol{x}_j$ to store these points.

The new position of the point $\boldsymbol{X}_p^{(t+1)}$ at iteration $t+1$ is obtained by the follow expression:

$$\boldsymbol{X}_p^{(t+1)} = (1-\lambda)\boldsymbol{X}_p^{(t)} + \lambda \boldsymbol{x}_k, \tag{7}$$

where the value λ is between $\alpha_k/(1-\alpha_k)$ and 1. As presented in [2], the optimal value λ is computed analytically by the following expression:

$$\lambda = \min\left(1, \max\left(\frac{-\alpha_k}{1-\alpha_k}, \lambda_u\right)\right) \tag{8}$$

where

$$\lambda_u = \begin{cases} \frac{(\boldsymbol{X}_p^{(t)} - \boldsymbol{X}_n^{(t)})(\boldsymbol{X}_p^{(t)} - \boldsymbol{x}_k)}{(\boldsymbol{X}_p^{(t)} - \boldsymbol{x}_k)^2} & \text{if } y_k = +1 \\ \frac{(\boldsymbol{X}_n^{(t)} - \boldsymbol{X}_p^{(t)})(\boldsymbol{X}_n^{(t)} - \boldsymbol{x}_k)}{(\boldsymbol{X}_n^{(t)} - \boldsymbol{x}_k)^2} & \text{if } y_k = -1 \end{cases}$$

The new position of the point $\boldsymbol{X}_n^{(t+1)}$ is computed similarly by $\boldsymbol{X}_p^{(t+1)}$. The Huller algorithm is resumed as follows.

```
STEP 1 - Initialize $\boldsymbol{X}_p^{(0)}$ and $\boldsymbol{X}_n^{(0)}$ by averaging a few points;
STEP 2 - Iterate $N$ times in this principal loop;
    Step 2.1 - Pick a random pattern with index $k$ such that $\alpha_k = 0$;
    Step 2.2 - If $\boldsymbol{x}_k$ has a positive label (i.e., $y_k = +1$) then compute $\boldsymbol{X}_p^{(t+1)}$;
    Step 2.3 - Otherwise; compute $\boldsymbol{X}_n^{(t+1)}$;
    Step 2.4 - Pick a random pattern with index $z$ such that $\alpha_z \neq 0$;
    Step 2.5 - If $\boldsymbol{x}_k$ has a positive label (i.e., $y_z = +1$) then compute $\boldsymbol{X}_p^{(t+1)}$;
    Step 2.6 - Otherwise; compute $\boldsymbol{X}_n^{(t+1)}$;
STEP 3 - Obtain the Lagrange multipliers from $\boldsymbol{X}_p^{(N)}$ and $\boldsymbol{X}_n^{(N)}$.
```

4 Opposite Maps SVM

The Opposite Maps (OM) method [14] was proposed in order to find a reduced set of training vectors to induce SVM and LS-SVM classifiers. For a classification problem with K classes, the original OM requires K self-organizing maps (SOM) [9] to be trained, one for each available class. It is worth pointing out, however, that any vector quantization algorithm other than the SOM can be used by the OM method.

STEP 1 - Split the available data set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ into two subsets:

$$\mathcal{D}^{(1)} = \{(\mathbf{x}_i, y_i) | y_i = +1\}, \quad i = 1, \ldots, l_1 \qquad \text{(for class 1)} \tag{9}$$
$$\mathcal{D}^{(2)} = \{(\mathbf{x}_i, y_i) | y_i = -1\}, \quad i = 1, \ldots, l_2 \qquad \text{(for class 2)} \tag{10}$$

where l_1 and l_2 are the cardinalities of the subsets $\mathcal{D}^{(1)}$ and $\mathcal{D}^{(2)}$, respectively.

STEP 2 - Train a SOM network using the subset $\mathcal{D}^{(1)}$ and another SOM using the subset $\mathcal{D}^{(2)}$. Refer to the trained networks as SOM-1 and SOM-2.

STEP 3 - For each vector $\mathbf{x}_i \in \mathcal{D}^{(1)}$ find its corresponding BMU in SOM-1. Then, prune all *dead neurons*[2] in SOM-1. Repeat the same procedure for each vector $\mathbf{x}_i \in \mathcal{D}^{(2)}$: find the corresponding BMUs in SOM-2 and prune all the dead neurons. Refer to the pruned networks as PSOM-1 and PSOM-2.

STEP 4 - At this step the BMUs for the data subsets are searched within the set of prototypes of the *opposite map*.

Step 4.1 - For each $\mathbf{x}_i \in \mathcal{D}^{(1)}$ find its corresponding BMU in PSOM-2:

$$c_i^{(2)} = \arg\min_{\forall j} \|\mathbf{x}_i - \mathbf{w}_j^{(2)}\|, \quad i = 1, \ldots, l_1, \tag{11}$$

where $\mathbf{w}_j^{(2)}$ is the j-th prototype vector in PSOM-2. Thus, $c_i^{(2)}$ denotes the index of the BMU in PSOM-2 for the i-th example in $\mathcal{D}^{(1)}$.

Step 4.2 - For each $\mathbf{x}_i \in \mathcal{D}^{(2)}$ find its corresponding BMU in PSOM-1:

$$c_i^{(1)} = \arg\min_{\forall j} \|\mathbf{x}_i - \mathbf{w}_j^{(1)}\|,, \quad i = 1, \ldots, l_2, \tag{12}$$

where $\mathbf{w}_j^{(1)}$ is the j-th prototype vector in PSOM-1. Thus, $c_i^{(1)}$ denotes the index of the BMU in PSOM-1 for the i-th example in $\mathcal{D}^{(2)}$.

STEP 5 - Let $\mathcal{C}^{(2)} = \{c_1^{(2)}, c_2^{(2)}, \ldots, c_{l_2}^{(2)}\}$ be the index set of all BMUs found in Step 4.1, and $\mathcal{C}^{(1)} = \{c_1^{(1)}, c_2^{(1)}, \ldots, c_{l_1}^{(1)}\}$ be the index set of all BMUs found in Step 4.2.

STEP 6 - At this step the reduced set of data vectors is formed.

Step 6.1 - For each PSOM-1 unit in $\mathcal{C}^{(1)}$ find its nearest neighbor among the data vectors $\mathbf{x}_i \in \mathcal{D}^{(1)}$. Let $\mathcal{X}^{(1)}$ be the subset of nearest neighbors for the PSOM-1 units in $\mathcal{C}^{(1)}$.

[2] Neurons which have never been selected as the BMU for any vector $\mathbf{x}_i \in \mathcal{D}^{(1)}$.

Step 6.2 - For each PSOM-2 unit in $\mathcal{C}^{(2)}$ find its nearest vector $\mathbf{x}_i \in \mathcal{D}^{(2)}$. Let $\mathcal{X}^{(2)}$ be the subset of nearest neighbors for the PSOM-2 units in $\mathcal{C}^{(2)}$. Then, the reduced set of data examples is given by $\mathcal{X}_{rs} = \mathcal{X}^{(1)} \cup \mathcal{X}^{(2)}$.

The main idea behind the Opposite Maps is to deliver to the SMO algorithm an "almost solved problem", since for all data examples out of the reduced set (i.e. $\mathbf{x}_i \notin \mathcal{X}_{rs}$) their Lagrange multipliers are set to zero, the SMO algorithm is run only over the data examples belonging to the reduced set. This approach is henceforth called OM-SVM.

5 Proposal: Opposite Maps Huller SVM

As stated before, Opposite Maps was proposed to find the overlapping area between the positive and the negative classes when the problem is non-linear. In our new proposal the OM method is used as a step to detect the intersection area. After that, we remove the patterns that belong to this area in order to transform the non-linear problem into a linear one. With respect to this approach, it is worth emphasizing that the training process is carried out without setting up the cost parameter C and the resulting hard margin SVM classifiers come out with few support vectors. It is also important to point out that our proposal avoids solving the problem twice (with two training processes) as required by other proposals [3,4,6,8]. In these approaches a first solution is achieved to construct the decision surface. A second solution is carried out aiming at removing the patterns located at the wrong side of the initial surface and thus re-training the classifier with the remaining patterns. Instead, an alternative is to construct a reduced-set classifier based on an approximation to the decision surface with fewer support vectors.

The training of this classifier, hereinafter called Opposite Maps Huller SVM (OM-HSVM), is resumed as follows.

STEP 1 - Apply Opposite Maps to a non-linear problem with a data set $\mathcal{D}$ and obtain the overlapping area $\mathcal{X}_{rs}$.

STEP 2 - Remove the patterns from $\mathcal{D}$ that are not in $\mathcal{X}_{rs}$, i.e., compute $\mathcal{D}^* = \mathcal{D} - \mathcal{X}_{rs}$;

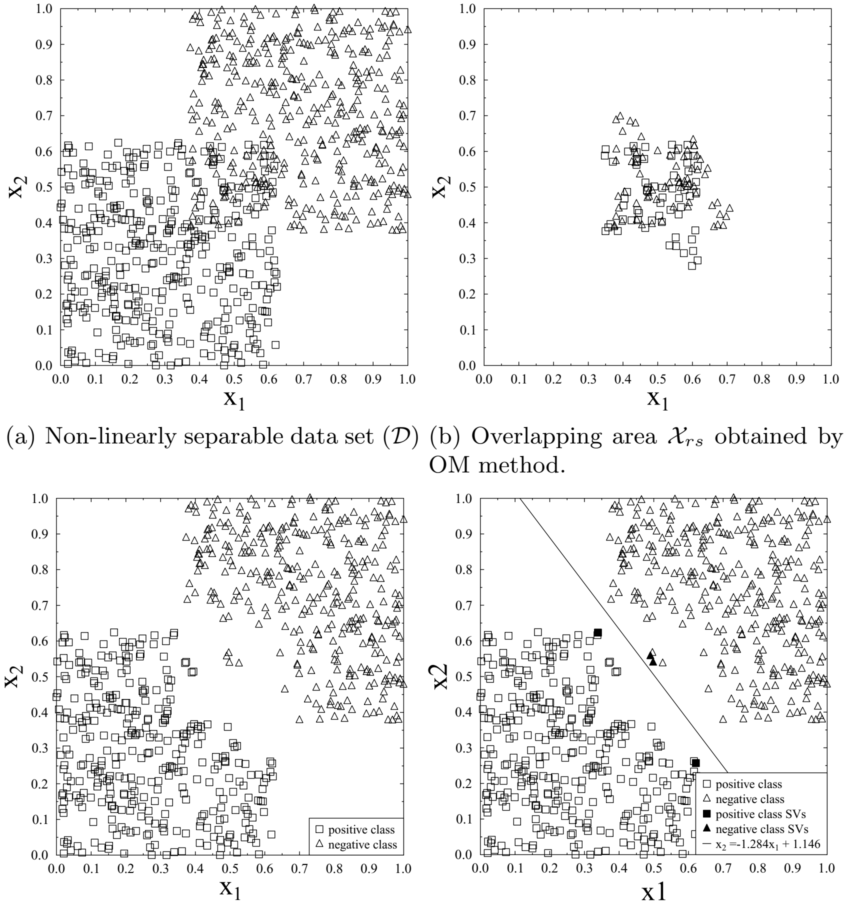
STEP 3 - Train a hard margin SVM classifier as Huller SVM over the resulting data set $\mathcal{D}^*$.

6 Simulations and Discussion

For all experiments to be described, 80% of the data examples were randomly selected for training purposes. The remaining 20% of the examples were used for testing the classifiers' generalization performances. All simulations were conducted using a standard 2-D SOM, hexagonal neighborhood, Gaussian neighborhood function, with random weight initialization.

For tests with the original OM method, we trained two SOMs with fixed $S \times S$ map grid, for 80 epochs with initial and final neighborhood radius (learning rate) of 5 (0.5) and 0.1 (0.01), respectively. For SVM-like classifiers we used linear kernels. The map grid size for each OM-SVM and OM-HSVM classifiers is presented in the result tables.

Initially, as a proof of concept, we have applied the OM-HSVM classifier to an artificial problem (called Two Squares), consisting of a non-linearly separable two-dimensional data set. Data instances within each class are independent and uniformly distributed with the same within- and between-class variances. The OM-HSVM classifier working is presented in Figure 1 and the results below indicate that the OM-HSVM produced a hard margin reduced-set SVM classifier using fewer SVs.



(a) Non-linearly separable data set ($\mathcal{D}$)

(b) Overlapping area $\mathcal{X}_{rs}$ obtained by OM method.

(c) Resulting linearly separable data set ($\mathcal{D}^*$).

(d) Decision surface of hard margin SVM classifier for data set ($\mathcal{D}^*$).

Fig. 1. OM-HSVM classifier applied to Two Squares problem

Table 1. Results for the SVM, OM-SVM and OM-HSVM classifiers

Data Set	Model	C	Tol.	Grid Size	Accuracy	Train. Size	SVs #	SV Red.
TWO SQUARES	SVM	1.0	0.01	–	93.9 ± 1.4	640	104.5	–
TWO SQUARES	OM-SVM	1.0	0.01	10 × 10	93.6 ± 2.0	640	96.6	7.6%
TWO SQUARES	OM-HSVM	–	–	10 × 10	93.7 ± 2.0	640	7.3	93.0%
BANANA	SVM	1.5	0.01	–	97.1 ± 1.0	1000	65.4	–
BANANA	OM-SVM	1.5	0.01	10 × 10	95.2 ± 1.6	1000	38.8	40.7%
BANANA	OM-HSVM	–	–	10 × 10	97.1 ± 0.8	1000	7.8	88.1%
RIPLEY	SVM	2.5	0.01	–	87.9 ± 1.6	1250	289.2	–
RIPLEY	OM-SVM	2.5	0.01	5 × 5	86.9 ± 2.0	1250	248.4	45.7%
RIPLEY	OM-HSVM	–	–	5 × 5	86.2 ± 2.1	1250	6.9	97.6%
BREST CANCER	SVM	0.04	0.001	–	97.2 ± 1.0	546	61.7	–
BREST CANCER	OM-SVM	0.04	0.001	10 × 10	95.9 ± 1.4	546	46.2	25.1%
BREST CANCER	OM-HSVM	–	–	10 × 10	97.0 ± 1.4	546	14.6	76.3%

In order to compare our proposal (OM-HSVM) to Opposite Maps SVM classifier (OM-SVM) and the default SVM classifier (SVM) tests with artificial (Two Squares, Banana and Ripley) and real-world benchmarking (Breast Cancer) data sets were also carried out and the obtained results are shown in Table 1. We report performance metrics (mean value and standard deviation of the recognition rate) on the testing set averaged over 20 independent runs. We also show the map grid size, the average number of SVs (SVs #), the reduction of the number of support vectors (SV Red.), as well as the values of the parameter C and the tolerance for SVM based on SMO algorithm.

By analyzing these tables, one can easily conclude that, as expected, the accuracies of all the reduced-set classifiers were equivalent to those achieved by the full-set classifiers. Moreover, one can also conclude that the accuracies of OM-HSVM classifier was similar to those achieved by the OM-SVM classifiers, with the advantage of significantly reducing the number of SVs at least 76.3% and up to 97.6%.

7 Conclusion

In this paper, we have proposed a novel approach to apply hard margin classifiers to non linear problems. The proposed approach, called Opposite Maps Huller SVM (OM-HSVM), consists in application of the OM algorithm to the original data set in order to detect the overlapping area between positive and negative classes and then remove it to transform the non linear problem into a linear one. The obtained results indicated that the OM-HSVM classifiers performs as well as the other SVM-like approaches providing a significant decrease in the number of SVs while maintaining equivalent accuracy. Currently, we are evaluation this proposal on multiclass problems.

References

1. Bennett, K.P., Bredensteiner, E.J.: Duality and geometry in svm classifiers. In: Proceedings of the Seventeenth International Conference on Machine Learning, ICML 2000, pp. 57–64 (2000)
2. Bordes, A., Bottou, L.: The Huller: A Simple and Efficient Online SVM. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.) ECML 2005. LNCS (LNAI), vol. 3720, pp. 505–512. Springer, Heidelberg (2005)
3. Burges, C.J.C.: Simplified support vector decision rules. In: Proceedings of the 13th International Conference on Machine Learning (ICML 1996), pp. 71–77. Morgan Kaufmann (1996)
4. Carvalho, B.P.R., Braga, A.P.: IP-LSSVM: A two-step sparse classifier. Pattern Recognition Letters 30, 1507–1515 (2009)
5. Crisp, D.J., Burges, C.J.C.: A geometric interpretation of v-svm classifiers. In: NIPS, pp. 244–250 (1999)
6. Geebelen, D., Suykens, J., Vandewalle, J.: Reducing the number of support vectors of svm classifiers using the smoothed separable case approximation. IEEE Transactions on Neural Networks and Learning Systems 23(4), 682–688 (2012)
7. Hoegaerts, L., Suykens, J.A.K., Vandewalle, J., De Moor, B.: A Comparison of Pruning Algorithms for Sparse Least Squares Support Vector Machines. In: Pal, N.R., Kasabov, N., Mudi, R.K., Pal, S., Parui, S.K. (eds.) ICONIP 2004. LNCS, vol. 3316, pp. 1247–1253. Springer, Heidelberg (2004)
8. Hussain, A., Shahbudin, S., Husain, H., Samad, S.A., Tahir, N.M.: Reduced set support vector machines: Application for 2-dimensional datasets. In: Proc. of the Second International Conf. on Signal Processing and Communication Systems (2008)
9. Kohonen, T.K.: Self-Organizing Maps. Springer (1997)
10. Lee, Y.J., Mangasarian, O.L.: SSVM: A smooth support vector machine for classification. Computational Optimization and Applications 20(1), 5–22 (2001)
11. Li, Y., Lin, C., Zhang, W.: Letters: Improved sparse least-squares support vector machine classifiers. Neurocomputing 69, 1655–1658 (2006)
12. Osuna, E., Girosi, F.: Reducing the run-time complexity of support vector machines. In: Proceedings of the Fourteenth International Conference on Pattern Recognition, Brisbane (1995)
13. Platt, J.C.: Fast training of support vector machines using sequential minimal optimization. MIT Press, Cambridge (1999)
14. Rocha Neto, A.R., Barreto, G.A.: A Novel Heuristic for Building Reduced-Set SVMs Using the Self-Organizing Map. In: Cabestany, J., Rojas, I., Joya, G. (eds.) IWANN 2011, Part I. LNCS, vol. 6691, pp. 97–104. Springer, Heidelberg (2011)
15. Suykens, J.A.K., Lukas, L., Vandewalle, J.: Sparse approximation using least squares support vector machines. In: Proceedings of 2000 IEEE International Symposium on Circuits and Systems, Geneva, Switzerland, pp. 757–760 (2000)
16. Suykens, J.A.K., Vandewalle, J.: Least squares support vector machine classifiers. Neural Processing Letters 9(3), 293–300 (1999)
17. Tang, B., Mazzoni, D.: Multiclass reduced-set support vector machines. In: Proc. of the 23rd International Conf. on Machine Learning, pp. 921–928 (2006)
18. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer, New York (1995)

Intuitive Volume Exploration through Spherical Self-Organizing Map

Naimul Mefraz Khan, Matthew Kyan, and Ling Guan

Ryerson University, Toronto, ON
{n77khan,mkyan,lguan}@ee.ryerson.ca

Abstract. Direct Volume Rendering is one of the most popular volume exploration methods, where the data values are mapped to optical properties through a Transfer Function (TF). However, designing an appropriate TF is a complex task for the end user, who may not be an expert in visualization techniques. The Self-Organizing Map (SOM) is a perfect tool to hide irrelevant TF parameters and, through unsupervised clustering, present a visual form of the topological relations among the clusters. This paper introduces a novel volume exploration technique which utilizes the cluster visualization ability of SOM to present a simple intuitive interface to the user for generating suitable TFs. Rather than manipulating TF or cluster parameters, the user interacts with the spherical lattice of the SOM to discover interesting regions in the volume quickly and intuitively. The GPU implementation provides real-time volume rendering and fast interaction. Experimental results on several datasets show the effectiveness of our proposed method.

1 Introduction

Volume exploration is an important technique to reveal inner structures and interesting regions in a volumetric dataset. However, exploring the volume is a difficult and non-intuitive task since there is no prior information available regarding the data distribution. 3D representation of a volume adds complexity to the whole process. To ease this process, Direct Volume Rendering (DVR) makes use of a Transfer Function (TF), which maps one or more features extracted from the data (the feature space) to different optical properties such as color and opacity. The TF design is typically a user-controlled process, where the user interacts with different widgets (usually representing feature clusters or 1D/2D histograms) to set color and opacity properties to the feature space. The user can also control some low-level properties like number of clusters, cluster variance etc. Most of the recently proposed DVR methodologies [1,6,16,13] are based on this philosophy.

However, interacting with the feature space is difficult for the end-user, who may not have any knowledge about feature extraction and clustering. Also, these kind of widgets try to represent the feature space directly, putting a restriction on the dimensionality of the feature space. Some methods use alternative ways to represent the higher-dimensional feature space through manageable widgets. For instance, in [10], spatial information is encoded in the color values while opacity is derived through intensity

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 75–84.
DOI: 10.1007/978-3-642-35230-0_8

and gradient. But these kind of alternatives are restrictive in the sense that the clustering or histogram generation is not directly derived from the full feature set. Also, only the specific features used in the proposed methods can be used for all datasets. Volume rendering has wide range of applications in different fields, and one set of features useful for a specific application might be completely irrelevant in another. Hence, there is a need to make the method independent so that any feature irrespective of its dimensionality can be represented to the user in a visual form while maintaining the topological relationship between various data distributions. This is exactly what a Self-Organizing Map (SOM) [8] can do. SOM preserves the input data topology and helps to generate a lower dimensional visualization of the clusters. The SOM structure is particularly of interest for DVR because of it's visualization capability.

This paper proposes such a DVR system where the feature space is represented to the user with the help of SOM. Our proposed system has the following advantages over existing DVR techniques:

- Rather than manipulating cluster parameters or optical properties, the user simply interacts with a color-coded SOM lattice representing cluster densities. Due to this visual nature of SOM, there is no need to tweak the cluster parameters and perform operations like split and merge to precisely determine the number of clusters or cluster spread. The user only has to intuitively select or de-select the SOM regions to reveal corresponding structures in a volume.
- The proposed model is independent of the dimensionality of the feature space. Any feature irrespective of its dimension or complexity can be used with the model, which makes it very robust.

We use the Spherical SOM structure (SSOM) [11,14] because of it's relatively less restrictive structure (explained later) . The GPU implementation of our method provides fast interaction with the SOM and real-time volume rendering.

The rest of the paper is organized as follows: Section 2 discusses the related works in TF and SOM. Section 3 details our proposed method. Section 4 provides results on some well-known datasets. Finally, Section 5 provides the conclusion.

2 Related Work

The relevant works fall into two categories: 1) Transfer Function Specification and 2) Spherical Self-Organizing Maps.

2.1 Transfer Function Specification

As TF specification is a huge research area itself, only recent and relevant works will be briefly discussed here. Traditionally, TF were only one-dimensional, where the color and opacity is derived from intensity value only. The work of Kindlmann and Durkin first popularized multi-dimensional TF, where intensity and gradient magnitude value were used to generate a 2D histogram to emphasize boundaries between different materials in a volume. Since then, many methods have been proposed to simplify the interaction with a multi-dimensional TF. A semi-automatic TF generation based on Radial

Basis Function (RBF) networks is presented in [13]. A nonparametric density estimation technique is introduced in [5]. A Gaussian Mixture Model-based clustering technique is presented in [16], where the Gaussians can be mapped to a set of elliptical transfer functions. A mixed model based on mean-shift and hierarchical clustering on the Low-High (LH) values of a volume is described in [1]. Since including only the intensity and gradient value results in local features, Roettger et al. [10] propose transfer functions that consider spatial information for clustering on 2D histograms. An intelligent user interface has been introduced in [15], where the user can paint on different slices to mark areas of interests. Neural network based techniques are then used to generate the TF. A spreadsheet-like interface based on Douglas-Peucker algorithm is presented in [4], where the user can combine simpler pre-defined TFs to generate complex ones.

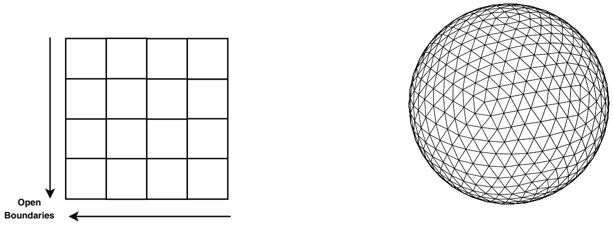
Despite all these efforts to make TF specification a simple and intuitive task, most of these methods still rely on some form of user control (e.g. number of clusters, variance of clusters, merging and splitting of clusters) in the feature space. An expert from medical or architectural background might not be familiar with these specifications. Moreover, most of these methods present visualization of the feature space itself. Hence, it is not possible to incorporate new features. As volume data can be complex, noisy and highly domain dependent, a straight-forward way to incorporate new features into the system is necessary. Our proposed model eliminates these limitations by using Self-Organizing Maps (SOM).

Our method draws some inspiration from the method proposed in [7]. However, the SOM visualization and volume exploration in our proposed method is completely different. Our method presents the visual representation of the cluster densities and allows the user to find the correspondence between SOM nodes and voxels interactively. On the other hand, the SOM is used in [7] mainly for dimensionality reduction. The Gaussian TF generation in [7] can also be difficult to control, as the direct correspondence between voxels and SOM nodes are not fully exploited. We also retain spatial information in our feature set (Section 2.1), which can produce better separation of voxel clusters. Lastly, in [7], a two-pass SOM training is used to better represent the boundary voxels, which can slow down the training process. Instead, we use the count-dependent parameter from [11] (details in Section 3), which can result in faster training times. We also use the Spherical Self-Organizing Map for it's advantages as described in the next section.

2.2 Spherical Self-Organizing Maps

The Self-Organizing Map (SOM) is an unsupervised clustering method proposed by Kohonen [8] that clusters the data and projects data points onto a lower-dimensional space while preserving the input topology. The data points are projected onto a regular lattice during training, when the weights of the nodes in the lattice are updated. In this way, after training the initial lattice "self-organizes" itself to represent the data distribution. Different coloring methods are then applied on the lattice to color code it so that the cluster regions can be visualized. As stated before, due to the easy visualization property of SOM, it is suitable for presenting volumetric data to the user for TF design.

The traditional SOM however has a rectangular 2D grid structure, which has a restricted boundary. This is because the boundaries are open (Figure 1), and the nodes on the boundary do not have the same number of neighbors as the inner nodes. Nodes on the opposite sides of the boundary are not topologically close in the SOM space. The same is true for even a 3D cubic structure [11]. In some cases, a wrap-around is introduced on the 2D structure to eliminate this boundary condition. However, this introduces other problems such as folding and twisting [11]. The ideal structure for SOM will be a lattice that minimizes topological discontinuity. Hence, we use the Spherical SOM (SSOM) [11] for our method. The SSOM structure is created by repeatedly subdividing an Icosahedron. This provides a spherical structure with symmetric node distribution.



(a) Open boundaries of 2D SOM (b) Closed structure of Spherical SOM

Fig. 1. Depiction of closed structure of spherical SOM compared to the traditional 2D SOM

As seen in Figure 1, the SSOM does not have the restricted boundary problem. The SSOM is also of particular interest because of it's 3D structure, which can be easier to navigate.

3 Proposed Method

In this section, we describe the details of our proposed system based on the SSOM structure. This section is divided into three subsections where we discuss different aspects of the proposed system: 1) Feature extraction, where we discuss about the features used in this paper to model a volume for TF generation; 2) SSOM training, where the detailed steps of the Spherical SOM training is described and 3) TF representation and exploration, where the simple and efficient manner in which a user can explore different Transfer Functions through the visual representation of an SSOM is explained.

3.1 Feature Extraction

As described in Section 2.1, multi-dimensional TFs mostly use some form of histogram based on intensity and gradient magnitude. However, one problem with histogram is that it cannot retain the spatial information present in a volume [10]. As present day scanning systems for volume data (CT, MRI etc.) have some inherent noise associated with them, losing spatial information might prove to be costly and may not cluster the volume data in a suitable way for volume rendering. On the other hand, incorporating the spatial information directly is difficult for any histogram-based approach, as the histogram will be even higher-dimensional and there is no easy way to represent it visually. As a result, in histogram-based approaches, the spatial information is used for color generation only [10]. However, as described before, our proposed model is independent of feature dimensionality due to the use of SSOM lattice. No matter how high the dimension of our feature set is, we can map it to a Spherical SOM and represent the clustering visually. As a result, in our proposed model, the spatial information is directly embedded into the feature definition. To emphasize the boundaries between materials [1], we also use the intensity value of each voxel and it's 3D gradient magnitude. Our 5D feature set consist of the following features:

- The X,Y and Z coordinates,
- the intensity value and
- the 3D gradient magnitude. The 3D gradient magnitude is defined by $G = \sqrt{G_x^2 + G_y^2 + G_z^2}$, where G_x,G_y and G_z are the gradient values along X, Y and Z direction, respectively.

All the features are scaled to fall between the value of $\{0,1\}$. Ideally, we would still like to emphasize the boundaries of materials over spatial similarity. Hence, the used features are weighted. For our experiments, we use a weight set of $\{0.5, 0.5, 0.5, 1, 2\}$ for the aforementioned features.

3.2 SSOM Training

The training phase of the SSOM is similar to the classical SOM [8]. Let, the input space of N voxels be represented by $\mathscr{X} = \{x_i\}_{i=1}^{N}$. Let, the SSOM be represented by M nodes ($M << N$). Each node in the SSOM lattice has a corresponding weight vector w. All these weight vectors together represent our SSOM space $\mathscr{W} = \{w_i\}_{i=1}^{M}$. Each node also has a *neighborhood* associated with it. A neighborhood is a set of nodes consisted of the node itself and its neighbors. Let, the neighborhood set for node i be represented by Θ_i^r. Here, r represents the neighborhood spread, $r = 1, \ldots, R$. R is the maximum neighborhood radius, which is set to a value such that it covers half of the spherical space [11].

The input voxels are randomly introduced to the SSOM during training. For each voxel, a *Best Matching Unit (BMU)* among all the nodes is selected. BMU is the node which is closest to the input voxel according to some similarity measure. Euclidean distance is usually used as distance measure. The update step then takes place, where the weight vector of the BMU and it's neighboring nodes (Θ_{BMU}^r) are updated in a way so that they are pulled closer to the weight of the input voxel. After training, the SSOM weight vectors are arranged in such a way that represents the underlying distribution of the input data (the voxel features in this case). The training algorithm is described below:

- **Initialization:** The weight vectors of the SSOM nodes are initialized first. Random values can be used for initialization, but as pointed out by Kohonen et al. in [8], random initialization will take more time to converge. We have followed the initialization method stated in [8] i.e. initialize the weight vectors with values that lie on the subspace spanned by the eigenvectors corresponding to the two largest principal components of the input data distribution. A count vector $\mathscr{C} = \{c_i\}_{i=1}^{M}$ [11] is used to keep track of the hits to each node. This vector is initialized to zero. This is used in the BMU selection step (explained below) to prevent cluster under-utilization. This is especially necessary in our case because in a volume, typically there is almost $70\% - 80\%$ homogeneous regions. Without a count-dependent control, homogenous regions will take over the whole map and important regions (boundaries) will not get enough map space to be noticed.

- **Training:** For each input voxel x, do the following:
  - **BMU Selection:** Calculate the Euclidean distance of the voxel feature vector x with all the nodes as follows:

$$e_i = (c_i + 1)\|x - w_i\|, \quad i = 1, \ldots, M. \tag{1}$$

    BMU is the node for which this distance is the smallest.
  - **Weight Update:** Update the weights for the BMU and it's neighboring nodes (defined by Θ^r_{BMU}) as follows:

$$w = w + b(t) * h(s,r) * \|x - w\|, \tag{2}$$

$$c_w = c_w + h(s,r), \tag{3}$$

    where $w \in \Theta^r_{BMU}$, $b(t) = \alpha e^{-\frac{t}{T}}$ and $h(s,r) = e^{-\frac{r^2}{s*R}}$.

    The functions $b(t)$ and $h(s,r)$ control the rate of learning and the neighborhood effect, respectively. $b(t)$ decreases in value as the epoch number $t = 1, 2, \ldots, T$ increases. It also depends on the learning rate α. $h(s,r)$ depends on the neighborhood size parameter s, which is user controlled. $h(s,r)$ is a Gaussian function. The further a neighboring node is from a BMU, the less it's weight will be affected.

    As discussed before, the count-dependent parameter c_w is increased here to prevent cluster under-utilization. Observing Equation (3) and Equation (1), we see that the BMU and its neighbors are "penalized" for winning by increasing the count. In the next update step, the distance measure with these nodes will be higher due to the increase of count. This ensures that one node (or its neighbors) does not win too many times, and the entire map is utilized [3].
- Repeat the training steps for a pre-defined number of epochs (T).

The main control parameters in SSOM training are the learning rate α, the number of epochs T and the neighborhood size parameter s. In case of volume rendering, we are dealing with a huge number of voxels (e.g. for a volume of dimension $256X256X256$, there are over 16 million voxels). As we have found experimentally, for such high number of voxels, the SOM typically converges within $2 - 3$ epochs. The α is set to 0.1 in our experiments. The neighborhood size parameter is set to $s = 2$, which is determined through trial and error.

3.3 TF Representation and Exploration

After training of the SSOM is completed, the weight vectors associated with the nodes of the SSOM represent the underlying clustering of the voxels. We map this SSOM to a suitable TF in three steps: 1) present a color coded graphical representation of the SSOM, 2) provide the user interaction options to select interesting regions in the SSOM, and 3) map selected regions of the SSOM to a suitable TF and show the rendering of the volume with the generated TF. Due to our GPU implementation, the user can see the rendering of the volume in real-time while interacting with the SSOM.

To color code the spherical lattice, the U-Matrix approach is used [11], which visualizes the distance between the weight vectors of the nodes. For each node, the average Euclidean distance of its weight vector with the weight vectors of all the immediate neighboring nodes is calculated. These distance measures are then mapped to a color-map for visualization purposes. In this way, a homogeneously colored region will represent a cluster, while the cluster boundaries will incur a change in coloring. An important point to note here is that since volume rendering is an entirely perceptual process, it is not important to strictly define how many clusters we have or whether the cluster boundaries are very well defined or not. The important point is to color the SSOM lattice in such a way that intuitively interacting (explained below) with it will directly result in meaningful rendering. Representation of cluster densities in the form of color-map through U-matrix serves this purpose.

Our target is to keep the user interaction as minimum and efficient as possible. In the proposed system, the user can select or de-select any region of the spherical lattice. The selection is provided in the form of a rubber-band tool, where the user can drag across the surface of the sphere to select one or multiple nodes. The de-selection is performed in a similar way.

The last step of our system is to map the SSOM to a TF. The TF is essentially an RGBA texture with corresponding entry for each voxel. The RGB corresponds to the voxel color, while the alpha component defines the voxel opacity. To speed up the rendering process, we keep the RGB channels fixed throughout a session. The alpha value is the most crucial component, since the opacity determines the visibility of a voxel. In our proposed model, we map the color channels to different features. One channel is mapped to intensity, while the other two channels are mapped to the 3D gradient magnitude. In this way, the boundaries between the materials will be colored differently.

The opacity of a voxel depends on the user selection. For voxels corresponding to the selected region $\mathscr{S}$ of the map, the opacity is calculated through following equation:

$$I(x) = 1 - \frac{||w_{BMU_x} - x||}{max_{y\in\mathscr{S}}(||w_{BMU_y} - y||)}, \tag{4}$$

where x represents the voxel for which the opacity is being calculated, and y represents all other voxels under the selected region $\mathscr{S}$. This essentially means that the closer the voxel will be to the weight vector of its winning node, the more opaque (less transparent) it will be. This assigns intuitive opacity values to the voxels corresponding to the selected SSOM nodes. This value is then scaled to be in a higher opaque region (0.6 – 1.0 in our experiments). The voxels corresponding to the unselected regions are assigned a very low but nonzero opacity value (0.01) so that the user still has some context while viewing the rendering of the selected region.

4 Experimental Results

To show the effectiveness of our system, we present some volume rendering results on popular datasets. We used three volume datasets [9], CT scans of a Foot, an Engine and a Piggy Bank. The size of the datasets are listed in Table 1. Generally, CT scan datasets have a lot of vacant regions (air around the object). If we build the SSOM directly on the dataset, these regions will generate a misleading clustering. A simple region growing algorithm [2] was used to separate the object from these regions first. These separated voxels were then fed to the SSOM training algorithm.

As stated before, due to the high number of voxels (in the range of millions), $2-3$ epochs of training is good enough for a visualization of the cluster. Such low number of epochs is reasonable here, since the convergence of map is not very critical, as long as we can have a color-coded SSOM where different cluster regions and the borders between them are visually distinguishable. The training times required for the datasets are listed in Table 1. Please note that the training of a SSOM has to be done only once for each dataset and does not effect the rendering process of our proposed method, which is required to be real-time.

Table 1. The size of the volume datasets and the required SSOM training times (in *seconds*)

Dataset Name	Size	Training Time
Foot	256X256X256	342.3
Engine	256X256X256	308.3
Piggy Bank	512X512X134	1341.4

After the training, we present a visual form of the spherical lattice to the user (details in Section 3.3). The user can then select or de-select regions of interest. The voxels of the selected regions were assigned opacity values using Equation (4). The RGBA texture according to the generated TF is then rendered on the GPU in real time. We have used the Visualization Toolkit [12] for our GPU rendering purposes.

Figure 2 shows some results obtained from our experiments. The top row (Figure 2.(a), (d), (g)) shows the rendering results when the full SSOM is selected. As we can see, although the surfaces of the volumes are visualized clearly, not much useful structural information can be gathered from these renderings. Figure 2.(b), (e), (h) shows the SSOMs corresponding to the three datasets with some parts of the maps selected by the user (the selected nodes are highlighted by white spots). Figure 2.(c), (f), (i) shows the corresponding renderings from the selected nodes. As we can see, for all three volumes, the important structures can be highlighted easily. The inner bones are visible in the Foot dataset (Figure 2.(c)). Similarly, the tubes of the Engine are visible and the coins inside the Piggy Bank are visible (Figure 2.(f), (i)). This clearly shows the effectiveness and efficiency of our method. Another interesting observation here is the nature of the selected regions on the map. As we can see, the selected regions contain

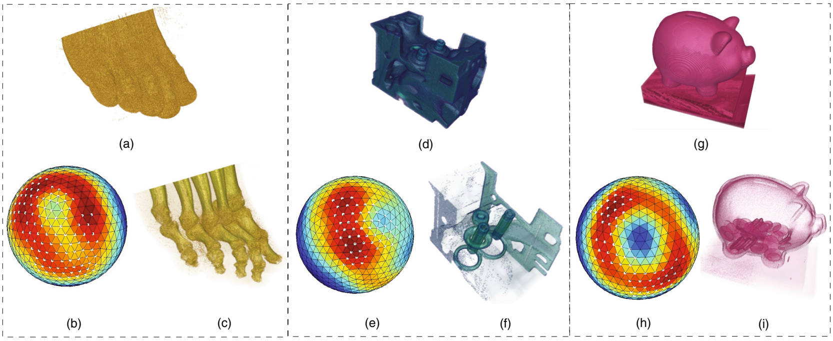


Fig. 2. Rendering results and corresponding SSOMs for the three datasets, (a)-(c): Foot; (d)-(f): Engine and (g)-(i): Piggy Bank

overlapping cluster regions in some cases. This is where the power of SSOM clustering is apparent. Since the clustering is not strict and the user has the freedom to select whatever region he or she wants, the whole process is very flexible. Also, due to the spherical structure of the map, it is easy to generate customized rendering depending on the users' need very easily. All the user has to do is select the appropriate regions on the map.

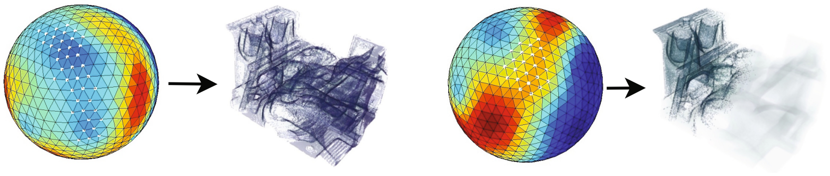


Fig. 3. Additional renderings corresponding to selected regions on the SSOM for the Engine

Figure 3 shows additional renderings corresponding to different selected regions on the SSOM. As we can see, the first region corresponds to a silhoutte-like rendering, while the second rendering focuses on the rear part of the engine. Since the selected region can be as small as only a single node, the system is very robust and adaptable.

5 Conclusion

In this paper, we have proposed a new intuitive way of direct volume rendering with the help of Spherical Self-Organizing Maps. The user interacts with the SSOM lattice to find interesting regions and generate suitable TF. Real-time rendering of the generated TF on the volume dataset provides instant feedback to the user. The proposed system is intuitive to interact with and robust in nature, since any feature can be used with the SSOM lattice irrespective of its complexity. Experimental results on some popular volume datasets verify the feasibility of our proposed approach. In future, we plan to extend this system to learn from user interaction and generate knowledge-assisted volume rendering accordingly.

References

1. Kindlmann, G., Durkin, J.W.: Semi-automatic generation of transfer functions for direct volume rendering. In: Proceedings of the 1998 IEEE Symposium on Volume Visualization, pp. 79–86. ACM, New York (1998)
2. Kohonen, T.: Self-organizing maps. Springer-Verlag New York, Inc., Secaucus (1997)
3. Krishnamurthy, A., Ahalt, S., Melton, D., Chen, P.: Neural networks for vector quantization of speech and images. IEEE Journal on Selected Areas in Communications 8, 1449–1457 (1990)
4. Liu, B., Wünsche, B., Ropinski, T.: Visualization by example - a constructive visual component-based interface for direct volume rendering. In: Proceedings of the International Conference on Computer Graphics Theory and Applications, pp. 254–259 (2010)
5. Maciejewski, R., Wu, I., Chen, W., Ebert, D.: Structuring feature space: A non-parametric method for volumetric transfer function generation. IEEE Transactions on Visualization and Computer Graphics 15, 1473–1480 (2009)
6. Nguyen, B., Tay, W., Chui, C., Ong, S.: A clustering-based system to automate transfer function design for medical image visualization. The Visual Computer 28, 181–191 (2012)
7. Pinto, F., Freitas, C.M.D.S.: Design of multi-dimensional transfer functions using dimensional reduction. In: Proceedings of the Eurographics Symposium on Visualization, pp. 131–138 (2007)
8. Pratt, W.K. (ed.): Digital Image Processing. John Wiley and Sons, New York (2007)
9. Roettger, S.: The volume library. Ohm Hochschule Nurnberg (2012), http://www9.informatik.uni-erlangen.de/External/vollib/
10. Roettger, S., Bauer, M., Stamminger, M.: Spatialized transfer functions. In: Proceedings of the IEEE/Eurographics Symposium on Visualization, pp. 271–278 (2005)
11. Sangole, A., Leontitsis, A.: Spherical self-organizing feature map: An introductory review, pp. 3195–3206 (2006)
12. Schroeder, W., Martin, K., Lorensen, B. (eds.): The Visualization Toolkit. Kitware, New York (2006)
13. Selver, M., Alper, M., Guzeli, C.: Semiautomatic transfer function initialization for abdominal visualization using self-generating hierarchical radial basis function networks. IEEE Transactions on Visualization and Computer Graphics 15, 395–409 (2009)
14. Tokutaka, H., Ohkita, M., Hai, Y., Fujimura, K., Oyabu, M.: Classification Using Topologically Preserving Spherical Self-Organizing Maps. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 308–317. Springer, Heidelberg (2011)
15. Tzeng, F.Y.: Intelligent system-assisted user interfaces for volume visualization. Ph.D. thesis, University of California, Davis, CA, USA (2006)
16. Wang, Y., Chen, W., Zhang, J., Dong, T., Shan, G., Chi, X.: Efficient volume exploration using the gaussian mixture model. IEEE Transactions on Visualization and Computer Graphics 17, 1560–1573 (2011)

Using SOM as a Tool for Automated Design of Clustering Systems Based on Fuzzy Predicates

Gustavo J. Meschino[1], Diego S. Comas[1,2], Virginia L. Ballarin[1],
Adriana G. Scandurra[1], and Lucía I. Passoni[1]

[1] Facultad de Ingeniería, Universidad Nacional de Mar del Plata
[2] Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)
{gmeschin,diegoscomas,vballari,scandu,lpassoni}@fi.mdp.edu.ar

Abstract. Clustering task is a never-ending research topic. New methods are permanently proposed. In particular, Fuzzy Logic and Self-organizing Maps and their mutual cooperation have demonstrated to be interesting paradigms. We propose a general approach to obtain membership functions for a ranked clustering system based on fuzzy predicates logical operations, considering Gaussian-shaped curves. We find membership functions parameters from trained Self-organizing Maps, which generalize the statistical characteristics of data. The system is self-configured and it has the advantages of other fuzzy approaches. Clustering quality is assessed by labeled data, which allow computing accuracy. The proposal must be tested with more real datasets, though the preliminary results obtained in well-known datasets suggest that it is a promising clustering scheme.

Keywords: fuzzy predicates, degree of truth, clustering, Self-organizing Maps.

1 Introduction

Clustering task is a never-ending research topic. New methods are permanently proposed [1]. In particular, Fuzzy Logic and Self-organizing Maps (SOM) and their mutual cooperation are demonstrated to be an interesting approach.

The theory of Fuzzy Logic based on fuzzy sets was proposed by Zadeh [2], who stated that a complex system will be better represented by descriptive variable of linguistic types [3]. The fuzzy rule-based approaches have been used in a wide range of classification and clustering problems [4, 5].

SOMs have been widely used to extract knowledge from data and to design clustering and classifying systems [6-8], many of them based on fuzzy rules as cited before [9-11], to define fuzzy inference systems (FIS).

Using simple rules is one of the main advantages of FIS. However, aggregation and defuzzification operations must be defined, which makes these models to be a little far from Boolean logic generalization. Defuzzification operation acts as a degree of freedom in a model based in the pragmatic combination of operators, but without an axiomatic link that justifies "logic" denomination [12].

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 85–94.
DOI: 10.1007/978-3-642-35230-0_9

Alternatively, in this work we propose the use of the Predicate Fuzzy Logic, which is a natural extension of Predicate Boolean Logic [13], to design a clustering system. We obtained a ranked clustering criteria represented as Fuzzy Predicates who consider the behaviour of the variables into the different clusters. These predicates can be evaluated using membership functions defined over input features. Hereby, we obtain a ranking for each datum, showing the degree of truth of "Datum *d* belongs to cluster *k*", being *k=1,2... K*, and *K* the amount of clusters. This ranking can be used for determining the group that new input data belongs to, adopting some criteria, but it also allows comparing the degree of membership to the clusters and assessing the contribution of individual features.

This approach adds value to previous knowledge-based guidance fuzzy clustering methods [14], but in this case the knowledge is data-driven obtained using the SOM generalization aptitude and taking advantage of the well-known SOM abilities to discover natural data grouping when compared with direct clustering [15, 16].

2 Materials and Methods

2.1 Fuzzy Predicates

In this subsection, some basic definitions regarding fuzzy predicates logic are given, in order to unify the notation.

Definition #1. A fuzzy predicate p is a linguistic expression (a proposition) with degree of truth μ_p into [0, 1] interval. It applies the "principle of gradualism" which states that a proposition may be both true and false, having some degree of truth (or falsehood) assigned.

Definition #2. A simple fuzzy predicate *sp* is a fuzzy predicate whose degree of truth μ_{sp} can be obtained by some of the next alternatives:

- The application of a membership function associated with a fuzzy term, to a quantitative variable.
- The association of discrete values into the interval [0, 1] to language labels (generally adjectives) of a variable.
- Determination of real value into the [0, 1] interval by an expert.

Definition #3. A compound predicate *cp* is a fuzzy predicate obtained by combination of simple fuzzy predicates or other compound fuzzy predicates, joined by logical connectives and operators (and, or, not, implication, double-implication).

Definition #4. Compound predicates can be represented as a tree structure, having its nodes associated by logical connectives and the successive branches related to lower hierarchical level predicates (simple or compound).

It is needed defining logics where the operations of conjunction, disjunction, order and negation are functions defined over a set of truth values for predicates, into the real interval [0, 1], such that when the truth values are restricted to {0, 1}, these operations become classic Boolean predicates [17].

In the present work, based on previous successful results, we choose some compensatory logic operations: Geometric Mean Based Compensatory Logic (GMCL) and Arithmetic Mean Based Compensatory Logic (AMCL) [17]. We also compare the results with the standard triangular norms (Max-Min). Operations involved in these logics are shown in Tables 1, 2 and 3. In these logics, negation (complement) operation is computed as $N(\mu_i) = 1 - \mu_i$.

Table 1. Operations for Max-Min logic

Logical Operator	Operation
Conjunction	$C(\mu_1,\mu_2,...,\mu_N) = \min(\mu_1,\mu_2,...,\mu_N)$
Disjunction	$C(\mu_1,\mu_2,...,\mu_N) = \max(\mu_1,\mu_2,...,\mu_N)$

Table 2. Operations for Geometric Mean Based Compensatory Logic (GMCL)

Logical Operator	Operation
Conjunction	$C(\mu_1,\mu_2,...,\mu_N) = (\mu_1,\mu_2,...,\mu_N)^{\frac{1}{N}}$
Disjunction	$D(\mu_1,\mu_2,...,\mu_N) = 1-\left[(1-\mu_1)(1-\mu_2)...(1-\mu_N)\right]^{\frac{1}{N}}$

Table 3. Operations for Arithmetic Mean Based Compensatory Logic (AMCL)

Logical Operator	Operation
Conjunction	$C(\mu_1,\mu_2,...,\mu_N) = \left[\min(\mu_1,\mu_2,...,\mu_N).\frac{1}{N}\sum_{i=1}^{N}\mu_i\right]^{\frac{1}{2}}$
Disjunction	$D(\mu_1,\mu_2,...,\mu_N) = 1-\left[\min(1-\mu_1,1-\mu_2,...,1-\mu_N).\frac{1}{N}\sum_{i=1}^{N}(1-\mu_i)\right]^{\frac{1}{2}}$

Compensatory operators are sensitive to the whole set of operands in opposite to the widely used operations Max-Min. In that way, the value of the conjunction and disjunction can be influenced by, and therefore "compensated" for, the value of any of the degrees of truth considered in the operation.

2.2 Self-Organizing Maps

Self-organizing Maps are a very known type of non-supervised and competitive neural network. Inputs have the same data dimension and they are "connected" to each neuron (also called cells), arranged generally in a 2-dimension array [18]. The amount of cells is a design-parameter and it is smaller than the number of training data. Alternatively to the input synaptic weights approach, we can think each cell containing an input-dimension vector. These are known as prototype vectors and they define the SOM codebook [19]. The codebook is initialized using random values or other methods such as linear-initialization.

A cell whose prototype vector is nearer to an input datum, according to a distance criterion, is called Best Matching Unit (BMU).

SOM training is an iterative process. For each input, BMU prototype vector is adapted to be more similar to input, driven by a learning rate. Not only BMU is adapted, but its neighbors are modified too, according a neighborhood function. Both learning rate and neighborhood size are decreased as iterations progress. This is as beginning with a rough training phase that is finer in each training step [18, 20].

In order to evaluate the quality of the trained map, two kinds of errors are considered: the quantization error and the topographic error. They tend to minimize when the map vectors perform an organized projection of the training pattern according to a similarity criterion [21].

Quantization error E_Q is computed as:

$$E_Q = \frac{1}{N}\sum_{i=1}^{N}\left\| x_i - m_i \right\|,$$

where $x_i, i = 1,2...,N$ are the training data, m_i is the BMU corresponding to datum x_i , and N is the number of data. This error allows assessing whether codebook represents training data properly.

Topographic error E_T is helpful to assess whether the data topology was preserved after training, and it is computed as:

$$E_T = \frac{1}{N}\sum_{i=1}^{N} u(x_i),$$

where $u(x_i) = 1$ if the BMU for datum x_i is not adjacent to the second BMU and $u(x_i) = 0$ if it is adjacent. The second BMU is defined as the cell having its prototype vector closest to the datum x_i after the BMU.

In a well-trained SOM, the codebook is a reduced dataset which is representative of training dataset, with a similar probabilistic density function [18, 22].

There are several approaches for visualizing and analyzing the codebook information [20, 23]. For example, component maps [22] allow a detailed analysis of the prototypes vectors, considering the topographic relationship between them.

By running a clustering algorithm on the SOM codebook [24], we can get groups of prototype vectors (and hence, cells), expecting that cells from the same cluster are topographically near [19].

Combining codebook clustering and component maps analysis is useful to "explain" the obtained groups in term of input components. This approach will be detailed in next section.

2.3 Proposed Method

We propose a general approach to obtain membership functions for a clustering system based on fuzzy predicates logical operations: we consider Gaussian functions having their parameters (center and standard deviation) computed from a trained SOM.

General Method: One SOM, *K* Predicates. The first general method stages are:

- Train a SOM using a training dataset taken from the available data till quantization and topographic errors are below tolerance values (training parameters).
- Find *K* cluster center $\{C_i\}_{i=1,2,\ldots,K}$ considering the SOM codebook by some classical clustering algorithm. In this work we applied Fuzzy C-Means (FCM) clustering [25], operating with Euclidian distances, but other algorithms and distances could be used.
- Take centers of Gaussian functions $\{c_{ik}\}_{\substack{i=1,2,\ldots,D\\k=1,2,\ldots,K}}$ for each feature, using the cluster centers computed in the previous step. Compute standard deviations of the Gaussian functions $\{\sigma_{ik}\}_{\substack{i=1,2,\ldots,D\\k=1,2,\ldots,K}}$ considering prototype vectors corresponding to each cluster (feature by feature).

Given a dataset where each datum is a feature vector $[f_1, f_2, \ldots, f_D]$, we will have *K* Gaussian membership functions for each feature f_i, $i = 1,2,\ldots,D$. Let's call them $\{mf_{ik}\}_{\substack{i=1,2,\ldots,D\\k=1,2,\ldots,K}}$.

Now we can create one fuzzy predicate for each class (K compound predicates) by logically operating with the degrees of truth:

$$p_k(d) \equiv mf_{1k}(d) \wedge mf_{2k}(d) \wedge \ldots, \wedge\, mf_{Dk}(d); k = 1,2,\ldots,K$$

where $p_k(d)$ can be linguistically read as "Datum *d* belongs to cluster *k*" and $mf_{ik}(d)$ can be linguistically interpreted as "Feature *i* of datum *d* is near the prototypes belonging to cluster *k*." The nearer the value of feature *i* of datum *d* to the center of the Gaussian function c_{ik} , the higher the degree of truth of $mf_{ik}(d)$. As $mf_{ik}(d)$ are higher, $p_k(d)$ should be higher too, reflecting the fact that if datum *d* is near the cluster center *k*, then datum *d* belongs to cluster *k*.

A datum *d* will be assigned to the class whose predicate has the bigger degree of truth. In addition, if no predicates have degree of truth higher than a determined threshold, *d* could be labeled as "outlier."

Dataset Partitioning: *M* SOMs. In order to capture more accurately the statistical characteristics of the data, we propose a random dataset partition into *M* disjoint subsets, in order to train *M* different SOMs. Despite this partition is random, it would be desirable that each partition is balanced; i.e. there is approximately the same number of data for each class.

Given the fact that data are unlabeled and therefore *M* codebook partitions are non-deterministically generated, we need to identify the *M* clusters that would correspond to the same data class according to their cluster center similarity (by considering Euclidian distances among cluster centers). As a result, *M* data clusters belonging to the same class will have the same cluster index $\{C_{ij}\}_{\substack{i=1,2\ldots,M\\ j=1,2\ldots,K}}$.

We propose three different options to create fuzzy predicates to represent the classes, described as follows.

Option 1: Classifier ensemble (K predicates for each SOM, one decision by SOM). In this option, we consider each SOM and its predicate set as an independent clustering system. We obtain *K* predicates for each SOM (labeled $\{C_{ij}\}_{\substack{i=1,2\ldots,M\\ j=1,2\ldots,K}}$). Given a datum, degree of truth of the *K* predicates for each SOM is obtained, and if required, a crisp cluster assignment is done. The final assigned cluster will be chosen by voting.

Option 2: M independent fuzzy predicates for each class. In this option, we take all predicates generated in the same way as the previous option. We obtain *M* predicates for each class. Given a datum, degrees of truth of the *K*x*M* predicates are obtained. If a crisp clustering is wanted, cluster assignment is done by taking the one represented for the predicate whose degree of truth is the maximum.

Option 3: A unique fuzzy predicate for each class. In this option, we explore the advantages of using predicate logic. We define a unique predicate for each class using the "or" connective to consider simple predicates for each feature.

$$\begin{aligned}
p_k(d) &\equiv \left[mf_{1k1}(d) \vee mf_{1k2}(d) \vee \ldots \vee mf_{1kM}(d)\right] \\
&\wedge \left[mf_{2k1}(d) \vee mf_{2k2}(d) \vee \ldots \vee mf_{2kM}(d)\right] \wedge \ldots \\
&\wedge \left[mf_{Dk1}(d) \vee mf_{Dk2}(d) \vee \ldots \vee mf_{DkM}(d)\right];\ k = 1,2,\ldots,K
\end{aligned}$$

Given a datum, degrees of truth of the *K* compound predicates are obtained. Like previous cases, cluster assignment is done by taking the cluster represented for the predicate whose degree of truth is the maximum.

Results would be different depending on the way the predicates are defined. In addition, there are some other parameters that should be chosen for each particular problem:

- SOM dimension: the number of SOM cells should be indicated, though it can be automatically determined based on the number of data *N* or applying some growing algorithm [26].

- Number of data partitions *M*: this should be chosen according to the number of data, taking into account that each SOM should be able to capture the characteristics of data, so it shouldn't deal with only a few training data.
- Type of logic chosen to compute the logical operations: compensatory fuzzy logics (GMCL or AMCL) or Max-Min logic.
- Number of classes *K*: defined by the problem to solve.
- Cluster algorithm used to partition the SOM codebooks.
- Type of distances used in the cluster algorithm.

Method Validation: Assessing the Generalization Abilities. In order to assess the generalization abilities of the different approaches, we applied k-fold cross-validation. In this method, the original dataset is randomly partitioned into *k* data-subsets (folds). Alternatively, one fold is taken as test data and the remaining *k*-1 folds are taken as training data. Classification error is computed on each iteration using the test data. The final classification error is estimated by computing the average of the *k* folds errors.

In this validation method, all elements of the dataset are used for validation exactly once. The number of folds *k* is heuristically chosen taking into account the number of data available. Typically *k*=10 is a good first choice, but in dataset with a few data it could generate folds with too few elements.

3 Results

To compare the different approaches given for the proposed paradigm, a lot of experiments were run. We focused in assessing confusion matrices and clustering accuracy for each test, defined as the ratio between number of data assigned correctly to different clusters to the number of test data (the ratio between the sum of the main diagonal of the confusion matrix to the sum of its elements). Accuracy is given after averaging 10 runs of a k-fold validation scheme, with *k*=10. This averaging allows getting rid of any randomness.

We tested the method for this preliminary work using three well-known datasets: Iris data (3 classes, 4 features, 150 data) [27], Wine data (3 classes, 12 features, 4898 data) [28], and 12000 pixels randomly selected from simulated magnetic resonance images (MRI) (4 classes, 3 features) [29]. We compare values obtained using different SOM sizes, SOM number (*M*), and logical operations. Clustering accuracies obtained are shown in Table 4.

Table 4. Best clustering accuracies (k-fold, *k*=10) obtained in test datasets: Iris (4 features, 3 classes), Wine (12 features, 3 classes) and MRI (3 features, 4 classes).

Dataset	SOM number (*M*)	SOM size	SOM Clustering	Option 1	Option 2	Option 3
Iris	1	Auto	0.919	**0.941**	**0.941**	**0.941**
Wine	2	10 x 10	0.927	0.966	**0.977**	0.961
MRI	20	20 x 20	0.865	**0.884**	**0.871**	**0.884**

Keeping SOM sizes fixed and *M*=10 (or *M*=5 for iris data) we can conclude that GMLC and AMLC give similar results in these datasets, and they give very much better results than Max-Min logic operations (results were at least 20 % worse using this logic). That is why these experiments results are not shown in detail in Table 4.

To help comparing the obtained accuracies, we show graphs in Fig. 1, 2 and 3. Experiments show that increasing SOM size is better when there are a big dataset (Fig. 3). However, keeping suggested automated SOM size (having different sizes in different data partitions) seems to be a valid conservative option. Horizontal lines mark the accuracy obtained using only SOM clustering without any predicates analysis.

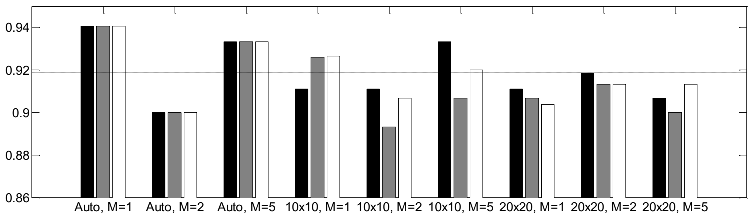


Fig. 1. Accuracy for Iris dataset Iris (4 features, 3 classes). Bars indicate the different ways to make predicates (black, option 1; gray, option 2; white, option 3). SOM sizes (or automatic size) and number of data partitions (*M*) are indicated for each bar group. Horizontal line marks the accuracy obtained using only SOM clustering.

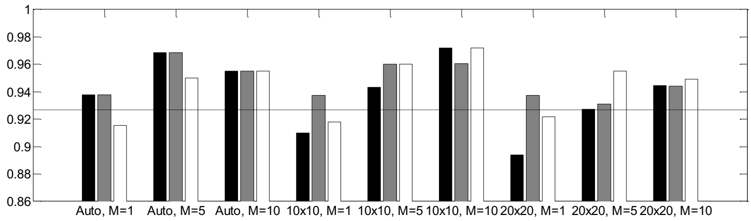


Fig. 2. Accuracy for Wine dataset (12 features, 3 classes). Bars indicate the different ways to make predicates (black, option 1; gray, option 2; white, option 3). SOM sizes (or automatic size) and number of data partitions (*M*) are indicated for each bar group. Horizontal line marks the accuracy obtained using only SOM clustering.

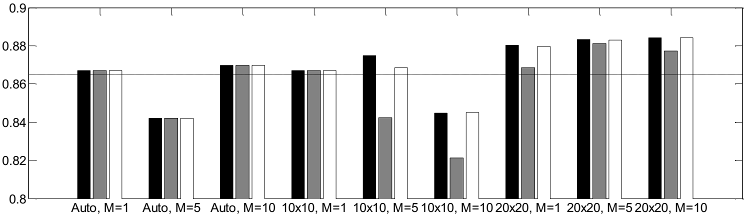


Fig. 3. Accuracy for MRI dataset (3 features, 4 classes). Bars indicate the different ways to make predicates (black, option 1; gray, option 2; white, option 3). SOM sizes (or automatic size) and number of data partitions (*M*) are indicated for each bar group. Horizontal line marks the accuracy obtained using only SOM clustering.

Partitioning datasets into more groups (*M* parameter) improves the clustering quality, but we must keep in mind that bigger partitions will lead to less data in each SOM.

Choosing option 3 gives best results in several cases, in particular the MRI dataset (the bigger one) compared with option 2. This suggests that the approach using only one composed predicate explaining each class is a good choice. That is why we are advancing in this way, searching other integration schemes. Instead the use of OR operation to aggregate the "opinion" of different SOMs, we are about to apply some Type-2 Fuzzy Logic basics. Assessing the algorithm performance with a broader range of datasets, including more complex datasets is also pending.

4 Conclusions

We proposed the use of the fuzzy predicates logic created by means of Self-organizing Maps to tackle ranked clustering problems. We gave several alternatives to configure the interpretation of the memberships for the clustering process. The proposed system is self-designed and it has the advantages of other fuzzy systems.

Results are interesting because the clustering accuracy is high when crisp clusters are required. The proposal must be tested with more real datasets, though the preliminary results obtained in known datasets suggest that it is a promising clustering paradigm. If further analysis is made, it is possible giving some linguistic interpretation to the predicates obtained by interpretation of membership functions.

This is a preliminary approach to be extended to be used with Type-2 Fuzzy Logic paradigm, which is proposed as immediate future work. Some steps in this way are already successfully developed.

References

1. Pizzi, N.J., Pedrycz, W.: Classifying High-Dimensional Patterns Using a Fuzzy Logic Discriminant Network. Advances in Fuzzy Systems 2012, 7 (2012)
2. Zadeh, L.A.: Fuzzy sets. Information and Control 8, 338–353 (1965)
3. Zadeh, L.A.: The Concept of a Linguistic Variable and its Application to Approximate Reasoning. Information Sciences 8, 199–249 (1975)
4. Thawonmas, R., Abe, S.: Function approximation based on fuzzy rules extracted from partitioned numerical data. Trans. Sys. Man Cyber. Part B 29, 525–534 (1999)
5. Kahramanli, H., Allahverdi, N.: Rule extraction from trained adaptive neural networks using artificial immune systems. Expert Syst. Appl. 36, 1513–1522 (2009)
6. Bortolan, G., Pedrycz, W.: An interactive framework for an analysis of ECG signals. Artif. Intell. Med. 24, 109–132 (2002)
7. Nomura, T., Miyoshi, T.: An adaptive rule extraction with the fuzzy self-organizing map and a comparison with other methods. In: Proceedings of the 3rd International Symposium on Uncertainty Modelling and Analysis (1995)
8. Drobics, M., Bodenhofer, U., Winiwarter, W.: Mining Clusters and Corresponding Interpretable Descriptions - A Three-Stage Approach. Expert Systems: The Journal of Knowledge Engineering and Neural Networks 19, 224–234 (2002)
9. Malone, J., McGarry, K., Wermter, S., Bowerman, C.: Data mining using rule extraction from Kohonen Self-organising maps. Neural Computing and Applications 15, 9–17 (2006)

10. Pal, N.R., Laha, A., Das, J.: Designing fuzzy rule based classifier using Self-organizing feature map for analysis of multispectral satellite images. International Journal of Remote Sensing 26, 2219–2240 (2009)
11. Sorayya, M., Aishah, S., Sapiyan, B.M., Sharifah Mumtazah, S.A.: A Self organizing map (SOM) guided rule based system for freshwater tropical algal analysis and prediction. Scientific Research and Essays 6, 5279–5284 (2011)
12. Espin Andrade, R., Mazcorro Téllez, G., Fernández González, E., Marx-Gómez, J., Lecich, M.I.: Compensatory Logic: a fuzzy normative model for decision making. Revista Investigación Operacional 27, 178–193 (2006)
13. Dubois, H., Prade, D.: Fuzzy Sets and Systems: Theory and Applications. Academic Press Inc., New York (1980)
14. Pedrycz, W.: Fuzzy clustering with a knowledge-based guidance. Pattern Recognition Letters 25, 469–480 (2004)
15. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11, 586–600 (2000)
16. Kiang, M.Y.: Extending the Kohonen self-organizing map networks for clustering analysis. Computational Statistics Data Analysis 38, 161–180 (2001)
17. Bouchet, A., Pastore, J.I., Espín Andrade, R., Brun, M., Ballarin, V.: Arithmetic Mean Based Compensatory Fuzzy Logic. International Journal of Computational Intelligence and Applications 10, 231–243 (2011)
18. Kohonen, T.: Self organized formation of topological correct feature maps. Biol. Cybernetics 43, 59–96 (1982)
19. Meschino, G.J., Passoni, L.I., Scandurra, A.G., Ballarin, V.L.: Representación automática pseudo color de imágenes médicas mediante Mapas Autoorganizados. In: Simposio Argentino de Informática y Salud, SIS 2006, Ciudad de Mendoza, Argentina (2006)
20. Vesanto, J.: Data exploration process based on the Self–organizing map (2002)
21. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice Hall, Nueva Jersey (1999)
22. Meschino, G.J., Passoni, L.I., Scandurra, A.G.: Análisis de Datos Multivariados de Pacientes Diabéticos Internados con Redes Neuronales Autoorganizadas. Investigación Operativa 24, 73–85 (2004)
23. Vesanto, J.: SOM-Based Data Visualization Methods (1999)
24. Jain, A.K., Murty, M.N., Flynn, P.J.: Data Clustering: A Review. ACM Computing Surveys 31, 264–323 (1999)
25. Ruspini, E.H.: A new approach to clustering. Information and Control 15, 22–32 (1969)
26. Forti, A., Foresti, G.L.: Growing hierarchical tree SOM: an unsupervised neural network with dynamic topology. Neural Network 19, 1568–1580 (2006)
27. Fisher, R.A.: The Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics 7, 179–188 (1936)
28. Cortez, P., Cerdeira, A., Almeida, F., Matos, T., Reis, J.: Modeling wine preferences by data mining from physicochemical properties. Decision Support Systems 47, 547–553 (2009)
29. Kwan, R.K.-S., Evans, A.C., Pike, G.B.: MRI simulation-based evaluation of image-processing and classification methods. IEEE Transactions on Medical Imaging 18, 1085–1097 (1999)

From Forced to Natural Competition in a Biologically Plausible Neural Network

Francisco Javier Ropero Peláez[1] and Antonio Carlos Godoi[2]

[1] Center of Mathematics, Computation and Cognition, Universidade Federal do ABC
francisco.pelaez@ufabc.edu.br
[2] Department of Telecommunications and Control Engineering, Universidade de São Paulo
antoniogodoi@usp.br

Abstract. In this paper we propose a new unsupervised neural network whose units exhibit intrinsic plasticity and metaplasticity. We describe three versions of the network: The first version is a two-layered neural network with intrinsic plasticity governing the shifting of the activation function, and the pre-synaptic rule altering synaptic weights. In this first version, competition is forced, so that the most activated neuron is set to one and the others to zero. In the second version, competition is not forced and occurs naturally due to inhibition between second layer's neurons. Competition also occurs naturally in the third version whose architecture resembles the one of the internal granular layer of the koniocortex. All versions of our network categorize input patterns similarly to a conventional competitive neural network.

Keywords: learning, competition, intrinsic plasticity, pre-synaptic rule.

1 Introduction

Competition between neurons is used in unsupervised neural networks for classification purposes. Competition means that, inside a neuron's pool, the most activated neuron is the only one that remains active after comparing all neurons' activations.

Competition is a frequently mentioned strategy for explaining auto-organization and information mapping in biological organisms [8]. However, the way competition is accomplished, by calculating the maximum output value among the neurons in the pool, is far from being biological. This algorithmic solution is not present in biology where competition emerges from the dynamic interaction between neurons. Competition in biological systems seems to be correlated to lateral inhibition. However, although lateral inhibition is mentioned regarding competition even in seminal neural network treatises [9,10 p. 190], it seems that, besides lateral inhibition, other correlated biological factors should also be considered for producing competition and pattern classification. In this sense, we consider biological factors like metaplasticity [1, 2] and intrinsic plasticity [4] that are able to make competition possible.

These biological mechanisms are gradually introduced along the different versions of the neural network proposed in this article. Initially we introduce a completely non-biological framework, the "Bayesian Decision Network" (see Fig.1.a) that was used for introducing some algebraic concepts used in the following versions of the

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 95–104.
DOI: 10.1007/978-3-642-35230-0_10

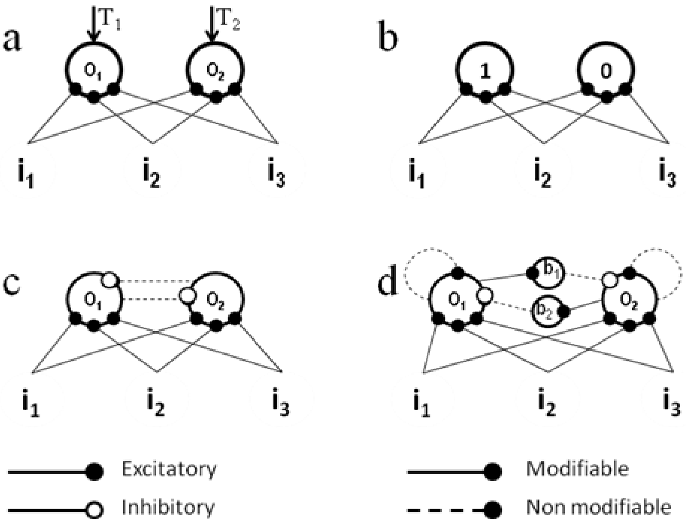


Fig. 1. The Bayes Decision network (a) and the three versions of the network (b,c,d) using probabilistic weights and intrinsic plasticity. In b) there is still forced competition in which the maximally stimulated output-neuron is set to 1 and the remaining neurons to zero. In c) competition takes place through lateral inhibition with non-modifiable connections. In d) the network architecture resembles internal granular layer of the koniocortex. In the koniocortex granular layer o_i neurons correspond to spiny stellate neurons, i_i neurons to thalamo-cortical neurons and b_i neurons to inhibitory interneurons.

network. A truly competitive version of the network is depicted in Fig. 1.b.where competition is not yet biological, but externally driven (by calculating which neurons is the most active). Its neurons posses the biological property called intrinsic plasticity (see section 2). This network performs classification tasks as any competitive network. The network in Fig. 1.c, called the "lateral inhibition network", also performs classification tasks. In this case, intrinsic plasticity, metaplasticity and lateral inhibition are orchestrated to allow competition in a biologically plausible way. Finally the network represented in Fig.1.d performs competition and pattern classification similarly to the previous "lateral inhibition network". Here, inhibitory interneurons substitute inhibitory lateral connections. Because of the similarity of this network to the first cortical layer receiving sensory inputs, it was called the Koniocortex-like network.

2 Methodology

In all versions of the network, we used a rate-code neuron model whose bounded output (between 0 and 1) represents the probability of occurrence of an action potential. The net-input of neuron j is calculated by the inner product of neuron's j weights and the normalized input pattern $\vec{i} = \vec{I} / \|\vec{I}\|$ (lower case notation meaning vector normalization). The type of normalization used here is the l_1-norm in which:

$$\left\|\vec{I}\right\| = \sum_{i=1}^{n} \left|I_i\right| \tag{1}$$

The weights to a certain neuron can be considered the components of a vector prototype $\vec{T}^j$ so that $\vec{T}^j = \vec{W}^j = [W_{j1}, W_{j2} \ldots W_{jn}]$. In this way, the net-input of neuron O^j can be calculated as $net_j = \left\|\vec{W}^j \cdot \vec{i}\right\| = \left\|\vec{T}^j \cdot \vec{i}\right\| = \left\|\vec{I}_{\vec{T}^j}\right\|$ which is the modulus of the projection $\left\|\vec{I}_{\vec{T}^j}\right\|$ of the input pattern $\vec{I}$ over prototype $\vec{T}^j$.

For altering synaptic weights, the incremental version of the presynaptic rule, was used:

$$\Delta w = \xi I (O - w) \tag{2}$$

where I and O are the presynaptic and postsynaptic action potential probabilities, respectively, and ξ a small positive constant.

As explained elsewhere [12], the presynaptic rule is not only able to reproduce the empirical plasticity curve obtained by Artola et al. [3] relating postsynaptic voltage to the increment of synaptic weight (Fig.2.a). The presynaptic rule is also able to reproduce metaplasticity [1, 2] which elongates the plasticity curves rightwards for higher initial synaptic weights as depicted in Fig. 2.a. Fig.2.b shows the computer simulation [12] of the presynaptic rule, yielding a family of curves which are similar to biological plasticity and metaplasticity curves. Regarding the activation function, a conventional sigmoid was used for relating the net-input of neuron O^j to its output value O_j:

$$O_j = \frac{1}{1 + e^{-k(net_j + 0.5 - 2s^j)}} \tag{3}$$

were s^j is the shift of the activation function and k is a curve-compressing factor that was set to 25. The range of s^j is $0<s^j<1$. For $s^j=0$ the sigmoid is completely shifted leftwards so that for $net_j=0$, $O_j=1$. In the case s^j=1 the sigmoid is completely shifted rightwards so that for net_j=1 the output value of the sigmoid is $O_j=0$.

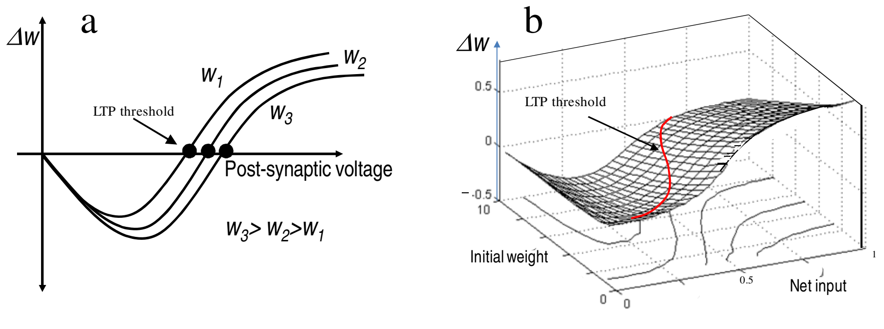


Fig. 2. Experimental synaptic plasticity, and its modeling through pre-synaptic rule. a) Metaplasticity [1, 2]: For higher values of the weight, the plasticity curve is more elongated to the right b) The probabilistic version of the presynaptic rule was computationally simulated [12]. It not only exhibits the same shape of biological obtained curves, but also exhibits metaplasticity that shifts the Long Term Potentiation Threshold (LTP threshold).

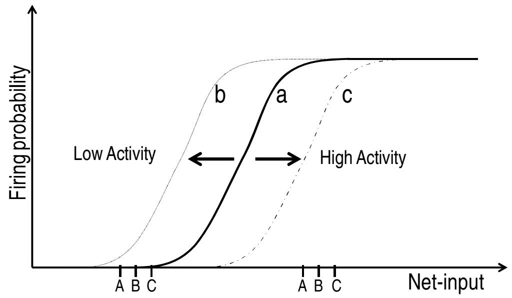


Fig. 3. Intrinsic plasticity of real neurons allows the neuron activation function to be shifted leftwards or rightwards so that this activation function is placed over intervals corresponding to the average activation of the neuron. a) Original position of the activation function (sigmoid). b) If the average net-input of the neuron is low (as in case of inputs A, B and C) intrinsic plasticity gradually drives the activation function to the left, allowing post-synaptic variations be translated into variations of firing probability. c) If the average net-input is high (as in D, E and F) the activation function is gradually shifted to the right and the neuron, instead of being saturated all the time, has a variable firing probability that follows the variations of net-input.

Real neurons also exhibits a property that levels the firing probability of neurons engaged in competition (see Fig. 3), making very active neurons moderate their activity and inactive neurons become more active. According to this property called intrinsic plasticity, the activation function gradually shifts rightwards or leftwards thus leveling the activation of highly or scarcely activated neurons, respectively. An important parameter of the activation function of neuron Oj is the value of its rightward shift, s^j, so that the activation function is better expressed as:

$$O^j = f\left(\left\|\vec{I}_{\vec{T}^j}\right\|, s^j\right). \tag{4}$$

The following equation calculates the shift of the activation function, *s* at time *t* in terms of the shift and output probability of the neuron at time t-1.

$$s_t^j = \frac{\upsilon \cdot O_{t-1} + s_{t-1}^j}{\upsilon + 1}, \tag{5}$$

where υ, is a small arbitrary factor that adjusts the shifting rate of the activation function. In the case of highly activated neurons, s^j increases, making the activation function shifts rightwards so that the output of the neuron will be down-regulated in the future. In the case of less active neurons, the activation function shifts leftwards, and the neuron increases its firing.

3 Results

In this section we will show the evolution of the network from a Bayesian network in which intrinsic plasticity is not necessary to a koniocortex-like network in which pattern categorization is obtained without any kind of supervision. In this latter, competition and categorization emerges as the result of the dynamics of the individual neurons of the network without the need of any kind of operation involving more than one neuron.

3.1 "Bayes Decision Rule" Network

The purpose of presenting this case is to establish some mathematical foundations for dealing with the different versions of the network. Here, as in the following network versions, input patterns are normalized by dividing each of the inputs by the l_1 norm (the sum of the input). When an input pattern belongs to a certain category, let's call it T^j, neuron O_j is forced to fire and the remaining neurons keep silent. Along the presentation of patterns of a certain category, the weights are modified so that they converge to the prototype $\vec{T}^j$ of that specific category. Afterwards, when a testing input pattern, $\vec{I}_{test}$, is projected over the different prototypes (performing the inner product of the normalized input vector by the prototype's weights), the neuron with highest output, $O*$, fires, indicating to which class $\vec{I}_{test}$ belongs to.

$$O^* = O^j \,/\forall k \neq j \quad \left\| \vec{I}_{test_{\vec{T}^j}} \right\| > \left\| \vec{I}_{test_{\vec{T}^k}} \right\| \tag{6}$$

Previous expression is similar to the Bayes decision rule, a classical rule used in pattern recognition for deciding the class, T^j, to which a certain pattern belongs to [5].

$$O^* = O^j \,/\forall k \neq j \quad P\left(T^j / \vec{I}_{test}\right) > P\left(T^k / \vec{I}_{test}\right) \tag{7}$$

3.2 Forced Competition Network

In this case the process of neuron´s activation is unsupervised, differently from previous case in which categories were imposed through the activation of certain output neurons. Here, a "winner takes all" operation compares the outputs of the different neurons, setting the winning neuron to one, and the remaining neurons to zero. Differently from the following cases, this result is algorithmic and does not emerge as the result of the dynamic interaction among neurons.

The Forced Competition Network is illustrated in Fig. 4.a in which input patterns are normalized through the l_1-norm (lower-case letters representing normalized inputs) and weights to each neuron are represented either as prototype vectors $\vec{T}^i$ or weight vectors $\vec{W}^i$. Above each neuron, the activation curves represent the preliminary output O_j of neuron O^j, in terms of its net-input ($\left\| \vec{I}_{\vec{T}^j} \right\|$, according to section 2). Here, competition takes place through an altered version of the previously explained Bayes algorithm in which comparison is performed between the outputs of activation functions.

$$O^* = O^j \,/\, \forall k \neq j,\ f\left(\left\|\vec{I}_{\vec{T}^j}\right\|, s^j\right) > f\left(\left\|\vec{I}_{\vec{T}^k}\right\|, s^k\right) \tag{8}$$

Notice that in this equation, activation functions are shifted according to parameter *s* (recall section 2). When a pattern $\vec{I}^1$ is input to the network (Fig.4.b.1), its projections over prototypes $\vec{T}^1$, $\vec{T}^2$ and $\vec{T}^3$ (Fig.4.b.2) are calculated. According to equation 8, applying the activation function to these projections and calculating the greatest output we obtain the neuron that wins the competition. In this example, the winning neuron O^* is O^1, and its output is set to one, while the other neurons' outputs are set to zero. This kind of competition is driven by an external algorithm which evaluates which is the higher output neuron.

Before projecting pattern $\vec{I}^2$ over prototypes $\vec{T}^j$ (Fig.4.c.2), notice that prototypes $\vec{T}^j$ have changed. This is because, according to the pre-synaptic rule, weights w_{1j} from active inputs i_j to O^1 (the winning neuron) increases, while weights from active inputs to non-active neurons, O^2 and O^3, are reduced. Weights from null inputs to non-active neurons remain the same. The result of this weight changing process is that vector $\vec{T}^1$ evolves towards $\vec{I}^1$ and vector $\vec{T}^2$ and $\vec{T}^3$ evolves towards a plane (in gray) orthogonal to $\vec{T}^1$. In a case with more neurons in the second layer, all non-winning $\vec{T}^j$ move towards a plane that is orthogonal to the winning prototype. Fig.4.c.1 represents the situation of the weights just before a second input-pattern $\vec{I}^2$ is presented to the network. Previously reinforced connections are thicker and, due to intrinsic plasticity, O^1 activation curve is shifted rightwards, while O^2 and O^3 curves are shifted leftwards.

When applying a second pattern $\vec{I}^2$ to the network, and due to the previous reinforcement of O^1 neuron's weights, the projection of $\vec{I}^2$ over $\vec{T}^1$ is again greater than the projection over $\vec{T}^2$ (see Fig. 4.c.2), so that O^1 (in gray) wins the competition once more. Figure 4.d.1 and 4.d.2 show that the weights of neuron O^1 increased significantly after several presentations of $\vec{I}^1$ and $\vec{I}^2$. Conversely, O^2 weights decrease along time. With higher weights, neuron O^1 always wins the competition unless something else takes place. Intrinsic plasticity, by shifting the activation function, helps other neurons to win, making neuron O^1 less active and neurons O^2 and O^3 more active. This allows that $O_2 = f\left(\left\|\vec{I}^2_{\vec{T}^2}\right\|, s^2\right)$, becomes greater than any other $O_j = f\left(\left\|\vec{I}^2_{\vec{T}^j}\right\|, s^j\right)$ so that O^1 eventually fails to win the competition that is won, in this in this case, by neuron O^2, as depicted in Figure 4.e.1

Figures 4.f.1 and 4.f.2 exhibit the value of the weights of the network after many presentations of patterns $\vec{I}^1$ and $\vec{I}^2$. When pattern $\vec{I}^2$ causes O^2 winning the competition, weights from non-zero inputs to neuron O^2 increment their value so that $\left\|\vec{T}^2\right\|$ becomes greater and $\vec{T}^1$ and $\vec{T}^3$ progressively tend to be orthogonal to $\vec{T}^2$.

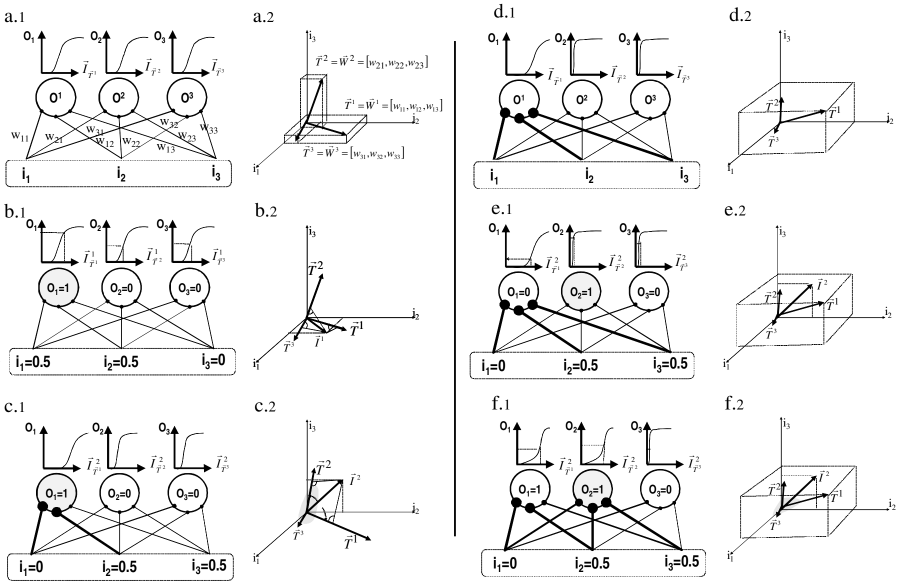


Fig. 4. Example of a forced competitive process in a biologically realistic neural network.: (a) The network is initialized with random weights. The shifts of the activation functions of neurons T_1 and T_2 are also initialized. (b) For calculating the projection $\vec{I}^{1}_{\vec{T}^j}$ of an input pattern $\vec{I}^1$over the different prototypes, the inner product of its normalized version $\vec{i}^1$ and the different weight vectors is calculated. The result of applying the activation functions over projections $\vec{I}^{1}_{\vec{T}^1}$ and $\vec{I}^{1}_{\vec{T}^2}$ yields the highest activated neuron which is, in this case, O^1. (c) Weights are slightly altered through the pre-synaptic rule and a new pattern $\vec{I}^2$ is presented to the network. Due to its previously reinforced weights, O^1 wins the competition again. (d) Situation of the weights after presenting $\vec{I}^1$ and $\vec{I}^2$several times. (e) Intrinsic plasticity allows an adjustment of the shift of the activation function of both neurons so that the activation of the less active neurons increases and vice-versa, allowing neuron O^2 win the competition when pattern $\vec{I}^2$ is presented again. (f) Situation of the weights and activation functions after many presentations of $\vec{I}^1$ and $\vec{I}^2$.

Along this process, each prototype is selected to represent each category of the input data. This neural network was used for different purposes like identifying the direction of moving objects [7], analyzing the illusion of movement in static images [11] or controlling a self-learning robot [13].

3.3 Lateral Inhibition Network

This case was represented in Fig.5. In this and in the koniocortex-like network there is not “forced competition” in the sense that there is not an explicit calculation of the most activated neuron in order to perform a winner-takes-all operation. Here the most activated neuron emerges from the internal dynamic of the network in which each neuron acts without any kind of external supervision. The process is as follows: Initially (Fig.5.a) weights are random and small. An activation function yields a preliminary output value that will be effective if it is higher than a "hard limit" threshold (HL, see horizontal line). Due to the small initial weights, no output reaches the HL so that all outputs are zero. A null output in all neurons makes all sigmoids shift leftwards ($s_t^j = s_{t-1}^j/(\upsilon+1)$) so that the most active neuron eventually fires (O^1 in the example Fig. 5.b) precluding remaining neurons to fire due to lateral inhibition. However, the winning neuron will not be permanently the winner because, once a certain neuron wins its sigmoid tends to shift rightwards according to intrinsic plasticity. All considerations regarding synaptic weights that were explained in the "Forced Competition" case serve also for this case. Note that inhibitory connections do not undergo weight variation (in this case inhibitory connections' weights were set to 1).

Regarding other parameters υ (see Eq.5) was fixed to a lower value than ξ in Eq.2 ($\upsilon = 0.01$, $\xi = 0.1$). This network performs the same pattern classification tasks performed by the "Forced Competition" version previously explained.

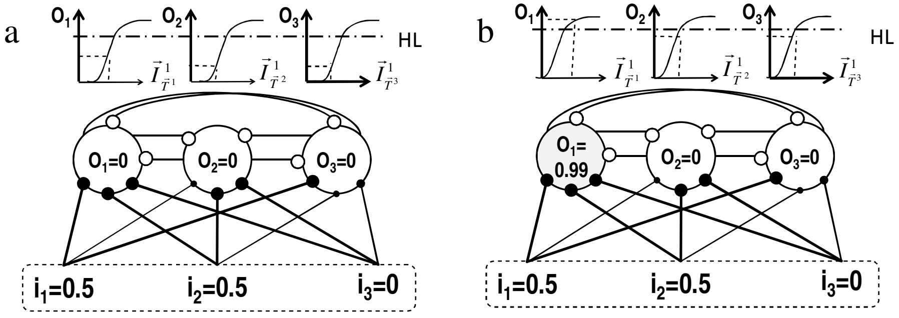


Fig. 5. Competition in a network with intrinsic plasticity, lateral inhibition and a hard limit (HL) threshold: a) Even when no neuron has yet reached the firing threshold they experiment intrinsic plasticity so that b) sigmoids shift leftwards until one of the neurons fires.

3.4 Koniocortex-Like Network

Active neurons in previous network are inhibitory. However, biological relay neurons linking brain layers or brain areas are usually excitatory, being inhibition usually locally restricted inside layers or areas. For understanding how lateral inhibition is performed, not directly, but through inhibitory intermediate neurons, we developed a model that resembles the architecture of the granular layer of the so-called koniocortex (Fig. 6) which is the first cortical layer that receives thalamic inputs. This model performs categorization tasks without any "external" algorithm for determining which neuron wins the competition. Fig. 6.a shows the architecture of this network, and Fig.6.b the result of a simulation in which three types of patterns are input to the network: type A (TA), type B (TB) and type C (TC). Each epoch in the example consists in nine patterns organized according to types as follows TA-TB-TC-TA-TB-TC-TA-TB-TC. As can be seen in Fig.6.b, the network is capable of identifying these three categories of patterns. Neurons 4, 5 and 6, that are similar to koniocortex spiny stellate cells, activate in different moments, as if a conventional winner-takes-all process was taking place. Each one of these neurons is active for one specific category of input patterns. This network is not very different from the Lateral Inhibition Network. The only difference is that direct inhibition is substituted by a neuron mediated inhibition. In this case, inhibitory neurons also have intrinsic plasticity for shifting their sigmoidal activation functions. Since each time a "spiny stellate cell" is active, it activates one corresponding inhibitory neuron, the sigmoids of these two connected neurons (spiny stellate and inhibitory) have a similar shift. In this way, although inhibitory connections are not modifiable, inhibition is modulated by intrinsic plasticity. Although the detailed explanation of the similarities between this network and the granular layer of the koniocortex [6] is out of the scope of this preliminary work, we can anticipate that both networks are formed by a layer of excitatory neurons with modifiable synapses and surrounded by inhibitory cells. In both networks excitatory neurons have recurrent connections [6]. Firing adaptation (intrinsic plasticity) is another characteristic that are present in the biological and artificial networks. As mentioned in the introduction, this type of network reveals that competition for pattern classification emerges from the inner dynamic of a network without the necessity of algorithmically determining which is the most activated neuron.

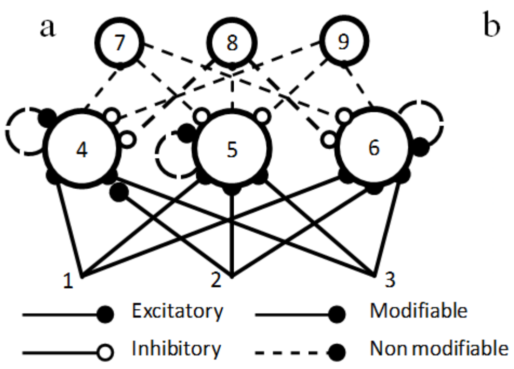


Fig. 6. Neural network model of koniocortex granular layer: a) 1, 2 and 3 represent thalamic input patterns. 4, 5 and 6 represent spiny stellate cells with recurrent connections. 7, 8 and 9 represent smooth basket cells. b) Matlab simulation of the network: Two epochs of nine input patterns are presented to the network. Each numbered ribbon represents the activity of each of the corresponding neuron with the same number at the left.

4 Conclusions

In this work we showed that biological properties (like metaplasticity, intrinsic plasticity and lateral inhibition), when properly orchestrated in an appropriate neural architecture, are able to give rise to complex emerging behaviors like a winner-takes-all computation leading to pattern classification. For a step by step explanation, this network was developed across several stages: from a very simple Bayes Decision model for establishing basic algebraic principles, through a "Forced Competition Network" and a "Lateral Inhibition Network" to a more complex koniocortex-like network. Competition and pattern classification naturally emerge from the internal dynamics of both the lateral-inhibition and konio-cortex network, without the necessity of executing any externally driven algorithm.

References

1. Abraham, W.C., Bear, M.F.: Metaplasticity: the plasticity of synaptic plasticity. Trends in Neuroscience 19, 126–130 (1996)
2. Abraham, W.C., Tate, W.P.: Metaplasticity: a new vista across the field of synaptic plasticity. Progress in Neurobiology 52, 303–323 (1997)
3. Artola, A., Brocher, S., Singer, W.: Different voltage-dependent threshold for inducing long-term depression and long-term potentiation in slices of rat visual córtex. Nature 347, 69–72 (1990)
4. Desai, N.S., Rutherford, L.C., Turrigiano, G.G.: Plasticity in the intrinsic excitability of cortical pyramidal neurons. Nature Neurosciences 2, 515–520 (1999)
5. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. John Wiley and Sons, Inc., New York (1973)
6. Hirsch, J.A.: Synaptic integration in layer IV of the ferret striate cortex. Journal of Physiology 483(1), 183–199 (1995)
7. Kinto, E.A., Del Moral Hernandez, E., Marcano, A., Ropero Peláez, F.J.: A Preliminary Neural Model for Movement Direction Recognition Based on Biologically Plausible Plasticity Rules. In: Mira, J., Álvarez, J.R. (eds.) IWINAC 2007. LNCS, vol. 4528, pp. 628–636. Springer, Heidelberg (2007)
8. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics 43, 59–69 (1982)
9. McClelland, J.L., Rumelhart, D.E., The PDP Research Group: Parallel distributed processing: Exploration in the microstructure of cognition. MIT Press, Cambridge (1986)
10. McClelland, J.L., Rumelhart, D.E.: Explorations in parallel distributed processing. MIT Press, Cambridge (1988)
11. Ropero Peláez, F.J., Ranvaud, R., Szafir, S., Ramírez-Fernández, F.J.: The illusion of movement in static images analyzed with a biologically plausible unsupervised neural network model. In: Proceedings of the Brain Inspired Cognitive Systems, BICS 2008, São Luiz (2008)
12. Ropero Peláez, J., Godoy Simoes, M.: A computational model of synaptic metaplasticity. In: Proceedings of the International Joint Conference of Neural Networks 1999, Washington DC (1999)
13. Ropero Peláez, F.J., Santana, L.G.R.: Doman's Inclined Floor Method for Early Motor Organization Simulated with a Four Neurons Robot. In: Ferrández, J.M., Álvarez Sánchez, J.R., de la Paz, F., Toledo, F.J. (eds.) IWINAC 2011, Part I. LNCS, vol. 6686, pp. 109–118. Springer, Heidelberg (2011)

Sparse Coding Neural Gas Applied to Image Recognition

Horia Coman[1,2], Erhardt Barth[1], and Thomas Martinetz[1]

[1] Institute for Neuro- and Bioinformatics, University of Lübeck
Ratzeburger Allee 160, 23538 Lübeck, Germany
{coman,barth,martinetz}@inb.uni-luebeck.de
http://www.inb.uni-luebeck.de
[2] LAPI, The "POLITEHNICA" University of Bucureşti
Splaiul Independenţei nr. 313, 060042 Bucureşti, Romania
http://imag.pub.ro

Abstract. A generalization of the Sparse Coding Neural Gas (**SCNG**) algorithm for feature learning is proposed and is then discussed in the context of modern classifier techniques for images. Two versions are presented. The latter obtains faster convergence by exploiting the nature of particular feature coding methods. The algorithm is then used as part of a larger image classification system, which is tested on the MNIST handwritten digit dataset and the NORB object dataset, obtaining results close to state-of-the-art methods.

Keywords: Neural Gas, Sparse Coding, Sparse Coding Neural Gas, Image Recognition, Matching Pursuit.

1 Introduction

The task of image recognition is a complex one. Simply training a classifier on raw image data will yield poor performance. A common strategy is to look at important properties of an image, called *features*. Then, given an image **I**, these features are used to compute a feature descriptor **F**. This contains, for each feature, a measure of the inclusion of that feature into the image. Then, in order to obtain an estimated class, only **F** is considered. The construction of features is thus a big part of most machine learning applications. However, for best performance, specific domain knowledge must be used. This makes both the design and comparison of learning systems harder.

In the last decade, automatic feature construction, also known as unsupervised feature learning [1–4], has become mainstream, surpassing hand-crafted methods on diverse problems [1, 5–10]. These techniques aim to build features by looking at the statistical properties of a dataset. In addition to this, a full framework for classification has been refined, based on the model of Convolutional Neural Networks (**CNNs**) [6], which has parallels with the structure of the V1 area of the mammalian brain.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 105–114.
DOI: 10.1007/978-3-642-35230-0_11

This paper studies a particular type of feature learning method, called Sparse Coding Neural Gas [11–14], on two tasks of image classification. The algorithm itself is an adaptation of the Neural Gas algorithm [15, 16].

2 Overview of Feature Extraction

Given an image $\mathbf{I} \in \mathbb{R}^{m \times n}$, the conceptual recognition pipeline can be summarized as:

$$\mathbf{I} \Rightarrow \mathbf{F} \rightarrow \omega \tag{1}$$

The → corresponds to actual classification. A simple classifier, usually a Logistic/SoftMax Regression or a Linear SVM is used. The burden falls on the feature extraction phase, denoted by ⇒, to produce descriptors of such a nature that the simple classifiers can properly discriminate the classes.

For our purposes, a feature is a small filter of some sort. More precisely, the full feature set consists of w normalized square images of size $d = p \times p$ with $p < \min(m, n)$. This set is denoted by $\mathbf{C} = \left[\mathbf{C}^1 \mid \mathbf{C}^2 \mid \ldots \mid \mathbf{C}^w\right] \in \mathbb{R}^{d \times w}$. Normalization is usually imposed by the feature learning method, but is, in general, a nice property to have, because it makes coding methods more "interpretable" and less susceptible to favoring one feature over another because of scale differences. A feature set can be obtained from several sources. Firstly, it can be generated randomly [17]. Secondly, a well-known set can be employed, such as DCT bases or Gabor Wavelet bases [9]. Thirdly, the set can be learned from a sample of patches extracted from the training set of the classification system [2–4]. **SCNG** is an example of an algorithm used for this kind of learning.

Actual extraction consists of three steps, called *coding*, *nonlinear*, and *reduce*. The coding step accepts as input the original image $\mathbf{I}$ and produces a set of w images of the same size as $\mathbf{I}$. In order to obtain the $(i, j)^{th}$ pixel of the l^{th} image, the patch of size $p \times p$ from $\mathbf{I}$, centered at position (i, j), is "coded", with regards to filter $\mathbf{C}^l$. The simplest form of coding is an inner product between the two images. Considering the whole image, this corresponds to doing a convolution with the filter $\mathbf{C}^l$. In fact, this is the strategy employed by CNNs. In general, the response for the $(i, j)^{th}$ pixel need not depend just on $\mathbf{C}^l$, but on the whole $\mathbf{C}$. Therefore, in the next section, which is dedicated to coding methods, we will consider the problem of finding the values of the pixels at position (i, j) in all w images at once.

The nonlinear step accepts as input the set of w images produced by the coding step and produces another set of w images, again of the same size as $\mathbf{I}$, but with elements mapped to a restricted interval, such as $[-1, +1]$. Each pixel in each image is transformed independently by passing its value through a sigmoid-like nonlinearity, such as the logistic function. The first and second steps, together, can be viewed as a feed-forward neural network, with mn input units and wmn output units, a very specific weight setup and a complex feed-forward rule.

Lastly, the reduce step accepts as input the previous set of w images and produces a final set of w images, but this time of smaller sizes than the original. More precisely, each image is divided into non-overlapping blocks of size $q \times q$ and all the values from each block are combined to form one value, according to some function. Common choices are the $\max(|\star|, \ldots, |\star|)$ and $\sum_{i,j} (\star)^2$ functions. The output of this stage is a set of w images of size $(m/q) \times (n/q)$. In general, CNNs can have a more flexible reduce step, but we've found this limited form, which considers each image individually, to be useful as well. The reasons given for the reduce step are that it introduces a certain kind of resistance to small translations. Basically, anywhere in a $q \times q$ block a feature is detected, the corresponding output of the reduce step should be large. Invariance to larger translations, scaling, rotation etc. is something that has to be captured by the features or by the classifier. Otherwise, enough data which cover these translations must be provided for learning.

The final feature vector $\mathbf{F}$ is a linearized version of these images, that is, a $\left[(mnw)/q^2\right]$-dimensional vector. Also notice that the whole system can be viewed as one heterogeneous neural network consisting of two different stages. In general, several of these modules can be linked, each with its own set of features, tailored to the type of images it receives from the previous layer. In principle, very deep feature extraction networks can thus be built, depending on the complexity of the dataset being studied. In our experiments, only feature extractors with one layer were used. Again, if a perceptron or a two layer MLP are used as the classifier and the inner product is used as the coding method, the whole system becomes a single large neural network. Classical back-propagation can then be used at the end to *fine-tune* the weights, starting from initial values assigned according to $\mathbf{C}$. In our experiments, this procedure was not used, but situations such as transfer-learning or self-taught learning [18] can make use of this property.

3 Coding

This section describes in more detail how to do the coding. For simplicity, we will assume we work with d-dimensional signals. Thus, the $p \times p$ patches previously discussed must be linearized such that $d = p^2$. Assume also that we are given a set of features $\mathbf{C}$, like in the previous section. Most of the times we will have $w > d$, that is, the set of features is *overcomplete*. Our goal is to approximate a signal $\mathbf{x} \in \mathbb{R}^d$ in terms of $\mathbf{C}$. The most common approach is to use a linear combination of the features. The *code* is then the signal $\mathbf{a} \in \mathbb{R}^w$ and the *approximation* is

$$\hat{\mathbf{x}} = \sum_{i=1}^{w} a_i \mathbf{C}^i = \mathbf{Ca} \ . \tag{2}$$

The quality of the code is determined by how well the reconstruction $\hat{\mathbf{x}}$ matches the original signal. For audio and image processing, it has been shown that a sparse code for $\mathbf{x}$, that is, one with numerous zero or close to zero components, has many desirable properties [2, 3]. Many methods for sparse coding have been proposed [19, 3]. We will focus on a group of iterative methods for computing $\mathbf{a}$, known as pursuits, which originate in the signal processing community. All assume $\mathbf{C}$ and $\mathbf{x}$ are given and run for a number of $k \leq d$ iterations. The general problem they try to solve is $\arg\min_a \|\mathbf{x} - \mathbf{C}\mathbf{a}\|_2^2$ subject to $\|\mathbf{a}\|_0 \leq k$. This an NP-complete problem. The pursuits are greedy approximations to it. Let the initial residual $\mathcal{R}^0\mathbf{x} = \mathbf{x}$. At iteration t, let $\mathbf{C}^\omega$ be the most similar feature in $\mathbf{C}$, relative to $\mathcal{R}^t\mathbf{x}$. The updated code and residual, $\mathbf{a}^{t+1}$ and $\mathcal{R}^{t+1}\mathbf{x}$, are produced by decomposing $\mathcal{R}^t\mathbf{x}$ in terms of $\mathbf{C}^\omega$. After k iterations, $\mathbf{a}^k$ is returned as the code associated to $\mathbf{x}$, and $\mathcal{R}^k\mathbf{x}$ is returned as a measure of the ability of the algorithm to reconstruct the signal in terms of $\mathbf{C}$. The difference between the several methods consists in how they find $\mathbf{C}^\omega$ and how they update $\mathbf{a}^{t+1}$. The general procedure is illustrated in **Algorithm 1**. At the end of this algorithm we obtain $\mathbf{x} = \mathbf{C}\mathbf{a}^k + \mathcal{R}^k\mathbf{x}$. Also, the norm of the final residual $\mathcal{R}^k\mathbf{x}$ tends to 0 as $k \to +\infty$ for sensible choices of **sim** and **next** functions. In the limit, the equality becomes $\mathbf{x} = \mathbf{C}\mathbf{a}^{+\infty}$.

Algorithm 1. The General Pursuit Method

```
input $\mathbf{C}, \mathbf{x}, k$
output $\mathbf{a}^k, \mathcal{R}^k\mathbf{x}$
  $\Lambda^0 \leftarrow \phi$
  $\mathbf{a}^0 \leftarrow \mathbf{0}$
  $\mathcal{R}^0\mathbf{x} \leftarrow \mathbf{x}$
  $t \leftarrow 0$
  while $t < k$ or $\|\mathcal{R}^t\mathbf{x}\|_2 \geq \delta$ do
    $\omega \leftarrow \arg\max_{i \in \mathbf{dom}} \mathbf{sim}(\mathcal{R}^t\mathbf{x}, \mathbf{C}, \Lambda^t, i)$
    $\Lambda^{t+1} \leftarrow \Lambda^t \cup \omega$
    $\mathbf{a}^{t+1} \leftarrow \mathbf{next}(\mathbf{a}^t, \mathcal{R}^t\mathbf{x}, \mathbf{C}, \Lambda^{t+1}, \omega)$
    $\mathcal{R}^{t+1}\mathbf{x} \leftarrow \mathcal{R}^t\mathbf{x} - \mathbf{a}_\omega^{t+1} C^\omega$
    $t \leftarrow t + 1$
  end while
```

The simplest pursuit method, introduced in [20], is Matching Pursuit (**MP**). **Table 1** shows what form the **sim** and **next** functions take in this case. An important property of this algorithm is that for every t, $\|\mathcal{R}^t\mathbf{x}\|_2^2 \geq \|\mathcal{R}^{t+1}\mathbf{x}\|_2^2$ and, furthermore, with a decay that is exponential. The two major drawbacks of this method are that the approximation at time t, $\mathbf{C}\mathbf{a}^t$, is not optimal with respect to the selection of features Λ^t; and that for the residual norm to actually reach small enough values, a $k > w$ could be necessary. However, these drawbacks are not critical for classification purposes, and, because of its simplicity and speed, we use it in our experiments.

An improvement to **MP** is Orthogonal Matching Pursuit (**OMP**) [21–23], which addresses the two issues discussed above. Again, **Table 1** shows the forms the **sim** and **next** functions take. Also, notice that at each iteration, only the features not considered before are processed. All the properties of **MP** hold here as well. At iteration t, the approximation computed is the closest point in $\overline{\text{span}(\mathbf{C}^{\Lambda^t})}$ to $\mathbf{x}$, according to the Euclidean norm. The version presented here

Table 1. The different parametrization for pursuit methods

Method	**sim** function	**next** function	**dom** domain
MP	$\lvert\langle \mathcal{R}^t \mathbf{x}, \mathbf{C}^i \rangle\rvert$	$\mathbf{a}^t + \langle \mathcal{R}^t \mathbf{x}, \mathbf{C}^\omega \rangle \delta_\omega$	$\overline{1{:}w}$
OMP	$\lvert\langle \mathcal{R}^t \mathbf{x}, \mathbf{C}^i \rangle\rvert$	$\arg\min_a \lVert \mathbf{x} - \mathbf{C}^{\Lambda^{t+1}} \mathbf{a} \rVert_2^2$	$\overline{1{:}w} \setminus \Lambda^t$

is suboptimal from an algorithmic point of view. More sophisticated methods based on QR decomposition have been developed [21, 23].

4 Learning a Feature Set

We now turn to the problem of learning the feature set $\mathbf{C}$, given a coding method $\hat{\mathcal{C}}_{\mathbf{C}}$ and a sample $\mathbf{X} = \left[\mathbf{X}^1 \mid \mathbf{X}^2 \mid \ldots \mid \mathbf{X}^N\right] \in \mathbb{R}^{d \times N}$ of linearized image patches of size $p \times p$, usually extracted from either the whole training set or from a larger "natural scenes" dataset [18]. As we previously mentioned, the method we employed here is the Sparse Coding Neural Gas approach, which is an adaptation of the Neural Gas algorithm introduced in the context of vector quantization. Vector quantization can be considered as a stricter version of feature learning, where the codes are 1-sparse and only a boolean "indicator" of the feature most similar to the input $\mathbf{x}$, as measured by the Euclidean distance, is stored.

The Neural Gas algorithm is an iterative one. It begins by initializing $\mathbf{C}$ to w random observations from the training sample $\mathbf{X}$. Then, for a number of T_{max} iterations, an adaptation process takes place, which slowly changes $\mathbf{C}$ in order to best represent the distribution over the input space. More precisely, at each iteration t an observation is randomly selected from $\mathbf{X}$ and distances to each element of $\mathbf{C}$ are computed. Each feature is then modified in a manner proportional to the distortions between it and the signal $\mathbf{x}$, on the one hand, and the ranking of this distortion in the list of all distortions, on the other hand. Therefore, the update process includes a local and a global component.

Algorithm 2. Neural Gas

```
input $\mathbf{X}, w, T_{max}, \mu^0, \mu^{T_{max}}, \lambda^0, \lambda^{T_{max}}$
output $\mathbf{C}$
  $\mathbf{C} \leftarrow$ randomly select $w$ observations from $\mathbf{X}$
  for $t = \overline{1{:}T_{max}}$ do
    $\mu^t \leftarrow \mu^0 (\mu^{T_{max}} / \mu^0)^{t/T_{max}}$          ▷ Current learning rate
    $\lambda^t \leftarrow \lambda^0 (\lambda^{T_{max}} / \lambda^0)^{t/T_{max}}$      ▷ Current neighborhood control
    $\mathbf{x} \leftarrow$ an observation from $\mathbf{X}$
    $\mathbf{a} \leftarrow [\ \lVert \mathbf{x} - \mathbf{C}^i \rVert_2^2$ for $i \in \overline{1{:}w}\ ]$
    $\mathbf{C} \leftarrow \mathbf{C} + [\ \mu^t e^{-rank_{\mathbf{a}}(a_i)/\lambda^t} (\mathbf{x} - \mathbf{C}^i)$ for $i \in \overline{1{:}w}\ ]$
  end for
```

Algorithm 3. Sparse Coding Neural Gas V1

```
input $\mathbf{X}, w, \mathcal{C}, T_{max}, \lambda^0, \lambda^{T_{max}}, \mu^0, \mu^{T_{max}}$
output $\mathbf{C}$
  $\mathbf{C} \leftarrow$ randomly initialize $w$ normalized features
  for $t = \overline{1{:}T_{max}}$ do
    $\mu^t \leftarrow \mu^0(\mu^{T_{max}}/\mu^0)^{t/T_{max}}$          ▷ Current learning rate
    $\lambda^t \leftarrow \lambda^0(\lambda^{T_{max}}/\lambda^0)^{t/T_{max}}$     ▷ Current neighborhood control
    $\mathbf{x} \leftarrow$ an observation from $\mathbf{X}$
    $\mathbf{a} \leftarrow \mathcal{C}_{\mathbf{C}}\{\mathbf{x}\}$
    $\mathbf{C} \leftarrow \mathbf{C} + [\ \mu^t e^{-rank_{\mathbf{a}}(a_i)/\lambda^t} a_i(\mathbf{x} - a_i\mathbf{C}^i)$ for $i \in \overline{1{:}w}$ ]
    $\mathbf{C} \leftarrow$ normalize each feature in $\mathbf{C}$
  end for
```

Algorithm 4. Sparse Coding Neural Gas V2

```
input $\mathbf{X}, w, \mathcal{C}, T_{max}, \lambda^0, \lambda^{T_{max}}, \mu^0, \mu^{T_{max}}$
output $\mathbf{C}$
  $C \leftarrow$ randomly initialize $w$ normalized features
  for $t = \overline{1{:}T_{max}}$ do
    $\mu^t \leftarrow \mu^0(\mu^{T_{max}}/\mu^0)^{t/T_{max}}$          ▷ Current learning rate
    $\lambda^t \leftarrow \lambda^0(\lambda^{T_{max}}/\lambda^0)^{t/T_{max}}$     ▷ Current neighborhood control
    $\mathbf{x} \leftarrow$ an observation from $\mathbf{X}$
    $S^0 \leftarrow$ initialize coding method specific state
    for $i = \overline{0{:}k}$ do
      $[\alpha^i\ \Lambda^i\ S^{i+1}] \leftarrow \mathcal{C}_{\mathbf{C}}\{S^i, \mathbf{x}\}$    ▷ $\alpha^i$ stores similarities for features in $\overline{1{:}w} \setminus \Lambda^i$
      $\mathbf{C} \leftarrow \mathbf{C} + [\ \mu^t e^{-rank_{\alpha^i}(\alpha^i_j)/\lambda^t} \alpha^i_j(\mathbf{x} - \alpha^i_j\mathbf{C}^j)$ for $j \in \overline{1{:}w} \setminus \Lambda^i$ ]
      $\mathbf{C} \leftarrow$ normalize each feature in $\mathbf{C}$
    end for
  end for
```

Algorithm 2 gives the whole picture. Note that both a time decreasing learning factor is used as well as a time decreasing neighborhood control. The algorithm is similar to the well-known Self-Organizing Map. The difference lies in changing the weight update from one which considers a pre-defined topology, to one which looks at the neighborhood withing the input space. More complete versions of this algorithm [15] can actually build a topological description of the input space which is useful for exploratory data analysis.

The Neural Gas algorithm works in the input space rather than feature set space. Adapting the algorithm to work with features and accept any coding method gives rise to a first version of the Sparse Coding Neural Gas. The major modification is the fact that each update is done according to Oja's Rule [24] instead of the simple error term of the Neural Gas. The full algorithm is described in **Algorithm 3**. Notice that the $rank_{\mathbf{a}}$ function considers absolute values, so that features are updated proportional to the magnitude of the associated response.

A further improvement is possible considering the fact that many coding methods are iterative and produce orderings of a subset $\overline{1{:}w} \setminus \Lambda^t$ of the feature elements at each iteration. **MP** and **OMP** are such methods. A second version of the Sparse Coding Neural Gas is presented as **Algorithm** 4. Notice that at each iteration only the subset of previously unselected features is updated, instead of the whole set. Also, the variable S^i, which is a substitute for all the abstracted coding method specific information, must contain a copy of the original feature set $\mathbf{C}$ at iteration t, before the inner-loop coding procedure. The reason for this is that $\mathbf{C}$ is updated in the inner-loop and it can cause problems for the coder to change the features as time progresses.

5 Experiments

In order to test the classifier system, two datasets were employed: the well known MNIST handwritten digit dataset [6] and the NORB object dataset [25]. Both of these are widely used for benchmarking classifiers. MNIST has 10 classes corresponding to the 10 Arabic numerals. It consists of 60000 training images and 10000 test images. NORB has 5 classes, corresponding to different categories of objects (animal, human, plane, truck, car). It consists of 24300 training images and 24300 test images. An example of the kind of features learned from these sets can be seen in **Figure** 1. For the NORB dataset, a pre-processing step of "whitening" is applied. This speeds up convergence and is achieved through the ZCA transform as described in [4].

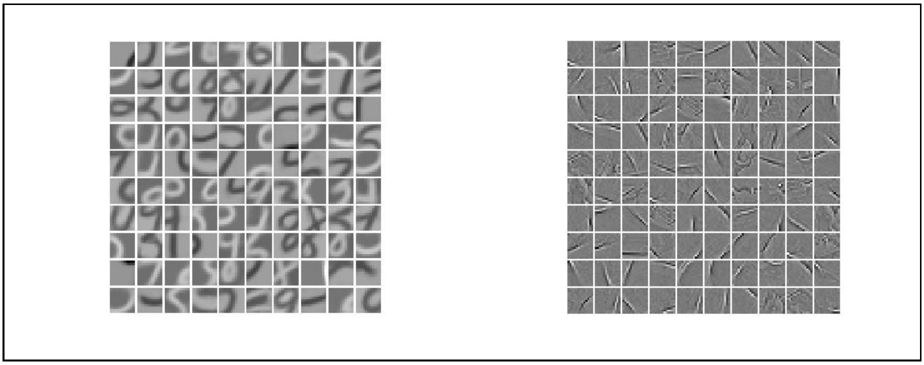


Fig. 1. SCNG learned features for the MNIST (left) and NORB (right) datasets. For NORB, the ZCA pre-processing step was applied.

Classification scores for the methods we used as well as other details are presented in **Table** 2 and **Table** 3. Using random patches as features and using features learned by gradient descent [3] in the feature set space are also presented. The best result in the literature is also included. For both datasets, only methods which dealt with the unmodified dataset are considered. Extending the dataset through elastic distortions [8] has been shown to improve performance. However, such methods are not always applicable. Notice that both **SCNG** and Gradient Descent in feature set space produce better results than simply using random features, but are otherwise close in performance.

Table 2. Classification results for MNIST

Method	**Error**	**Notes**
Baseline	5.34	Linear SVM on raw pixel data
CNN (unsupervised pretraining)	0.53	Best without dataset extension. See [1].
Our Method (SCNG)	0.71	MP-11 with $p = 11$ and $q = 4$ and $w = 1024$
Our Method (Gradient)	0.77	MP-11 with $p = 11$ and $q = 4$ and $w = 1024$
Our Method (Random)	0.87	MP-11 with $p = 11$ and $q = 4$ and $w = 1024$
Our Method (SCNG)	0.69	MP-25 with $p = 11$ and $q = 4$ and $w = 1024$
Our Method (Gradient)	0.67	MP-25 with $p = 11$ and $q = 4$ and $w = 1024$
Our Method (Random)	0.77	MP-25 with $p = 11$ and $q = 4$ and $w = 1024$

Table 3. Classification results for NORB

Method	**Score**	**Notes**
Baseline	25.13	Linear SVM with standardization
CNN (back-propagation)	7.86	Best without dataset extension. See [27].
Our Method (SCNG)	11.59	MP-11 with $p = 17$ and $q = 3$ and $w = 1024$
Our Method (Gradient)	11.51	MP-11 with $p = 17$ and $q = 3$ and $w = 1024$
Our Method (Random)	12.46	MP-11 with $p = 17$ and $q = 3$ and $w = 1024$
Our Method (SCNG)	10.87	MP-25 with $p = 17$ and $q = 3$ and $w = 1024$
Our Method (Gradient)	11.01	MP-25 with $p = 17$ and $q = 3$ and $w = 1024$
Our Method (Random)	11.81	MP-25 with $p = 17$ and $q = 3$ and $w = 1024$

After features were extracted from a dataset, a classifier was trained on the features training dataset. We employed a simple Linear SVM, as implemented by LIBLINEAR [26]. The regularization parameter C was fine-tuned through cross-validation. A subset of 20% of the training set instances was put aside for this purpose. We tested C with 20 possible values, logarithmically distributed between 10^{-3} and 10^{-1}. For each value, 5 random 50/50 splits of the dataset were performed. The classifier was trained on one half of the data and was tested on the other half. Average scores were then computed for model selection purposes. After a good C was found, a classifier was built on the whole training dataset and evaluated on the provided testing dataset. We also tried Logistic Regression and Gaussian Kernel SVMs. For the former, the scores were slightly lower, while for the latter, the scores were similar. We thus preferred the Linear SVM for performance and computational reasons.

6 Conclusion

In this paper we have shown a way to build a complex image classifier with the Sparse Coding Neural Gas. The classifier consists of several components (feature learning, coding system, proper classifier), and we investigated whether the Sparse Coding Neural Gas algorithm is applicable as a feature learning method. The resulting classifier was tested on the MNIST and NORB datasets and found to perform close to state-of-the-art. Better methods usually employ two or more feature extraction layers, more sophisticated classifiers, or groups of classifiers which vote on the final class. Our simple setup is promising to reach or perhaps even surpass state-of-the-art results with some further extensions along these lines.

Acknowledgments. Horia Coman would like to thank, for supporting this work, The German Academic Exchange Service Programme “Ostpartnerschaften”, the University of Lübeck, and The Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labor, Family and Social Protection through the Financial Agreement POSDRU/86/1.2/ S/61756.

References

1. Jarrett, K., Kavukcuoglu, K., Ranzato, M., Keen, Y.: What is the Best Multi-Stage Architecture for Object Recognition? In: Proc. International Conference on Computer Vision, ICCV 2009 (2009)
2. Olshausen, B., Field, D.: Emergence of Simple-Cell Receptive Field Properties by Learning a Sparse Code for Natural Images. Nature 381, 607–609 (1996)
3. Olshausen, B., Field, D.: Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1? Vision Research 37, 3311–3325 (1998)
4. Krizhevsky, A.: Learning Multiple Layers of Features from Tiny Images (2009)
5. Kavukcuoglu, K., Sermanet, P., Boureau, Y., Gregor, K., Mathieu, M., LeCun, Y.: Learning Convolutional Feature Hierarchies for Visual Recognition. In: Advances in Neural Information Processing Systems (NIPS 2010), vol. 23 (2010)
6. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-Based Learning Applied to Document Recognition. In: Intelligent Signal Processing, pp. 306–351. IEEE Press (2001)
7. LeCun, Y., Kavukcuoglu, K., Farabet, C.: Convolutional Networks and Applications in Vision. In: Proc. International Symposium on Circuits and Signals (ISCAS 2010). IEEE Press (2010)
8. Simard, P.Y., Steinkraus, D., Platt, J.C.: Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. In: Int’l Conference on Document Analysis and Recognition, pp. 958–963 (2003)
9. Labusch, K., Barth, E., Martinetz, T.: Simple Method for High-Performance Digit Recognition Based on Sparse Coding. IEEE Transactions on Neural Networks 19(11), 1985–1989 (2008)
10. Henaff, M., Jarret, K., Kavukcuoglu, K., LeCun, Y.: Unsupervised Learning for Scalable Audio Classification. In: Proceedings of International Symposium on Music Information Retrieval, ISMIR 2011 (2011)

11. Labusch, K., Barth, E., Martinetz, T.: Sparse Coding Neural Gas: Learning of Overcomplete Data Representations. Neurocomputing 72(7-9), 1547–1555 (2009)
12. Labusch, K., Barth, E., Martinetz, T.: Learning Data Representations with Sparse Coding Neural Gas. In: Proceedings of the 16th European Symposium on Artificial Neural Networks, pp. 233–238 (2008)
13. Labusch, K., Barth, E., Martinetz, T.: Demixing Jazz-Music: Sparse Coding Neural Gas for the Separation of Noisy Overcomplete Sources. Neural Network World 19(5), 561–579 (2009)
14. Labusch, K., Barth, E., Martinetz, T.: Sparse Coding Neural Gas for the Separation of Noisy Overcomplete Sources. In: Kůrková, V., Neruda, R., Koutník, J. (eds.) ICANN 2008, Part I. LNCS, vol. 5163, pp. 788–797. Springer, Heidelberg (2008)
15. Martinetz, T., Schulten, K.: A "Neural-Gas" Network Learns Toplogies. Artificial Neural Networks 1, 397–402 (1991)
16. Martinetz, T., Berkovich, S., Schulten, K.: "Neural-Gas" Network for Vector Quantization and its Application to Time-Series Prediction. IEEE Transactions on Neural Networks 4(4), 397–402 (1991)
17. Saxe, A., Koh, P.W., Chen, Z., Bahand, M., Suresh, B., Ng, A.: On Random Weights and Unsupervised Feature Learning. In: Proceedings of the Twenty-Eight International Conference on Machine Learning (2011)
18. Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.: Self-taught Learning: Transfer Learning from Unlabeled Data. In: Proceedings of the 24th International Conference on Machine Learning (ICML 2007), pp. 759–766 (2007)
19. Donoho, D.: For Most Large Underdetermined Systems of Linear Equations the Minimal $\mathcal{L}_1$-norm Solution is also the Sparsest Solution. Communications on Pure and Applied Mathematics 59, 797–766 (2007)
20. Mallat, Z., Zhang, Z.: Matching Pursuits With Time-Frequency Dictionaries. IEEE Transactions on Signal Processing 41, 3397–3451 (1993)
21. Davis, G., Mallat, S., Zhang, Z.: Adaptive Time-Frequency Decomposition with Matching Pursuits. SPIE Journal of Optical Engineering 33, 2183–2191 (1994)
22. Pati, Y., Rezaiifar, R., Krishnaprasad, P.: Orthogonal Matching Pursuit: Recursive Function Approximation with Applications to Wavelet Decomposition. In: Asilomar Conference on Signals, Systems and Computers (1993)
23. Blumensath, T., Davies, M.: On the Difference Between Orthogonal Matching Pursuit and Orthogonal Least Squares (2007)
24. Oja, E.: Simplified Neuron Model as a Principal Component Analyzer. Journal of Mathematical Biology 15, 267–273 (1982)
25. LeCun, Y., Huang, F.-J., Bottou, L.: Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting. In: Proceedings of CVPR 2004 (2004)
26. Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., Lin, C.-J.: LIBLINEAR: A Library for Large Linear Classification. J. Mach. Learn. Res. 9, 1871–1874 (2008)
27. Cireşan, D., Meier, U., Masci, J., Gambardella, L., Schmidhuber, J.: High-Performance Neural Networks for Visual Object Classification (2011)

Hand Tracking with an Extended Self-Organizing Map

Andreea State[1,2], Foti Coleca[1,3], Erhardt Barth[1,3], and Thomas Martinetz[1]

[1] Institute for Neuro- and Bioinformatics, University of Lübeck
Ratzeburger Allee 160, 23538 Lübeck, Germany
{state,coleca,barth,martinetz}@inb.uni-luebeck.de
http://www.inb.uni-luebeck.de

[2] University "POLITEHNICA" of Bucureşti
Splaiul Independenţei 313, 060042 Bucureşti, Romania
http://www.upb.ro

[3] Gestigon GmbH, Innovations Campus Lübeck
Maria-Goeppert Straße 1, 23562 Lübeck, Germany
http://www.gestigon.de

Abstract. We introduce an extension of the self-organizing map for performing 3D hand skeleton tracking. We use a range camera for data acquisition and apply a SOM-like learning process within each frame in order to capture the hand pose. Our method uses a topology consisting of 1D and 2D segments for an improved representation of the hand. The proposed algorithm is very efficient and produces good tracking results.

Keywords: hand skeleton tracking, self-organizing maps, kinect.

1 Introduction

The problem of object tracking and pose estimation has gained much attention during the last years, due to the variety of new technologies and devices designed for 3D image acquisition. While standard 2D cameras necessitate more complex image processing techniques, 3D cameras provide a more favorable framework for tracking algorithms, as depth information enables the reconstruction of 3D objects. In particular, hand tracking can be used in a wide variety of applications, e.g. gesture recognition, and represents a milestone in human-machine interaction. The difficulty lies in the fact that the state space is extremely large, due to the 27 degrees of freedom of the human hand [1].

Our work focuses on developing a hand tracking algorithm for 3D cameras. Having acquired 3D image information, we aim at building a tracking algorithm that is both accurate and of low computational cost. This is achieved with an extension of the approach introduced in [10]. After a simple preprocessing step which assumes that the hand is the closest object to the camera, we obtain a point cloud in 3D of the hand which we then represent with Kohonen's self-organizing map [3]. For this purpose the topology of the Kohonen network is crafted according to the skeleton of a hand, as illustrated in Fig. 1. However, to

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 115–124.
DOI: 10.1007/978-3-642-35230-0_12

be able to obtain good tracking results, we have to extend the SOM algorithm such that the point cloud of the hand is represented not only by the nodes of the network, but also by the line and plane segments between the nodes.

Compared to our approach based on a SOM, the authors of [4] use kinematic models and build a hand state model which consists in a set of lines and points generated by the projection of the hand model into the image plane. Hand pose estimation based on features derived from projections of the hand and its shadow is presented in [5]. Nevertheless, the method requires controlled background and lighting, and is susceptible to occlusion. The authors of [6] introduce a machine learning architecture for matching image features to 3D hand example poses, through solving an optimization problem based on Bayes' rule. Another approach is to estimate hand pose with a database of synthetic hand images. For instance, in [7] an indexed image database is used to retrieve the closest hand match, with an adapted chamfer distance and line matching algorithm. In [8], the authors implement a cascade of increasingly complex classifiers to determine hand pose from synthetic training data. In order to handle occlusion, particle filters can be used. In [9], the authors apply a meta-descent algorithm to minimize the distance between a predicted position and the observed position, while particle filters predict new sample positions and help the optimization algorithm to recover from local minima. As shown in [2], the combined usage of intensity images and range information provides a good framework for body tracking.

Section 2 will consist in a detailed explanation of our method, followed by evaluation and results in Section 3. Finally, we present a few conclusions in Section 4.

2 The Extended SOM

We use a network topology that can suitably describe a human hand. Our network is made up of sixteen nodes, and the defined connections are illustrated in Fig. 1.

The standard SOM algorithm starts with the initialization of the network weights, followed by the iteration of two procedures, the competition and the update of the weights. At each iteration, a sample point from the dataset is

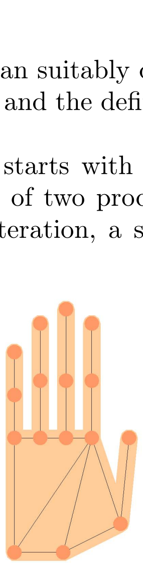


Fig. 1. The extended SOM left-hand topology

randomly chosen. During the competition phase, the algorithm determines a winner node, that is, the node characterized by the weight with the minimum Euclidean distance from the sample point.

Given a network with n neurons and a sample point $\mathbf{x}$, we determine the winner node $\hat{i}$ as follows:

$$\hat{i} = arg\{\underset{i}{min}||\mathbf{x} - \mathbf{w}_i||_2\},\ i = 1, \ldots, n \tag{1}$$

with $\mathbf{w}_i$ being the weight of node i. Next, the update phase aims at decreasing the distance between the winner-node weight and the sample point, by an amount given by the learning rate $\epsilon(t)$. First, let us define the learning rate function as

$$\epsilon(t) = \epsilon_i \left(\frac{\epsilon_f}{\epsilon_i}\right)^{\frac{t}{t_{max}}}, \tag{2}$$

where ϵ_i is the initial learning rate, ϵ_f is the final learning rate, t is the current iteration and t_{max} is the maximum number of iterations performed on the network. Then, the weight $\mathbf{w}_{\hat{i}}$ is updated at step t according to:

$$\mathbf{w}_{\hat{i}}(t+1) = \mathbf{w}_{\hat{i}}(t) + \epsilon(t)(\mathbf{x} - \mathbf{w}_{\hat{i}}(t))\ . \tag{3}$$

The standard SOM algorithm also uses a neighbourhood update, in the sense that not only the winner-node weight is updated, but also the weights of the neighbour-nodes.

Our proposed algorithm extends the competition and the update step to 1D and 2D network segments. The 1D-segments are the lines between pairs of connected nodes, and the 2D-segments are the triangles determined by triples of connected nodes. 1D-segments allow to represent the fingers more accurately, and the 2D-segments form the palm of the hand. By segment updates we aim at minimizing the average distance between network segments and points from the dataset (point cloud of the hand provided by the 3D-camera). Now we not only have elements of dimension zero (nodes) like in the classical case, but also elements of dimension one and two for representing the data distribution. This approach is motivated by the fact that a hand-like topology involves a difficult separation between the nodes corresponding to fingers. A node that belongs to one finger can easily be attracted by another finger, given the topological closeness. This may lead to an erroneous tracking of the hand and destroy the topological relations. With these 1D and 2D segments we can represent fingers and parts of the palm more accurately and expect the self-organizing maps to be less prone to this type of errors.

The competition phase in our extended SOM algorithm determines whether a single node or a 1D-segment or a 2D-segment is closest to the randomly chosen sample point $\mathbf{x}$. Depending on this result, the update phase will either perform a classical node update as shown in Equation 3 or a segment update.

The distance to a 1D-segment $[\mathbf{w}_i; \mathbf{w}_j]$ (see Fig. 2a) is determined via the projection point $\mathbf{p}$ of $\mathbf{x}$ onto the given segment. Let us define $\mathbf{d}_{ji} = \mathbf{w}_j - \mathbf{w}_i$ and $\mathbf{d}_{ij} = \mathbf{w}_i - \mathbf{w}_j$. Then, we may write $\mathbf{p}$ as

$$\mathbf{p} = \mathbf{w}_i + \eta_{ji}\mathbf{d}_{ji},\ 0 \leq \eta_{ji} \leq 1\ . \tag{4}$$

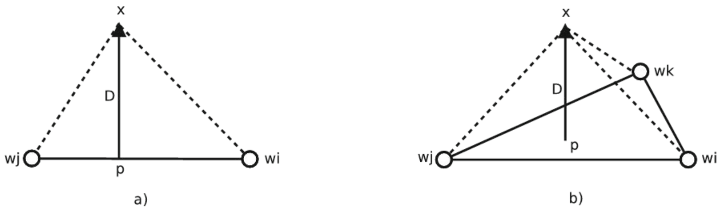


Fig. 2. a) The projection $\mathbf{p}$ of sample point $\mathbf{x}$ onto 1D-segment $[\mathbf{w}_i; \mathbf{w}_j]$. b) The projection $\mathbf{p}$ of sample point $\mathbf{x}$ onto 2D-segment $[\mathbf{w}_i; \mathbf{w}_j; \mathbf{w}_k]$.

Similarly,

$$\mathbf{p} = \mathbf{w}_j + \eta_{ij}\mathbf{d}_{ij},\ 0 \le \eta_{ij} \le 1 \tag{5}$$

and $\eta_{ji} + \eta_{ij} = 1$. Given the unit vectors $\hat{\mathbf{d}}_{ji}$ and $\hat{\mathbf{d}}_{ij}$, the coefficients η_{ji} and η_{ij} are

$$\eta_{ji} = \frac{(\mathbf{x} - \mathbf{w}_i)\hat{\mathbf{d}}_{ji}}{||\mathbf{d}_{ji}||} \tag{6}$$

$$\eta_{ij} = \frac{(\mathbf{x} - \mathbf{w}_j)\hat{\mathbf{d}}_{ij}}{||\mathbf{d}_{ij}||} . \tag{7}$$

Then the squared distance $||\mathbf{D}||^2$ of $\mathbf{x}$ to the 1D-segment $[\mathbf{w}_i; \mathbf{w}_j]$ is

$$\begin{aligned} ||\mathbf{D}||^2 &= ||\mathbf{x} - \mathbf{p}||^2 \\ &= ||\mathbf{x} - \mathbf{w}_i||^2 - ||\mathbf{p} - \mathbf{w}_i||^2 \\ &= ||\mathbf{x} - \mathbf{w}_i||^2 - \eta_{ji}^2 ||\mathbf{w}_j - \mathbf{w}_i||^2 . \end{aligned} \tag{8}$$

The 1D-segment that is closest to $\mathbf{x}$ is determined by

$$(\hat{i}, \hat{j}) = arg\{\underset{ij}{min}||\mathbf{D}_{ij}||\}, i, j = 1, \ldots, n . \tag{9}$$

Evidently, the above equation applies only to pairs of connected nodes (i, j).

Similarly, the distance to a 2D-segment (see Fig. 2b) can be determined. Let us define $\mathbf{d}_{ji} = \mathbf{w}_j - \mathbf{w}_i$, $\mathbf{d}_{ki} = \mathbf{w}_k - \mathbf{w}_i$. Then, we may write $\mathbf{p}$ as

$$\mathbf{p} = \mathbf{w}_i + \eta_{ji}\mathbf{d}_{ji} + \eta_{ki}\mathbf{d}_{ki},\ 0 \le \eta_{ji}, \eta_{ki} \le 1,\ \eta_{ji} + \eta_{ki} \le 1 . \tag{10}$$

Analogously,

$$\mathbf{p} = \mathbf{w}_j + \eta_{ij}\mathbf{d}_{ij} + \eta_{kj}\mathbf{d}_{kj},\ 0 \le \eta_{ij}, \eta_{kj} \le 1,\ \eta_{ij} + \eta_{kj} \le 1 \tag{11}$$

and

$$\mathbf{p} = \mathbf{w}_k + \eta_{ik}\mathbf{d}_{ik} + \eta_{jk}\mathbf{d}_{jk},\ 0 \le \eta_{ik}, \eta_{jk} \le 1,\ \eta_{ik} + \eta_{jk} \le 1 . \tag{12}$$

We then compute the squared distance $||\mathbf{D}||^2$ of $\mathbf{x}$ to the 2D-segment (triangle) determined by $[\mathbf{w}_i; \mathbf{w}_j; \mathbf{w}_k]$ according to

$$\begin{aligned} ||\mathbf{D}||^2 &= ||\mathbf{x} - \mathbf{p}||^2 \\ &= ||\mathbf{x} - \mathbf{w}_i - \eta_{ji}\mathbf{d}_{ji} - \eta_{ki}\mathbf{d}_{ki}||^2 \end{aligned} \tag{13}$$

where

$$\eta_{ji} = \frac{(\mathbf{x}-\mathbf{w}_i)\hat{\mathbf{d}}_{ji} - \Big((\mathbf{x}-\mathbf{w}_i)\hat{\mathbf{d}}_{ki}\Big)\,(\hat{\mathbf{d}}_{ki}\hat{\mathbf{d}}_{ji})}{||\mathbf{d}_{ji}||\,\Big(1-(\hat{\mathbf{d}}_{ki}\hat{\mathbf{d}}_{ji})^2\Big)} \tag{14}$$

$$\eta_{ki} = \frac{(\mathbf{x}-\mathbf{w}_i)\hat{\mathbf{d}}_{ki} - \Big((\mathbf{x}-\mathbf{w}_i)\hat{\mathbf{d}}_{ji}\Big)\,(\hat{\mathbf{d}}_{ki}\hat{\mathbf{d}}_{ji})}{||\mathbf{d}_{ki}||\,\Big(1-(\hat{\mathbf{d}}_{ki}\hat{\mathbf{d}}_{ji})^2\Big)}\,. \tag{15}$$

The 2D-segment that is closest to $\mathbf{x}$ is determined by

$$(\hat{i},\hat{j},\hat{k}) = arg\{\underset{ijk}{min}||\mathbf{D}_{ijk}||\}, i,j,k = 1,\ldots,n \tag{16}$$

with (i,j,k) three connected nodes.

After having determined whether one of the nodes, a 1D-segment or a 2D-segment is closest to the randomly chosen sample point, the update procedure takes place. The simplest situation is illustrated in Fig. 3a, when a single node is closest and has to be updated. This is done according to the standard SOM algorithm. In case a segment has to be updated, the nodes which determine the segment have to be moved such that the distance $||\mathbf{D}||$ is reduced. We derive the respective movement by gradient descent minimization on the squared segment distance. For node j we obtain

$$\begin{aligned}
-\frac{1}{2}\frac{\partial\mathbf{D}^2}{\partial\mathbf{w}_j} &= -\frac{1}{2}\frac{\partial}{\partial\mathbf{w}_j}\left((\mathbf{x}-\mathbf{w}_i)^2 - \eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i)^2\right)\\
&= \frac{1}{2}\frac{\partial}{\partial\mathbf{w}_j}\left(\eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i)^2\right)\\
&= \frac{1}{2}\left(\frac{\partial}{\partial\mathbf{w}_j}\eta_{ji}^2\right)(\mathbf{w}_j-\mathbf{w}_i)^2 + \frac{1}{2}\eta_{ji}^2\left(\frac{\partial}{\partial\mathbf{w}_j}(\mathbf{w}_j-\mathbf{w}_i)^2\right)\\
&= \eta_{ji}\left(\frac{\partial}{\partial\mathbf{w}_j}\eta_{ji}\right)(\mathbf{w}_j-\mathbf{w}_i)^2 + \eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i)\\
&= \eta_{ji}\left[(\mathbf{x}-\mathbf{w}_i) - 2\frac{(\mathbf{x}-\mathbf{w}_i)(\mathbf{w}_j-\mathbf{w}_i)}{(\mathbf{w}_j-\mathbf{w}_i)^2}(\mathbf{w}_j-\mathbf{w}_i)\right] + \eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i)\\
&= \eta_{ji}(\mathbf{x}-\mathbf{w}_i) - 2\eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i) + \eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i)\\
&= \eta_{ji}(\mathbf{x}-\mathbf{w}_i) - \eta_{ji}^2(\mathbf{w}_j-\mathbf{w}_i)\,.
\end{aligned} \tag{17}$$

Given the above result and with the symmetry in i and j, the two displacements applied to the winner 1D-segment nodes are

$$\mathit{\Delta}\mathbf{w}_{\hat{j}} = \epsilon(t)\eta_{\hat{j}\hat{i}}\left(\mathbf{x}-\mathbf{w}_{\hat{i}} - \eta_{\hat{j}\hat{i}}(\mathbf{w}_{\hat{j}}-\mathbf{w}_{\hat{i}})\right) \tag{18}$$

$$\mathit{\Delta}\mathbf{w}_{\hat{i}} = \epsilon(t)\eta_{\hat{i}\hat{j}}\left(\mathbf{x}-\mathbf{w}_{\hat{j}} - \eta_{\hat{i}\hat{j}}(\mathbf{w}_{\hat{i}}-\mathbf{w}_{\hat{j}})\right)\,. \tag{19}$$

The movement of the two nodes is orthogonal to the line segment and illustrated in Fig. 3b.

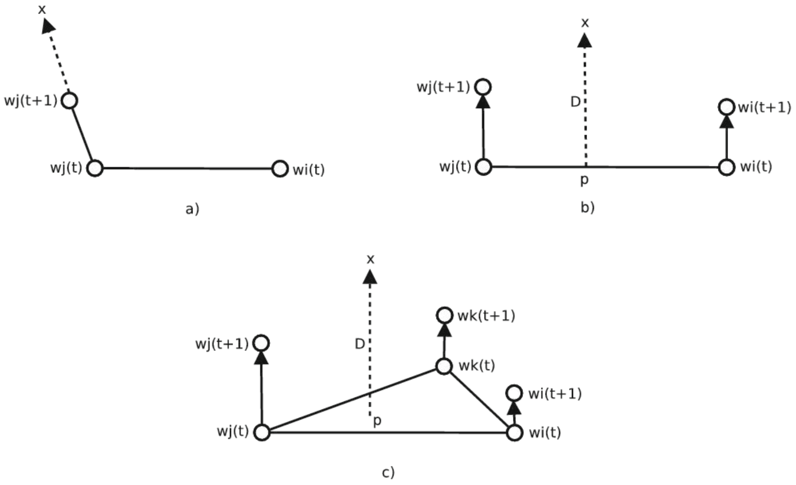


Fig. 3. a) Weight $\mathbf{w}_j$ is displaced towards the data point $\mathbf{x}$, on the direction given by vector $\mathbf{x} - \mathbf{w}_j$. b) Weights $\mathbf{w}_i$ and $\mathbf{w}_j$ are displaced towards the data point $\mathbf{x}$ on directions parallel with the vector formed by $\mathbf{x}$ and its projection $\mathbf{p}$. c) Weights $\mathbf{w}_i$, $\mathbf{w}_j$ and $\mathbf{w}_k$ are displaced towards the data point $\mathbf{x}$ on directions parallel with the vector formed by $\mathbf{x}$ and its projection $\mathbf{p}$. The closer $\mathbf{p}$ is to a node, the larger its update.

In case a 2D-segment is closest, like in Fig. 3c, three nodes have to be updated. With gradient descent similar to above we obtain the three displacements

$$\Delta\mathbf{w}_{\hat{\imath}} = (1 - \eta_{\hat{\jmath}\hat{\imath}} - \eta_{\hat{k}\hat{\imath}})(\mathbf{x} - \mathbf{w}_{\hat{\imath}} - \eta_{\hat{\jmath}\hat{\imath}}\mathbf{d}_{\hat{\jmath}\hat{\imath}} - \eta_{\hat{k}\hat{\imath}}\mathbf{d}_{\hat{k}\hat{\imath}}) \tag{20}$$

$$\Delta\mathbf{w}_{\hat{\jmath}} = (1 - \eta_{\hat{\imath}\hat{\jmath}} - \eta_{\hat{k}\hat{\jmath}})(\mathbf{x} - \mathbf{w}_{\hat{\imath}} - \eta_{\hat{\jmath}\hat{\imath}}\mathbf{d}_{\hat{\jmath}\hat{\imath}} - \eta_{\hat{k}\hat{\imath}}\mathbf{d}_{\hat{k}\hat{\imath}}) \tag{21}$$

$$\Delta\mathbf{w}_{\hat{k}} = (1 - \eta_{\hat{\imath}\hat{k}} - \eta_{\hat{\jmath}\hat{k}})(\mathbf{x} - \mathbf{w}_{\hat{\imath}} - \eta_{\hat{\jmath}\hat{\imath}}\mathbf{d}_{\hat{\jmath}\hat{\imath}} - \eta_{\hat{k}\hat{\imath}}\mathbf{d}_{\hat{k}\hat{\imath}})\,, \tag{22}$$

this time orthogonal to the triangle.

At this point, a short discussion is required concerning the segment updates. Given that the displacements orthogonal to the line or triangle are of finite size, with each update the respective line or triangle will be slightly enlarged. With many update steps the network might increase over the borders of the data space. Several solutions to this problem are possible. The most canonical one is to add a "spring-like" forgetting term, which has to be weighted such that an appropriate shortening of the distances of the updated nodes takes place with each update step. Details are explained in [11].

3 Results and Evaluation

Our tracking algorithm uses a Kinect [12] device for data acquisition. The Kinect has an infrared depth sensor and a special microchip that allow obtaining depth information from the scene. The Kinect functions properly for distances in the interval 1.2 − 3.5 meters. The image stream is characterized by a 640 × 480 pixels resolution.

The points corresponding to the hand are extracted with a threshold segmentation performed on the given depth frame, based on the assumption that the hand is the closest object to the camera. This yields the point cloud our algorithm works on (see Fig. 4). Note that our extended SOM algorithm is applied to each individual frame, each time with 5000 training steps (a training step consists of a random choice of a data point, followed by a competition and update step). The following frame always uses as starting position for the network the result of the previous frame. Only in the very beginning of the tracking procedure we need to initialize the network such that we start with an open hand and the network is aligned to the fingers and the thumb. A random initialization might have problems to converge correctly since the network topology is asymmetrical.

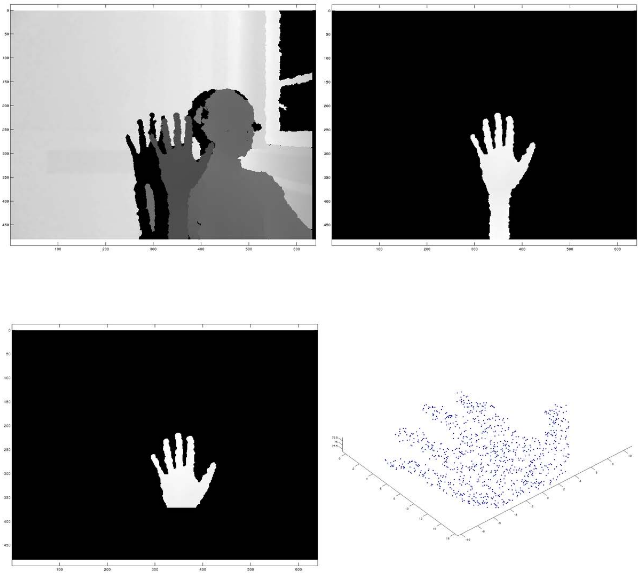


Fig. 4. Starting from a frame captured with the Kinect, we extract the hand based on the closest object assumption. We remove the points corresponding to the forearm and we obtain the 3D point cloud used for the tracker's learning stage.

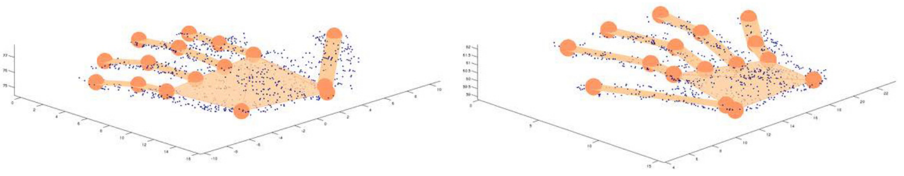


Fig. 5. The extended SOM tracker converges to the open palm topology, for both left and right hand

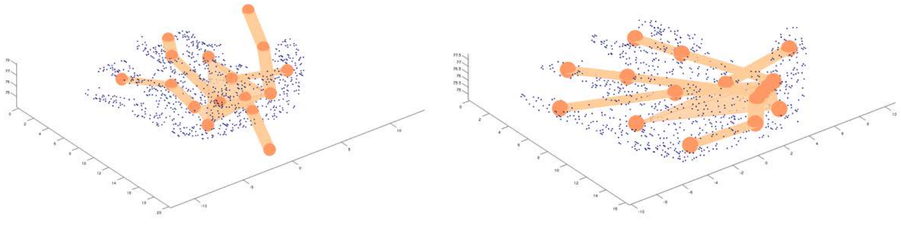


Fig. 6. A standard SOM tracker does not converge to a given hand topology

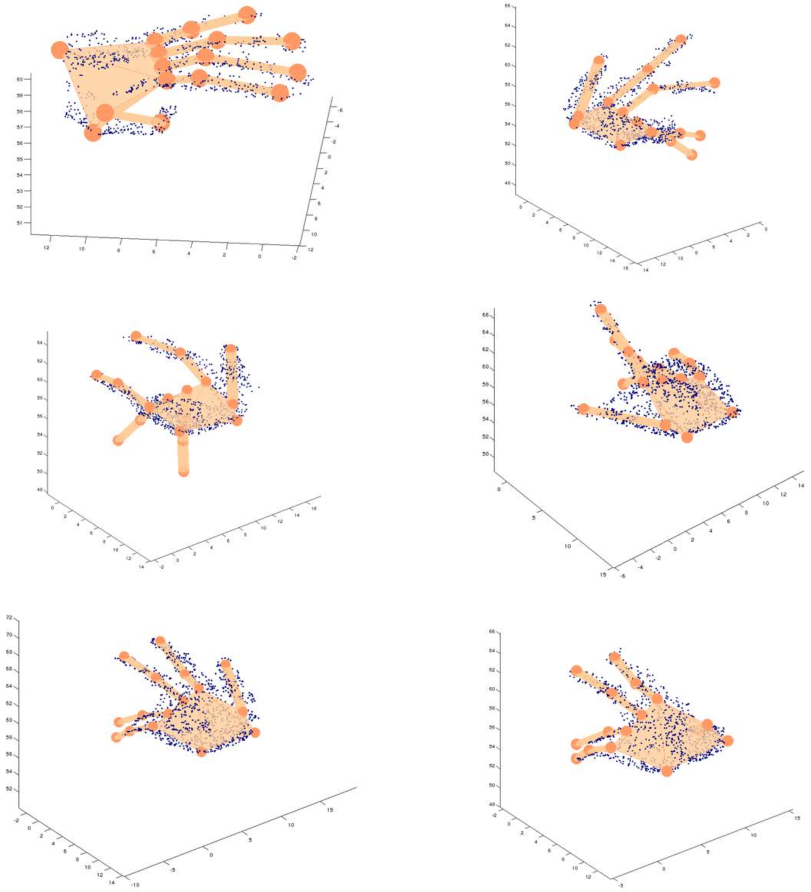


Fig. 7. Different hand postures and bent fingers. The extended SOM converges to the correct topology.

In each new frame, we perform a training during which the network receives as input the data points from the given hand point cloud. Fig. 5 shows how the hand network has converged into the point cloud of a given frame, for the left and right hand. As desired, the network represents the given hand topology by minimizing the mean squared distance between the data points and the network with its 1D and 2D segments.

On the contrary, as illustrated in Fig. 6, by performing a standard SOM learning without using the 1D and 2D segments of the network, the tracker does not manage to represent the given hand topology and converges to an entangled form of the network.

Fig. 7 shows results for different hand postures and bent fingers. In all these cases our extended SOM converged correctly and can now be used for representing corresponding gestures.

4 Conclusion

We presented an extension of the SOM which can successfully be applied to the problem of hand skeleton tracking with 3D cameras. The algorithm is efficient enough to be applied to each individual frame of a kinect camera. The extended SOM algorithm is able to produce a rough, but accurate estimate of the hand pose, at comparatively low computational cost.

Our extension by which we represent the data cloud not only by the nodes of the network, but also by the line and plane segments between the nodes, is necessary for the network to converge correctly to the complicated hand topology. The tracker can be further improved by adding constraints to the hand model, in accordance with the anatomy of the hand.

Our method can easily be applied not only to hand tracking, but also to other types of objects, e.g., the human body or animals, by simply varying the SOM network topology. In further developments we will use the network topology to infer useful features for gesture recognition.

Acknowledgments. This research is supported by the German Ministry of Education and Research (BMBF) under grant number 01IS10049B. A. State would like to thank for the support by The German Academic Exchange Service Programme "Ostpartnerschaften", the University of Lübeck and The Sectoral Operational Programme Human Resources Development 2007-2013 of the Romanian Ministry of Labor, Family and Social Protection through the Financial Agreement POSDRU/86/1.2/S/61756.

References

1. ElKoura, G., Singh, K.: Handrix: Animating the Human Hand. In: Proc. SCA, pp. 110–119 (2003)
2. Haker, M., Böhme, M., Martinetz, T., Barth, E.: Deictic Gestures with a Time-of-Flight Camera. In: Kopp, S., Wachsmuth, I. (eds.) GW 2009. LNCS, vol. 5934, pp. 110–121. Springer, Heidelberg (2010)
3. Kohonen, T.: Self-Organizing Maps. Springer, Berlin (1995)

4. Rehg, J., Kanade, T.: Visual Tracking of High DOF Articulated Structures: An Application to Human Hand Tracking. In: Eklundh, J.-O. (ed.) ECCV 1994. LNCS, vol. 801, pp. 35–46. Springer, Heidelberg (1994)
5. Segen, J., Kumar, S.: Shadow Gestures: 3D Hand Pose Estimation using a Single Camera. In: Proceedings of Conference on Computer Vision and Pattern Recognition (1999)
6. Rosales, R., Athitsos, V., Sclaroff, S.: 3D hand pose reconstruction using specialized mappings. In: Proc. International Conference on Computer Vision, vol. 1, pp. 378–385 (2001)
7. Athitsos, V., Sclaroff, S.: Estimating 3D hand pose from a cluttered image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2003)
8. Stenger, B., Thayananthan, A., Torr, P.H.S., Cipolla, R.: Hand Pose Estimation Using Hierarchical Detection. In: Proc. International Workshop Human-Computer Interaction, pp. 105–116 (2004)
9. Bray, M., Koller-Meier, E., Gool, L.V.: Smart particle filtering for 3D hand tracking. In: Sixth IEEE International Conference on Automatic Face and Gesture Recognition, p. 675. IEEE Computer Society, Los Alamitos (2004)
10. Haker, M., Böhme, M., Martinetz, T., Barth, E.: Self-Organizing Maps for Pose Estimation with a Time-of-Flight Camera. In: Kolb, A., Koch, R. (eds.) Dyn3D 2009. LNCS, vol. 5742, pp. 142–153. Springer, Heidelberg (2009)
11. Ehlers, K., Timm, F., Barth, E., Martinetz, T.: A generalization of k-means for real time hand skeleton tracking (in preparation)
12. Kinect home page, http://www.xbox.com/en-US/kinect/

Trajectory Analysis on Spherical Self-Organizing Maps with Application to Gesture Recognition

Artur Oliva Gonsales and Matthew Kyan

Ryerson University
350 Victoria Street, Toronto, Ontario, Canada, L5J 1C9
aolivago@ryerson.ca, mkyan@ee.ryerson.ca

Abstract. We propose a new approach to gesture recognition using the properties of Spherical Self-Organizing Map (SSOM). Unbounded mapping of data onto a SSOM creates not only a powerful tool for visualization but also for modeling spatiotemporal information of gesture data. Once mapped onto a SSOM the gesture data is treated as a series of postures. A set of postures describing a specific path on the SSOM for a gesture is used as a trajectory. Although some trajectories may share the same postures, the path consisting of posture transitions will always be unique. Different variations of posture transitions occurring within a gesture trajectory are used to classify new unknown gestures. Experimental results on datasets involving full body and hand gestures show the effectiveness of our proposed method.

Keywords: spherical SOM, gesture recognition, trajectories.

1 Introduction

Nowadays, many applications require the use of powerful visualization tools that can assist data analysts in evaluating their data. This allows deriving meaningful inferences, therefore gaining deeper understanding about the physical phenomenon characterizing their data. One example of such visualization tool is immersive virtual reality which was created due to recent advances in research and can provide a rich visualization and interactive modeling and analysis tool [1]. Such advanced, interactive, and task-driven display and analysis tools utilize a full range of human sensorimotor capabilities and provide an insight on large volumes of experimentally acquired data. The data modeling approach discussed in this paper based on trajectory analysis presents a different view into the use of Self-Organizing Maps, which gives a viable mechanism to generate a spatio-temporal representation of data from multidimensional data. Several gesture data sets are used to demonstrate the effectiveness of the proposed methodology in building spatio-temporal trajectories.

In this paper, we create gesture trajectories with the help of Self-Organizing Maps (SOM) [2]. In particular, we use the Spherical SOM structure (SSOM) [3], because of its ability to map multi-dimensional data without boundaries. SOM is an unsupervised clustering approach proposed by Kohonen [2] that clusters data from high-dimensional

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 125–134.
DOI: 10.1007/978-3-642-35230-0_13

space into low-dimensional space, while still preserving its topology. For sequences of input data whose features are expected to temporarily change in a smooth way, it is anticipated that a topology preserved mapping can allow for the formation of a smooth trajectory on the map. Regular SOMs map the data points onto a flat 2D lattice during training by updating the weights of the nodes in the lattice. However, this setup has a restricted boundary, normally pushing the data to be mapped along its boundaries. Also, it can be argued that the opposite sides of the boundary are not topologically close in the SOM space. A more optimal choice for SOM structure consists of sphere [3], which minimizes topological discontinuity. This SOM structure is created by subdividing an Icosahedron, providing the SOM structure with a symmetric node distribution depicted in Fig. 1. One of the advantages of the SSOM is that regions of density found in the feature space will map equally spaced and well separated locations on the sphere due to the wrap-around effect of the lattice. In this work, we leverage this property to build trajectory based features to distinguish between human full body motion gestures.

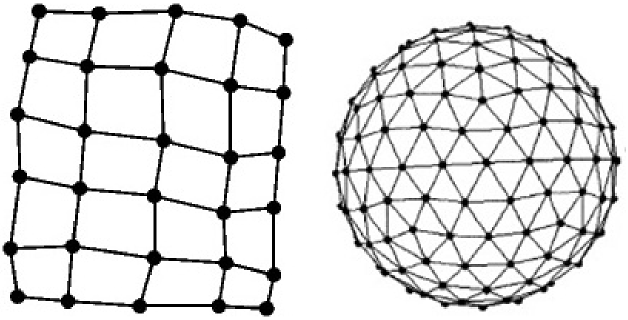


Fig. 1. Open vs. Closed structure of 2D- and Spherical SOM

The rest of the paper is organized as follows: Section 2 discusses the related works in gesture recognition using SOMs. Section 3 gives details about the experimental setup and the datasets. Section 4 provides experimental results. Finally, in Section 5 we conclude the paper with a conclusion.

2 Related Work

The use of SOFM in the area of gesture recognition has been relatively recent. Some methods which will be discussed shortly in this section have used SOFM in various ways to divide the sample data into clusters of phases and are further processed with the help of other tools and techniques. In [4], Oshit and Matsunaga use a SOM to first process the gesture data and then apply a Support Vector Machine (SVM) to partition the feature space into regions belonging to separate classes. Their approach is interesting because they divide each gesture into short phases and then apply a pattern recognition technique for multi-dimensional data to recognize each phase. By using Dynamic Programming (DP), authors match the trajectory from the input signals and the sample trajectory from a gesture. By *trajectory* we refer to the temporal path that the data maps into on the SOM lattice based on a set of consecutive Best Matching Units (BMUs). The disadvantage of this approach is that the trajectories must be projected onto low-dimensional feature space. Furthermore, a valid threshold must be specified to measure the similarity of two trajectories. One of the limitations of standard SOM that is tried to be overcome in this work is to eliminate the restricted boundary of a 2D lattice. This is due to the boundaries being open and nodes on the boundaries not having the same number of neighbors as the inner nodes.

A. Shimada and R. Taniguchi in [5] use a Sparse Code of Hierarchical SOM (HSOM). A Hierarchical SOM [6] is a two layer SOM network, where the lower layer has a connection with an input layer. In this case, the second layer receives an input vector from the first layer directly. The method proposed by Shimada uses the property of HSOM to first learn postures (minimum unit of a gesture) in the first layer, and the learn short gestures consisting of some time-series postures in the second layer. Authors argue that the time length of a human gesture is not always the same even if same gestures are compared. They highlight that the key issue in their method is to absorb the time variant appropriately in order to make clusters which include the same gesture class.

The interesting part of the approach by Shimada and Taniguchi for gesture recognition is how they tackle the problem of time invariance or length invariance, speaking in terms of trajectories. The use of multi-layer SOM allows them to obtain a more general gesture path on the SOM lattice without worrying about its length.

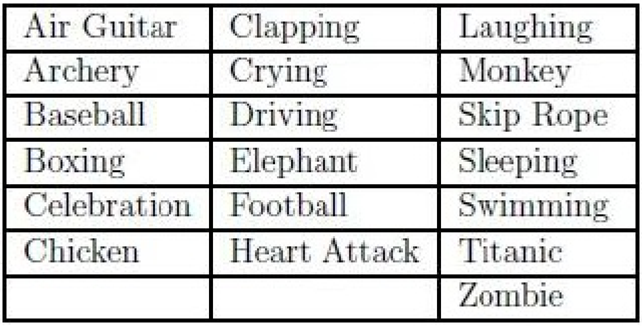
Another method to gesture recognition is suggested in *Video-Based Gesture Recognition Using Self-Organizing Feature Maps* [7], [8]. This work introduces a probabilistic recognition scheme for hand gestures, where SOFMs are used to model spatiotemporal information extracted from images. It uses a combination of SOFM and Markov models for gesture classification. The classification scheme consists of tracking the transformation of gesture representations from a series of coordinate movements.

3 Experimental Setup and Datasets

All of the methods discussed previously and most seen in literature involve the use of two-dimensional SOFMs. The attempt in this paper is to show how the properties of a SOFM can be used for smoothly varying data such as body or hand gestures to create a good recognition system by implementing a 3D version of the SOFM. Following this section is an overview of the experimental set up and the datasets used.

3.1 Dataset

A dataset involving full body gestures was used in this work. This dataset was collected using sensor equipment in the Microsoft Kinect camera. A virtual version of the game Charades was used to collect full body gesture data. Nineteen gestures were selected randomly out of a classic commercial version of Charades. Figure 2 alphabetically lists the 19 different gestures that were used in the database. It is easy to see how these gestures are very open to interpretation. Of the 19 gestures (classes), 50 full samples of each gesture were sampled. The Kinect primarily samples user 'gesture' information from the IR depth camera. The data coming from the camera is oriented relative to its distance from the Kinect. This becomes problematic when searching for the solution to universal truths in gestures. Normalization was used to that convert all depth and position data into vectors relative to a single joint presumed most neutral. In this case the torso was considered as the neutral position of the body. Figure 2 shows the skeleton model with the points (body parts) used in the dataset. The result includes positive and negative x, y, and z-axis values. The feature vector consists of 60 features (three displacement vectors -x,y,z multiplied by 20 body points). The average temporal length of each gesture in the database is 200-300 frames.



Air Guitar	Clapping	Laughing
Archery	Crying	Monkey
Baseball	Driving	Skip Rope
Boxing	Elephant	Sleeping
Celebration	Football	Swimming
Chicken	Heart Attack	Titanic
		Zombie

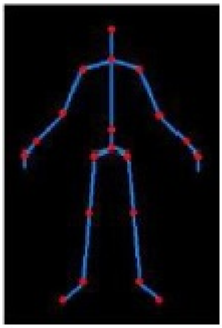


Fig. 2. Microsoft Kinect Full body gesture data and skeleton showing the gestures (left), body parts (right) being tracked

3.2 Experimental Setup

In all the experiments performed in this work the setup was identical. A Spherical SOFM was used with specific settings and size which will be discussed shortly. All the experiments were performed on standalone PC with Windows 7, 4GB of RAM and Intel Core i7 CPU (2.67GHz). MATLAB R2011b environment was used for all the experiments and the visualization part. On average it took several minutes to train the network with one gesture depending on the feature vector size of a specific gesture.

3.3 SSOM Training

The training phase of the SSOM is identical to the conventional 2D SOM [2]. Let, the input space of N nodes be represented by $\chi = \{x_i\}_{i=1}^{N}$. Let the SSOM be represented by M nodes ($M << N$). Each node in the SSOM lattice has a corresponding weight vector w. All these weight vectors together represent the SSOM space $\Psi = \{\psi_i\}_{i=1}^{M}$. Each node also has a *neighborhood* associated with it. A neighborhood is a set of nodes consisted of the node itself and its neighbors. Let, the neighbourhood set for node *i* be represented by Θ_i^r. Here, *r* represents the neighborhood spread, *r = 1,…,R. R* is the maximum neighborhood radius, which is set to a value such that it covers half of the spherical space [3].

The input nodes are randomly introduced to the SSOM during training. For each voxel, a *Best Matching Unit (BMU)* among all the nodes is selected. BMU is the node which is closest to the input voxel according to some similarity measure. Euclidean distance is usually used as distance measure. The update step then takes place, where the weight vector of the BMU and its neighboring nodes (Θ_{BMU}^r) are updated in a way so that they are pulled closer to the weight of the input voxel. After training, the SSOM weight vectors are arranged in such a way that represents the underlying distribution of the input data (the node features in this case). The training algorithm is described below:

- **Initialization:** The weight vectors of the SSOM nodes are initialized first. Random values can be used for initialization, but as pointed out by Kohonen et al. in [2], random initialization will take more time to converge. A count vector $0 = \{o_i\}_{i=1}^{M}$ [3] is used to keep track of the hits to each node. This vector is initialized to zero. This is used in the BMU selection step (explained below) to prevent cluster underutilization.

  Training: For each input *x* (a feature vector containing the coordinates of the 20 body points), do the following:

— **BMU Selection:** Calculate the Euclidean distance of the node feature vector x with all the nodes as follows:

$$e_i = (o_i + 1)\left|\left|x - w_i\right|\right|, i = 1, \ldots M \quad (1)$$

BMU is the node for which this distance is the smallest.

— **Weight Update:** Update the weights for the BMU and its neighboring nodes (defined by Θ_{BMU}^r) as follows:

$$w = w + b(t) * h(s,r) * \left|\left|x - w\right|\right|, \quad (2)$$

$$c_w = c_w + h(s,r), \quad (3)$$

Where $w \in \Theta_{BMU}^r, b(t) = \alpha e{-}\frac{t}{T} and\, h(s,r) = e{-}\frac{r^2}{s*R}$.

The functions *b(t)* and *h(s,r)* control the rate of learning and neighborhood effect, respectively. *b(t)* decreases in value as the epoch number *t= 1,2,…T* increases. It also depend on the learning rate α. *h(s,r)* depends on the neighborhood size parameter *s,* which is user controlled. *h(s,r)* is a Gaussian function. The further a neighboring node is from a BMU, the less its weight will be affected.

- Repeat the training steps for a pre-defined number of epochs (20)

The main control parameters in SSOM training are the learning rate α, the number of epochs T and the neighborhood size parameter *s*. In the following experiments, the setting used were T=20, and *s*=4.

4 Experimental Results

Figure 3 shows some sample trajectories that were obtained during the mapping process. These gesture trajectories are a representation of the BMUs hit sequence that gesture mapped onto the SSOM. The data that is being used in the trajectory mapping comes from the training portion of the datasets. All the BMUs are in 3D space although they appear as 2D images. The lattice of the Spherical SOFM was removed on purpose so that the trajectories could be seen more clearly.

Each gesture class was displayed at a different angle from the rest in order to show the data path more clearly. From Figure 3 it is evident that the trajectories for each gesture class trace a similar if not identical path on the spherical lattice of the SOFM. It is also clear that each gesture leaves a path which is unique if comparing to other gestures. It is important to note that many gestures may share common BMUs since they may contain similar postures that trace a specific c gesture. We count every BMU hit of a gesture as a posture belonging to a given gesture class. A collection of these postures form a gesture. In the next sections the methods for gesture recognition will be described.

4.1 Gesture Recognition: Using All Postures (Method 1)

The gesture recognition and classification initially starts with a simple approach. As mentioned earlier, all the BMUs that trace a trajectory for a specific gesture are considered as postures. All the BMUs from each gesture class are used as a collection of points or postures for the purpose of classification of new unknown data. This is done in the following manner:

1. All the BMUs falling into trajectories belonging to a specific gesture are recorded into a set G_i as follow:

$$G_i = \{Tr_{(i,1)}, \dots, Tr_{(i,m-1)}, Tr_{(i,m)}\}, \tag{4}$$

Where $Tr_{(i,j)}\ for\ j = 1,2, \dots m$ is a trajectory forming a gesture G_i and i is the gesture index which represents a gesture class, also

$$Tr_{(i,j)} = \{P_k, \dots, P_{n-1}, P_n\}, \tag{5}$$

Where P_k is the k^{th}node in the SSOM lattice (i.e. posture) and *n* is the number of nodes or postures in the trajectory $Tr_{(i,j)}$.

2. Feature vectors (consisting of the coordinates of the body parts) of an unknown gesture coming from the testing portion of the dataset are then compared against the weights of the SOFM and the BMUs from the new trajectory of an unknown gesture are collected into a new set *Tp*.
3. A frequency posture counter K_i assists in determining the class of the unknown gesture, where i represents the index of a known gesture. The counter K_i for a gesture i is incremented if a posture from an unknown gesture belongs to a gesture

being compared against. This way, Tp is compared against all the G_i in the database. So, if $K_i \geq K_{1,}K_2, \ldots, K_n$, where K_n is a counter belonging to a gesture with index n, then K_i is chosen as the winning counter the unknown gesture is classified as gesture G_i.

4.2 Gesture Recognition: Weighted Aggregation of All Postures (Method 2)

The approach taken for gesture recognition in the last section only takes into consideration a set of postures as a primary data to classify an unknown gesture. This set does not take into account the frequency with which a specific posture is encountered in a gesture's trajectories. For this reason a frequency factor is introduced in this approach. All postures are aggregated into a set, but at the same time each posture is associated with a weight. The more a posture appears in a gesture path while training the network, the more weight it has towards that gesture. The reason behind the weight factor is that the path that a trajectory representing a specific gesture maps on the lattice of the SOFM tends to activate the same neurons (postures), giving it a higher probability to appear again if the same gesture is traced.

4.3 Gesture Recognition: Using Posture Transitions (Method 3)

Previous methods tested, treated each BMU as a posture and no dynamic information was used. Dynamic information in this case refers to the posture transitions that occur during the tracing of a gesture onto the SOFM. The main argument here is that trajectories belonging to the same gesture should follow not only the same path on the Spherical lattice but also have similar transitions in terms of postures. For example, when a person performs a "Driving" gesture, he or she will follow the same posture transition as he or she moves the hands in a 3D space. A *similar* posture transition is defined as having two identical consecutive BMU hits from node *A* to node *B,* in two different trajectories. For this specific reason the classification of an unknown gesture is evaluated based on similar posture transitions during the formation of its trajectory.

4.4 Gesture Recognition: Weighted Aggregation of All Posture Transitions (Method 4)

The last approach in this paper uses a weighted aggregation of all posture transitions. Similarly as in the method with weighted aggregation of all postures a weight is introduced. This approach not only takes in consideration the posture transitions happening in a gesture trajectory but also the frequency with which these transitions occur.

4.5 Discussion

Tables 2-5 show the gesture recognition rate for all the approaches implemented in this work. From Fig. 3 it is evident that a good data separation is obtained, which can

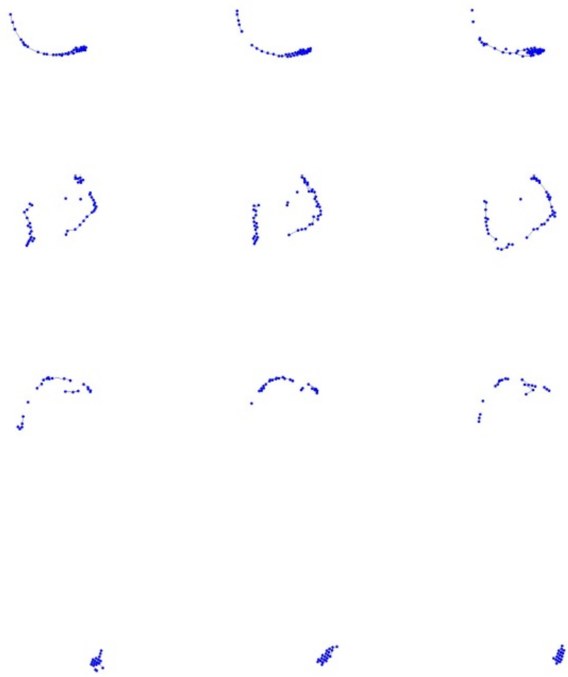


Fig. 3. Sample Microsoft Kinect Gesture trajectories. (Three samples for four gestures). From first to last row: Air Guitar, Archery, Baseball, Boxing.

Table 1. Recognition rate: using all postures (Method 1).

Gesture	Rate %	Gesture	Rate %	Gesture	Rate %
Air Guitar	80(20)	Clapping	52(13)	Laughing	80(20)
Archery	72(18)	Crying	56(14)	Monkey	84(21)
Baseball	76(19)	Driving	68(17)	Skip Rope	64(16)
Boxing	88(22)	Elephant	64(16)	Sleeping	72(18)
Celebration	84(21)	Football	68(17)	Swimming	96(24)
Chicken	48(12)	Heart Attack	80(20)	Titanic	52(13)
Zombie	60(15)				

Table 2. Recognition rate. Weighted aggregation of all postures (Method 2).

Gesture	Rate %	Gesture	Rate %	Gesture	Rate %
Air Guitar	100(25)	Clapping	92(23)	Laughing	100(25)
Archery	100(25)	Crying	88(22)	Monkey	92(23)
Baseball	96(24)	Driving	100(25)	Skip Rope	80(20)
Boxing	88(22)	Elephant	96(24)	Sleeping	28(7)
Celebration	100(25)	Football	100(25)	Swimming	100(25)
Chicken	48(12)	Heart Attack	84(21)	Titanic	96(24)
Zombie	100(25)				

Table 3. Recognition rate. Using posture transitions (Method 3).

Gesture	Rate %	Gesture	Rate %	Gesture	Rate %
Air Guitar	100(25)	Clapping	100(25)	Laughing	100(25)
Archery	100(25)	Crying	80(20)	Monkey	100(25)
Baseball	88(22)	Driving	100(25)	Skip Rope	80(20)
Boxing	60(15)	Elephant	44(11)	Sleeping	100(25)
Celebration	96(24)	Football	100(25)	Swimming	100(25)
Chicken	76(19)	Heart Attack	100(25)	Titanic	100(25)
Zombie	100(25)				

Table 4. Recognition rate. Weighted aggregation of all posture transitions (Method 4).

Gesture	Rate %	Gesture	Rate %	Gesture	Rate %
Air Guitar	100(25)	Clapping	100(25)	Laughing	100(25)
Archery	92(23)	Crying	100(25)	Monkey	100(25)
Baseball	100(25)	Driving	100(25)	Skip Rope	92(23)
Boxing	100(25)	Elephant	100(25)	Sleeping	100(25)
Celebration	100(25)	Football	100(25)	Swimming	96(24)
Chicken	100(25)	Heart Attack	100(25)	Titanic	88(22)
Zombie	92(23)				

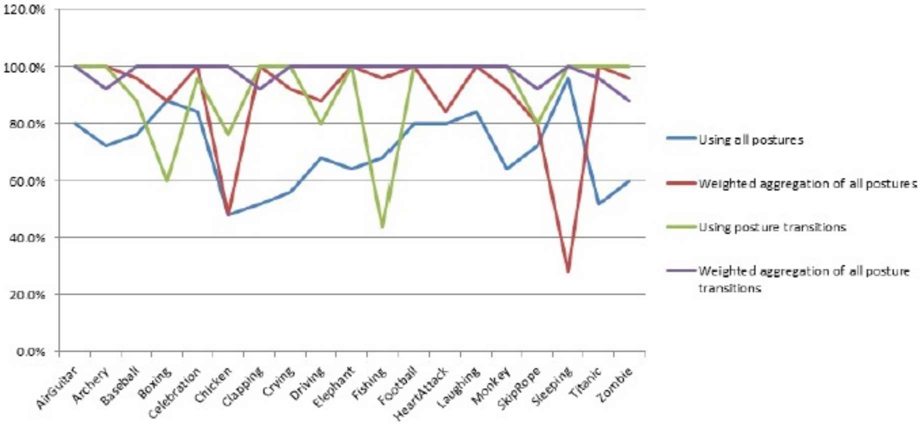


Fig. 4. Gesture recognition comparison chart

be seen from the trajectories: different gestures have different trajectories. As discussed earlier, such separation is reached because of the wrap-around effect of the SSOM lattice. It is also clear that a higher classification rate is obtained when using the dynamic structure of the trajectories such posture transitions. The reason why, for instance, Method 4 works better than others is because it uses dynamic information (posture transitions) that other methods do not. By introducing a weight factor, we increase the chances of classifying unknown gesture correctly, because gestures trajectories tend to have the same transitions from one posture to the next. Fig. 4 depicts all the results in a chart, clearly showing the advantages of using posture transitions with implementation of frequency weights.

5 Conclusions

In this paper, we have proposed the use of SSOM for trajectory analysis with application to gesture recognition. We implemented four different approaches to classify new gesture data, clearly showing the advantages of using the dynamic structure of the gesture data. An overall result of 97.9% of correct classification was obtained by using the *weighted aggregation of all posture transition* method. As a future work, we would like to seek methods for describing full trajectories with a descriptor. The challenge that lies in creating such a descriptor is the length of the sample, which is always variable. Creating trajectories from samples of variable length (i.e. gesture data) also makes the trajectories to be different in its lengths. A low frequency descriptor such as Fourier Descriptor may be used to describe a trajectory, but first the length factor has to be addressed. The advantage of using a spherical SOM is that it offers a constrained spherical coordinate system on which such a descriptor can be based.

References

1. Furht, B. (ed.): Immersive virtual reality. Encyclopedia of Multimedia. Springer (2006)
2. Pratt, W.K. (ed.): Digital Image Processing. John Wiley and Sons, New York (2007)
3. Sangole, A., Leontitsis, A.: Spherical self-organizing feature map: An introductory review. International Journal of Bifurcation and Chaos 16, 3195–3206 (2006)
4. Oshita, M., Matsunaga, T.: Automatic Learning of Gesture Recognition Model Using SOM and SVM. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Chung, R., Hammoud, R., Hussain, M., Kar-Han, T., Crawfis, R., Thalmann, D., Kao, D., Avila, L. (eds.) ISVC 2010, Part I. LNCS, vol. 6453, pp. 751–759. Springer, Heidelberg (2010)
5. Shimada, A., Taniguchi, R.I.: Gesture recognition using sparse code of hierarchical SOM. In: International Conference of Pattern Recognition (ICPR), pp. 1–4 (2008)
6. Lampinen, J., Oja, E.: Clustering properties of hierarchical self-organizing maps. J. Mathematical Imaging and Vision 2(2-3), 261–272 (1992)
7. Caridakis, G., Pateritsas, C., Drosopoulos, A., Stafylopatis, A., Kollias, S.D.: Probabilistic Video-Based Gesture Recognition Using Self-organizing Feature Maps. In: de Sá, J.M., Alexandre, L.A., Duch, W., Mandic, D.P. (eds.) ICANN 2007. LNCS, vol. 4669, pp. 261–270. Springer, Heidelberg (2007)
8. Caridakis, G., Karpouzis, K., Pateritsas, C., Drosopoulos, I., Stafylopatis, A., Kollias, D.: Hand trajectory based gesture recognition using self-organizing feature maps and markov models. In: Internation Conference on Multimedia and Expo (ICME), pp. 1105–1108 (2008)

Image Representation Using the Self-Organizing Map

Leandro A. Silva, Bruno Pazzinato, and Orlando B. Coelho

College of Computing and Informatics, Mackenzie Presbyterian University, São Paulo, Brazil
{leandroaugusto.silva,orlandoc}@mackenzie.br,
bruno.pazzinato@gmail.com

Abstract. This paper introduces a new approach to image representation for multimedia databases based on the Self-Organizing Map (*SOM*) neural network. The distance between each image from a database and the SOM weight vectors trained on the same database is used as a representation for the image. In order to assess the performance of this proposal we compare it with a reference technique in image representation: the *Thumbnails* method. The results are satisfactory for an initial experiment since it was possible to identify the effectiveness of the *SOM*-based proposed representation. In order to verify the efficiency of the representations, a classification experiment is performed using the k-*NN* algorithm. For all image representation experiments, the *SOM* approach outperforms the *Thumbnails* reference technique. For example, in one experiment the representation results in a reduction of image size to 2% of its original size and the correct classification rates achieved are 83.33% and 35.42% for *SOM* and *Thumbnails* respectively.

Keywords: multimedia database, image representation, self-organizing map, thumbnails.

1 Introduction

The use of multimedia information, such as image, video or audio, is widespread for computer users either in a home setting, like looking at photos in a cell phone, or in a professional one, as applies to a doctor examining a computerized tomography.

For an efficient access to multimedia information is common that the data is stored in specific databases, the so-called Multimedia Databases. Lew et. al. [1] defines a Multimedia Database as a system that can store and retrieve multimedia objects, such as two-dimensional color images, medical images in 2-D or 3-D grayscale, voice or music, video clips and even transactional data. For Rakow, Neuhold and Löhr the large amount of data and the high cost of transmitting Multimedia data are the main reasons for not using conventional databases [2].

In addition to the storage problem associated to using multimedia data in a conventional database, the standard query system supported by conventional databases is also a major problem. For example, in case a user needs to find an image, the use of SQL language (Structured Query Language)-based queries, which were developed to be used for structured data, is only possible if based on metadata. However, the

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 135–143.
DOI: 10.1007/978-3-642-35230-0_14 © Springer-Verlag Berlin Heidelberg 2013

content-based image retrieval (CBIR), which the query is based on the image characteristics like, for example, color and shapes, is an approach much more appropriate for a multimedia database, is still a challenging problem. The main reason for this is the difficulty in finding a representation technique for characterizes images which ensure a good compromise between representation and dimensionality [1].

In problems using medical images, where the data dimension is huge, a method that has proven effective if compared to many others is the so called *Thumbnails* [3]. This method is based on a simple algorithm that eliminates rows or columns from the original image in order to reduce its dimensionality [3], [4], [5].

Motived by the difficulty in finding an efficient representation technique for images and by the need to improve the results accomplished by the *Thumbnails* technique, this work proposes a novel method, based on the Self-Organized Map (SOM) Neural Network [6]. The method has the advantage of performing well for different ratios of image reduction and for colour images.

The remainder of the paper is organized as follows: the image representation problem is discussed in Section 2. The proposed approach to image representation based on the SOM is presented in Section 3. Experimental results are presented and discussed in Section 4.

2 The Image Representation Problem

Image representation could be defined as the process of extracting characteristics that describe the content of an image, such as color, texture or shape. Thus, each image in an image database is represented by low-dimensional vectors using methods of feature extraction. Fig. 1 shows a schematic illustration of feature extraction. In the figure, hand and foot images are represented by features combined in a vector which is called a feature (or content or signature) vector ($\mathbf{v} \in \Re^d$). The feature vector has to maintain the main characteristics of an image in low-dimension, at the same time also

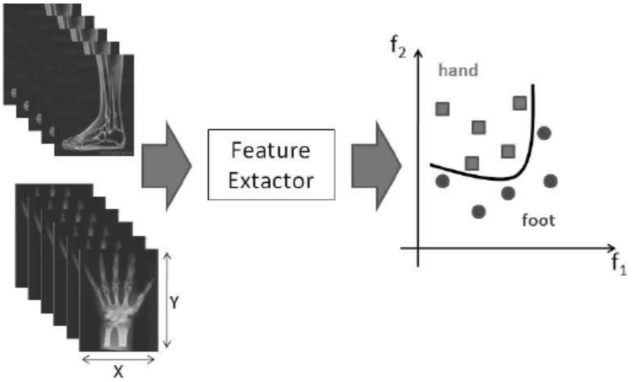


Fig. 1. Illustration of a feature extractor to generate an image representation [5]

maintaining the discrimination of the class, in this example, foot and hand. The feature extraction is an essential part of a multimedia database [1], [2], [3], [4], [5].

There are many feature extraction techniques in the literature such Discrete Wavelet Transform Tamura texture, Castelli texture, Discrete Cosine Transform, edge characteristics of the Canny detector and *Thumbnails* [6]. In experiments with medial images, the *Thumbnails* technique has shown good results for representation image [3], [4], [5]. After this, in the medical image context, this technique is considered as a baseline method, which is easily implemented and with satisfactory results in the literature. In the next subsection the *Thumbnail* process is quick introduced.

2.1 Image Representation with *Thumbnails*

The *Thumbnail* technique algorithm is a way of reducing the dimensionality of the data. In applications that do not require high quality (as it is *not* the case of medical applications, for example), the Thumbnails method does not deliver the best result, but allow for fast processing, when compared to more traditional methods for dimensionality reduction, like Principal Component Analysis [8]. This higher speed to generate the feature vector is due to the simplicity of the method. The idea of the algorithm is to remove rows and columns of the original image, depending on the desired dimension reduction. In a simple example, with a grayscale image represented as a matrix where each element is the pixel value of the image (with 0 representing black and 255 representing white), one can decrease the size of an image to a quarter of its original size by removing one row and one column every two rows or columns.

In general, an original image with width W and height H, when rescaled by N using *Thumbnails*, will generate a reduced image with width W/N and height H/N, as depicted in Fig. 2, for N = 2. In this example, if the image is transformed into a vector, the number of elements of the feature vector is reduced from 100 positions to only 25 positions. Note that the rows and columns can be reduced either by removing every other row or column or by replacing each adjacent pair of rows or columns by its average value.

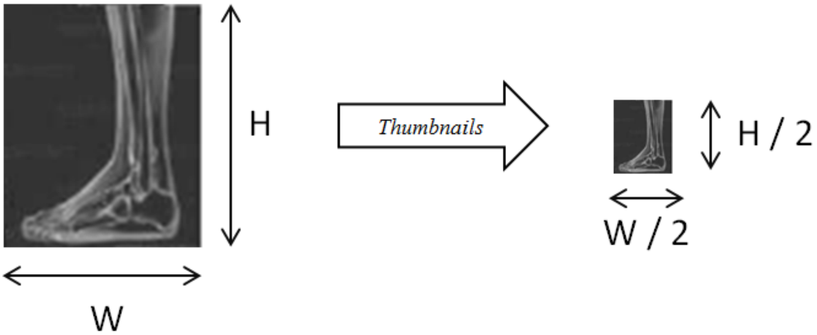


Fig. 2. An example using *Thumbnails* to reduce by 4 the foot image

3 The Proposed Solution: *SOM*-Based Image Representation

3.1 The Self-Organizing Map

A Self-Organizing Map (SOM) consists of neurons located on a regular low-dimensional grid, usually two-dimensional (2-D). The lattice of the 2-D grid is either hexagonal or rectangular. Assume that each input pattern from the set of patterns (X) $\mathbf{x}^{\mu}$ is defined as a real vector $\mathbf{x}^{\mu} = [x_1, x_2,..., x_d]^{T} \in \Re^{d}$. Each neuron has a d-dimensional weight vector $\mathbf{w} = [w_1,w_2, ...,w_d]^{T} \in \Re^{d}$ called a prototype [6].

The SOM training algorithm is iterative. Initially, in $t = 0$, the weight vectors are randomly initialized, preferably from the input vectors domain [6]. At each training step, an input pattern $\mathbf{x}^{\mu}$ is randomly chosen from a training set (X). General distances between $\mathbf{x}^{\mu}$ (t) and all weight vectors $\mathbf{w}_{ij}$, are computed, where i and j are the grid indices of the SOM Map. The winning neuron is the prototype closer to $\mathbf{x}^{\mu}$ (t) or the Best Match Unit (BMU). The BMU weight vector is updated, as well as the vector of weights of neighboring neurons, although with minor intensity (see [6] for the complete SOM training algorithm).

SOM is especially suitable for data survey because it has prominent visualization properties. It creates a set of prototype vectors representing the set of input patterns and carries out a topology-preserving projection of the prototypes from the n-dimensional input space onto a low-dimensional grid. This ordered grid can be used as a convenient visualization surface for showing different features of the SOM (and thus of the input patterns); for example, its underlying cluster structure [5]. In this work, this topology-preservation property is explored to verify the distance relationship between specific maps regions, which can be an interesting approach for image representation. This approach is developed in the next subsection.

3.2 Image Representation Using *SOM*

The image representation with the *Thumbnails* technique is performed with the pixel values of the reduced images. On the other hand, the image representation process using *SOM* is conducted by comparing the image converted into vector form to the weight vectors of the neurons of the map.

In order to use the SOM, the map must be trained as follows:

1. Each image in the database is converted to a vector $\mathbf{x}^{\mu}$, which will form a matrix X of image vectors, called the training matrix.
2. The training matrix is always shuffled so that the vector reading order does not influence the SOM training.
3. The number of neurons in the SOM grid, i and j, is defined. This defines the reduction dimension for the image representation.
4. The SOM map, finally, is trained

After training, each image vector $\mathbf{x}^{\mu}$ from the matrix X is again showed to the SOM. The Euclidean distance (or any alternative metric used by the SOM algorithm) between $\mathbf{x}^{\mu}$ and all weight vectors $\mathbf{w}_{ij}$ is measured. The matrix composed by the distances so calculated can then be used as an image representation, whose dimension is much smaller than the original image. In Fig. 3 there is a schematic example of this representation for a single image.

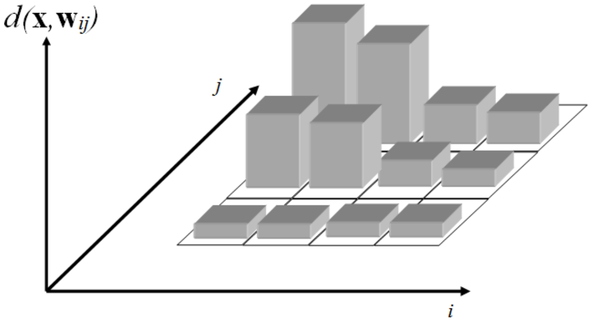


Fig. 3. Schematic example of SOM image representation

In summary, these are the steps to represent an image with SOM:

1. A vector image is picked from the training matrix.
2. The distance between this vector and the weight vectors is calculated.
3. The distances so measured produce a series of values which is then taken as the image representation.

The *SOM*–based approach for image representation, which is proposed herein, has the advantage of reducing the original image dimension in any desired scale, since the reduction is defined by the number of neurons in the SOM Map. In comparison, the *Thumbnail* technique is limited to work with dimensions that are multiple of two.

For colour images, thare is further advantage in using the *SOM*-based representation, since all RGB bands are represented in a single step. On the other hand, for the *Thumbnails*-based representation, it is necessary to perform a reduction for each RGB band.

4 Experimental Results

The system was implemented in MATLAB, using the SOM Toolbox [8].

For the experiments two different databases were used. The first is a public database available in Image Processing Place (http://www.imageprocessingplace.com/). This database is formed by monochromatic images of different objects with forms and their geometric transformations such as rotation, scale and translation. A sample of 3 classes (apple, bone and mug) chosen from the 60 images is show in Fig. 4.

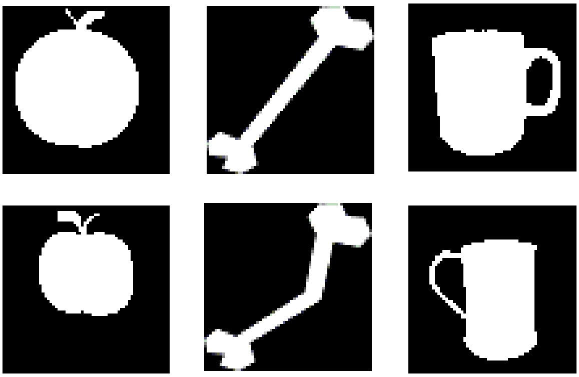


Fig. 4. Examples of images in Database 1

The second database is a synthetic one, with three geometric pictures generated for this experiment. Each image class (circle, square and triangle) is modified with application of geometric transformation (rotation, scale and translation) and noise. For each class, 20 different images are generated, what results in a database with 60 images. A sample of this database is show in Fig. 5.

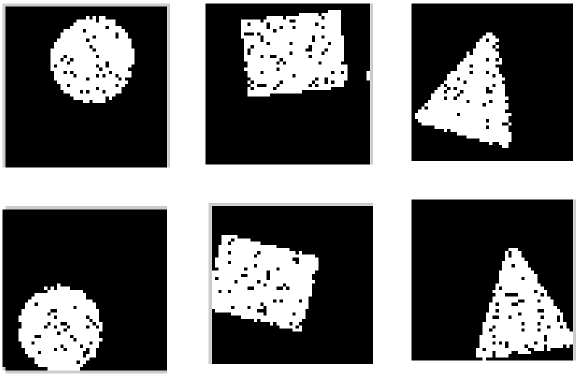


Fig. 5. Examples of images in Database 2

For the experiments, the following steps are taken:

1. The image size for images in Database 1 was standardized to 64×64.
2. Image representations were generated of in two differences sizes, for both representations techniques (*SOM* and *Thumbnails*): 10×10 and 25×25.

In order to compare the representation power of both techniques, after the each representation is generated, the resulting image is changed into a vector and submitted to a classification task by the k-*Nearest Neighbor* (k-*NN*) algorithm. The integer value *k* is taken to be 1 for all experiments. The correct classification results are used as a comparison metric. They are summarized in Table 1 and Table 2.

Table 1. Classification results for Database 1

Database 1	**10×10**	**25×25**
Thumbnails	35.42%	35.42%
SOM	81.25%	83.33%

Note that the original image dimension for images in Database 1 is 64 x 64. For the 10 x 10 representation, the image is resized to 2% of its original size and, for the 25 x 25 representation, the original image is resized to 15%. of its original sizes. In both cases SOM-based technique outperforms *Thumbnails*.

Table 2. Classification results for Database 2

Database 2	**10×10**	**25×25**
Thumbnails	31.25%	33.33%
SOM	35.42%	41.27%

In the case of Database 2, as shown in Table 2, the SOM representation again outperforms the *Thumbnails* technique in both dimensions, 10 x 10 and 25 x 25. Note however that, the low classification performance for both techniques can be the result of the high index of image deformation when applying rotation, translation and noise. In this process, a range between 20 and 50 was considered in each deformation, where using an index of 30 for a square object means that the square can be translated 30 points (in north, south, east or west), rotated in 30 degree (positive or negative), scaled in 30 points (in width or height) and noised in such a way that the value of 30 pixels can be changed.

5 Conclusions

This work shows a new approach to image representation based on the SOM neural network. It is shown that this is a real alternative to *Thumbnails*, which is considered as a reference method when used in medical images. The two drawbacks of this reference method are its difficulty in working with images in different relations of width and height and when these dimensions are not a multiple of 2. Moreover, for using it in colour images, a representation for each RGB channel has to be developed.

On the other hand, the *SOM*-based representation works with images that display any relation between width and height sizes and does not need any additional processing when working on colour images. In addition, this approach positively explores one of the most discussed issues about the *SOM*: the map size. This value is used here as a parameter that defines the image representation size.

Furthermore the experimental results using two databases show that the SOM-based representation achieves the best results. In order to compare the results, the k-*NN* classifier was used, with k=1, and the correct classification was used as a comparative index.

For a real database (Database 1), with 3 images classes, the *SOM* and *Thumbnails* image representations were experimented in two versions: 10 x 10 and 25 x 25. These representations amount to a reduction to 2% and 15% of the original image, respectively. In both cases the difference between classification results from Table 1 showed that the *SOM*-based performed 45.83% and 47.91% respectively better than *Thumbnails*.

Another database (Database 2) was generated for this study with different shapes and geometric transformation. As previously discussed, the database generation was designed to represent a hard problem of image representation. In this case, the differences between classification results for *SOM*-based representation were slightly better than for *Thumbnails* (4.27% and 7.94%), for the case of original images of dimension 10 x 10 and 25 x 25, respectively.

As future work, we intend to experiment with a new database, where the geometric transformation steps are performed over a smoother range of values, not only over a constant value, as in Database 2. Thus we will get a better idea of the amount of deformation that each representation technique can stand. Moreover, other databases with more classes and examples as well as colour images could be considered.

Furthermore, we intend to explore other dimension sizes for the SOM map, running several experiments exploring new dimensions, varying the dimension values smoothly from lower than 10 to more than 25 (the values used in the current work).

References

1. Lew, M.S., Sebe, N., Djeraba, C., Jain, R.: Content-based multimedia information retrieval: State of the art and challenges. ACM Trans. Multimedia Comput. Commun. Appl. 2, 1–19 (2006)
2. Rakow, T.C., Neuhold, E.J., Löhr, M.: Multimedia Database Systems: The Notions and the Issues, Em Datenbanksysteme in Büro, Technik und Wissenschaft BTW, GI-Fachtagung, pp. 1–29 (1995)
3. Cohen, H.A.: Retrieval and Browsing Image Database Using Image Thumbnails. Journal of Visual Computing and Image Representation 8(2), 226–234 (1997)
4. Lehmann, T.M., Glda, M.O., Deselaersb, T., Keysersb, D., Schubertc, H., Spitzera, K., Neyb, H., Weinc, B.B.: Automatic categorization of medical images for content-based retrieval and data mining 29, 143–155 (2005)

5. Silva, L.A., Del-Moral-Hernandez, E., Moreno, R.A., Furuie, S.S.: Combining Wavelets Transform and Hu moments with Self-Organizing Maps for Medical Image Categorization. Journal of Electronic Imaging 1, 1–20 (2011)
6. Kohonen, T.: Self-Organizing Maps, 3rd extended edn., vol. 30. Springer, Heidelberg (2001)
7. Castelli, V., Bergman, L.: Image Databases- Search and Retrieval of Digital Imagery, 1st edn. John Wiley Professio., New York (2001)
8. Gupta, B., Gupta, S., Tiwari, A.K.: Face Detection Using Gabor Feature Extraction and Artificial Neural Network, ABES Engineering College, Ghaziaba (2010)
9. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: SOM Toolbox for Matlab 5. Technical Report A57, Helsinki University of Technology, Finland (2000)

Inverse Halftoning by Means of Self-Organizing Maps

Flavio Moreira da Costa, Sidnei Alves de Araújo, and Renato José Sassi

Industrial Engineering Post-Graduation Program, Nove de Julho University – UNINOVE
Av. Francisco Matarazzo, 612, Água Branca, São Paulo – SP, Brazil
flavio.costa@iduo.com.br, {saraujo,sassi}@uninove.br

Abstract. Halftoning is the process used to convert a grayscale image into another binary image such that the binary image appears to be similar to grayscale when observed from a certain distance. This process is useful for many printers which are binary in nature, once it allows the printer to deposit the ink as series of dots of constant darkness to print grayscale images. Inverse Halftoning is the reconstruction of grayscale image from its halftoned version. This process can be used in several applications when some image processing operation requires the original grayscale image and only its binary version is available. In this paper we present a method for inverse halftoning using Self-Organizing Maps that is able to reconstruct grayscale images from their halftoned versions generated by dispersed-dot ordered dithering and error diffusion algorithms. Obtained results demonstrate that the proposed method is a good alternative for the investigated purpose.

Keywords: Image processing, Self-organizing maps, Inverse halftoning, Ordered dithering, Error diffusion.

1 Introduction

Halftoning is the process that converts a grayscale image *G* in a corresponding binary image *B* such that *B* resembles *G* when viewed from a certain distance. Inverse Halftoning is the reverse process, that is, reconstruction of grayscale image from its halftoned version [1,2].

Halftoning algorithms are widely used for bilevel output devices, such as a monochrome display or a laser printer. For example, in most laser printers currently used, for reproducing grayscale images, it is necessary to generate a pattern of tiny dots, distributed according to the used halftoning method, to give the impression of an image with different gray levels [1,3].

Furthermore, some operations such as enlargement, reduction, rotation, adjusting of brightness and contrast, edge enhancement, noise suppression, texture detection and segmentation, among others, are made easier with grayscale images [3]. Thus, for various situations, it is necessary converting halftone images to grayscale images prior to any further processing.

A very simple inverse halftoning method is a low-pass filter such as Gaussian convolution. However, this process has the disadvantage of blurring edges in the image. Thus, a good halftoning process should be better than Gaussian filter.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 145–152.
DOI: 10.1007/978-3-642-35230-0_15

In the last two decades several authors have proposed different halftoning methods in the literature using techniques such as look-up-table (LUT) [4,5], decision trees [6], supervised artificial neural networks [7], statistics [3], combination of linear filters with stochastic models [8], sparse representation [9], among others.

In this paper we have proposed a method for inverse halftoning using Self-Organizing Maps (SOM) that is able to reconstruct grayscale images from their halftoned versions generated by dispersed-dot ordered dithering and error diffusion algorithms. Obtained results indicate that the proposed method is a good alternative for the investigated purpose.

2 Materials and Methods

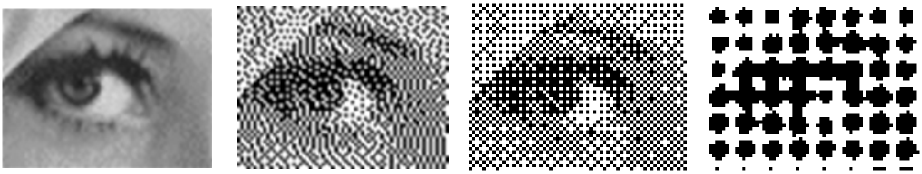
2.1 Halftoning and Inverse Halftoning Processes Definitions

There are several methods used for generating halftone images including error diffusion (Fig. 1b) and ordered dithering (Fig. 1c and Fig. 1d). The latter can generate halftone images with dispersed or clustered dots [8].

Given a grayscale image G , with real values between 0 and 1, halftoning can be mathematically formalized as a process that builds a halftoned binary image B from G , such that:

$$\overline{B}(i,j) \approx G(i,j) \tag{1}$$

where $\overline{B}(i,j)$ is the average of values around the pixel (i,j), considering a window *W*.



a. Original grayscale image

b. error diffusion

c. dispersed-dot ordered dithering

c. clustered-dot ordered dithering

Fig. 1. Examples of halftoned versions of a grayscale image

Inverse halftoning consists in reconstructing an estimated grayscale image $\hat{G}$ from B , such that:

$$\hat{G}(i,j) \approx G(i,j) \tag{2}$$

In halftoning process, regardless of used method, some information of original grayscale image is lost. Thus, by making the inverse halftoning, the resulting image is an approximation of original image. The problem is that not always the original image is known.

For this reason, given a halftone image B, we cannot recover the grayscale image $\hat{G}$ exactly the same as G. To express the degree of proximity between $\hat{G}$ and G, a similarity measure such as mean squared error (MSE), peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) have been widely used. PSNR and MSE are metrics that simply estimates error differences between original and reconstructed images, not being able to identify if calculated differences cause improvement or degradation in terms of quality, while SSIM is based on the quality perceived by human viewers [10].

2.2 Self-Organizing Maps

Self-Organizing Maps or Kohonen Neural Network is composed of two layers (input and output) and employs an algorithm for unsupervised learning to translate the similarities of the patterns presented in the input layer in relations of distance between the neurons that compose its output layer [11,12].

The Self-Organizing Maps works basically as follows: when a pattern is presented to the network input layer, a neuron of the output layer is chosen to represent this pattern by means of a competitive process.

During the training phase, the network increases the similarity of chosen neuron and their neighbors to the pattern presented in the input layer. Thus, it is constructed a topological map in which the output layer neurons that are topologically close respond similarly to input patterns with similar characteristics.

2.3 Experimental Setup

In the experiments described in this paper we test the proposed method, called SOM-IH, with halftone images generated by error diffusion and dispersed-dot ordered dithering methods. In both cases, we used small windows (3x3 and 5x5) and three sample images (Fig. 2) for training SOM. We glued the three sample images to compose a unique training image with size 2912x1296, containing different textures, brightness and contrast.

In the sequence, we generated from training image, two halftoned versions by applying error diffusion and dispersed-dot ordered dithering methods. From these two halftone images we extracted instances to compose the two training sets, for each halftoning method, taking into account windows of 3x3, 4×4 and 5x5. Each training set was composed by 9,000 instances that are randomly chosen. The input layers of build networks were composed by 9 ,16 and 25 neurons.

In the experiments we also evaluated the efficiency of SOM with respect to the number of neurons in the output layer. Thus, we set output layers with sizes 16×16 20×20 and 24×24.

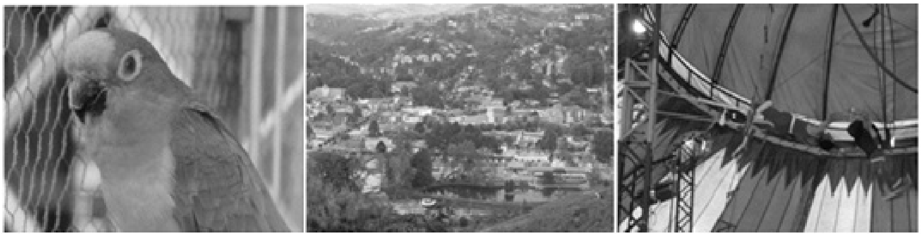


Fig. 2. Image used for training SOM

The algorithms employed in the experiments were implemented in C/C++ language using Proeikon library [13].

3 Inverse Halftoning Using Self-Organizing Maps

We refer to the method of inverse halftoning using SOM as SOM-IH. The proposed method consists of an operator ψ (from binary image to grayscale image) restricted to a window (W-operator) which is a function that maps B to $\hat{G}$. Thus, the SOM algorithm should play the role of the operator ψ where:

$$\psi : \{0,1\} \rightarrow [0,1] \quad (3)$$

Figure 3 illustrates the idea of reconstruction estimated grayscale image from a halftone binary image.

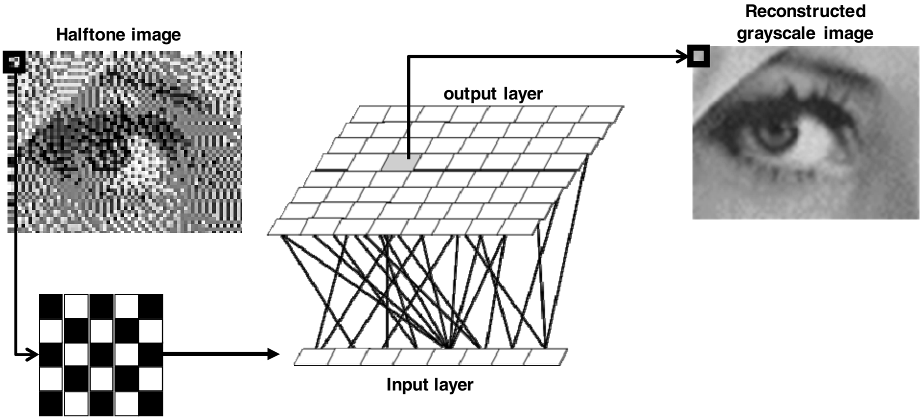


Fig. 3. Schematic diagram of SOM -IH

The reconstruction process of $\hat{G}$ from halftone image B can be described as follows: for each pixel of B, besides the pixel itself, it is also considered their neighbors within a window W as an input vector of SOM. Thus, by means of SOM algorithm, the pattern presented in the input layer (a binary vector) is mapped into a gray level according to activated neuron in the output layer.

4 Experimental Results

The results of tests with SOM-IH, considering windows (W) of size 3×3, 4×4 and 5×5, are summarized in Tables 1, 2 and 3, respectively. The quality of reconstructed grayscale images by SOM-IH were measured in terms of PSNR and SSIM.

Table 1. Obtained results with SOM-IH considering window of size 3×3, varying the number of neurons in the output layer

	Image	Output layer with 16×16 neurons		Output layer with 20×20 neurons		Output layer with 24×24 neurons	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Error diffusion	Airplane	23.268	0.236	23.851	0.252	23.896	0.254
	Goldhill	23.240	0.301	23.809	0.324	23.892	0.329
	Lenna	23.387	0.243	24.101	0.263	24.051	0.263
	Mandrill	20.042	0.382	20.656	0.428	20.425	0.413
	Peppers	22.809	0.209	23.326	0.227	23.360	0.228
	Average	**22.549**	**0.274**	**23.149**	**0.299**	**23.125**	**0.297**
Dispersed-dot ordered dithering	Airplane	21.718	0.204	21.800	0.204	21.650	0.203
	Goldhill	21.645	0.234	21.692	0.230	21.726	0.230
	Lenna	22.413	0.196	22.449	0.195	22.430	0.194
	Mandrill	19.049	0.311	19.316	0.319	19.175	0.309
	Peppers	21.734	0.165	21.765	0.165	21.838	0.165
	Average	**21.312**	**0.222**	**21.405**	**0.223**	**21.364**	**0.220**
General average		**21.931**	**0.248**	**22.277**	**0.261**	**22.244**	**0.259**

Using a microcomputer core 3, 2.13 GHz, with 4GB of RAM, the processing time spent by SOM-IH for reconstruction each grayscale image with size 512×512 varies from 2.5 to 7.0 seconds, depending on the architecture of SOM.

According to Tables 1 to 3, the best results of SOM-IH were obtained for haftone images generated with error diffusion method, W of size 3×3 and network with output layer of size 20×20 neurons. Considering W with size 3×3, the results of SOM-IH were lower for ordered dithering haftoning method, in most cases. However, considering W of size 4×4 and 5×5 the results for ordered dithering method was increased.

We also conducted experiments with larger W, for example 7×7, but it was evident that SOM-IH showed no improvement in the quality of reconstructed images. The same observation could be made in cases of networks with output layers less than 16×16 and greater than 24×24 neurons.

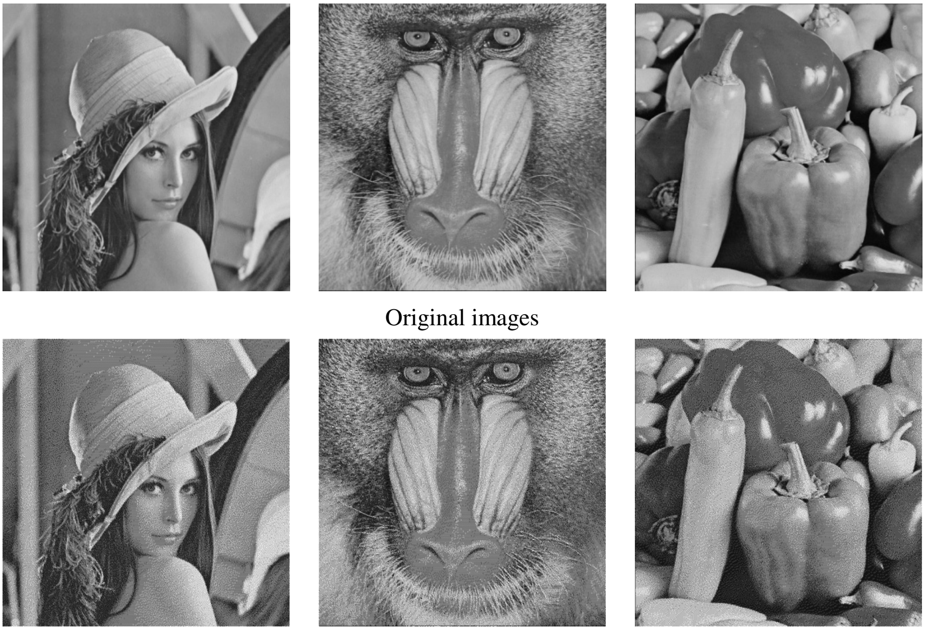
Table 2. Obtained results with SOM-IH considering window of size 4×4, varying the number of neurons in the output layer

	Image	Output layer with 16×16 neurons		Output layer with 20×20 neurons		Output layer with 24×24 neurons	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Error diffusion	Airplane	19.909	0.148	21.060	0.163	21.007	0.165
	Goldhill	20.937	0.171	21.523	0.182	21.605	0.190
	Lenna	20.816	0.151	21.443	0.161	21.423	0.168
	Mandrill	18.006	0.193	18.334	0.208	18.363	0.213
	Peppers	20.408	0.133	20.966	0.141	21.096	0.150
	Average	**20.015**	**0.159**	**20.665**	**0.171**	**20.699**	**0.177**
Dispersed-dot ordered dithering	Airplane	22.642	0.196	22.495	0.203	22.331	0.197
	Goldhill	22.876	0.245	23.380	0.248	23.206	0.242
	Lenna	23.131	0.209	23.754	0.216	23.137	0.207
	Mandrill	19.254	0.271	19.573	0.277	19.364	0.275
	Peppers	22.410	0.182	22.949	0.191	22.661	0.184
	Average	**22.063**	**0.220**	**22.430**	**0.227**	**22.140**	**0.221**
General average		**21.039**	**0.190**	**21.548**	**0.199**	**21.419**	**0.199**

Table 3. Obtained results with SOM-IH considering window of size 5×5, varying the number of neurons in the output layer

	Image	Output layer with 16×16 neurons		Output layer with 20×20 neurons		Output layer with 24×24 neurons	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Error diffusion	Airplane	20.547	0.136	20.337	0.139	20.806	0.149
	Goldhill	20.353	0.132	20.290	0.135	20.828	0.150
	Lenna	20.140	0.127	20.291	0.129	20.709	0.145
	Mandrill	17.591	0.130	17.579	0.134	17.802	0.144
	Peppers	19.802	0.118	19.748	0.117	20.188	0.131
	Average	**19.687**	**0.129**	**19.649**	**0.131**	**20.067**	**0.144**
Dispersed-dot ordered dithering	Airplane	23.047	0.201	23.246	0.206	24.046	0.233
	Goldhill	23.525	0.245	23.596	0.243	23.703	0.251
	Lenna	24.068	0.216	24.285	0.221	24.404	0.234
	Mandrill	19.508	0.228	19.538	0.246	19.527	0.256
	Peppers	23.247	0.195	23.708	0.201	23.727	0.213
	Average	**22.679**	**0.217**	**22.875**	**0.223**	**23.081**	**0.237**
General average		**21.183**	**0.173**	**21.262**	**0.177**	**21.574**	**0.191**

Figure 4 shows examples of results of SOM-IH considering the best parameters.



Original images

Reconstructed images

Fig. 4. Examples of SOM-IH results

5 Conclusions

In this paper we presented a method based on Self-Organizing Maps to reconstruct grayscale images from their halftoned versions generated by ordered dithering and error diffusion algorithms. In order to evaluate the proposed method we conduct a set of experiments and used PSNR and SSIM to measure the quality of reconstructed grayscale images.

From the obtained results we can conclude that SOM with 9 neurons in the input layer and 400 neurons in the output layer, disposed in a matrix of size 20×20, represents a good alternative for the investigated purpose. However, the SSIM indexes presented in Tables 1 to 3 shows that the qualities of reconstructed images need to be increased.

We believe that the results of the SOM-IH can be improved by making a refinement in the training, for example, using Learning Vector Quantization algorithm. Another alternative is to select only the most important attributes for composing the training sets. This could be done by means of Genetic Algorithms or Rough Sets Theory. Currently we are investigating both ideas.

References

1. Gonzalez, R.C., Wintz, P.: Digital Image Processing. Addison-Wesley, Massachusetts (1992)
2. Castleman, K.R.: Digital Image Processing. Prentice-Hall, New Jersey (1996)

3. Karni, Z., Freedman, D., Shaked, D.: Fast Inverse Halftoning. In: 31st International Congress on Imaging Science (ICIS 2010), Beijing, China (2010)
4. Mese, M., Vaidyanathan, P.P.: Look-Up Table (LUT) Method for Inverse Halftoning. IEEE Transactions on Image Processing 10(10), 1566–1578 (2001)
5. Mese, M., Vaidyanathan, P.P.: Tree-Structured Method for LUT Inverse Halftoning and for Image Halftoning. IEEE Transactions on Image Processing 11(6), 644–655 (2002)
6. Kim, H.Y., Queiroz, R.L.: Inverse Halftoning by Decision Tree Learning. In: IEEE International Conference on Image Processing, Barcelona, vol. 2, pp. 913–916 (2003)
7. Huang, W.B., Chang, W.C., Lu, Y.W., Su, A.W.Y., Kuo, Y.H.: Halftone/Contone Conversion Using Neural Networks. IEEE Transactions on Image Processing 5, 3547–3550 (2004)
8. Freitas, P., Farias, M., Araujo, A.: Fast Inverse Halftoning Algorithm for Ordered Dithered Images. In: 24th SIBGRAPI - Conference on Graphics, Patterns and Images, pp. 250–257 (2011)
9. Son, C.H.: Inverse halftoning based on sparse representation. Optics Letters 37(12), 2352–2354 (2012)
10. Wang, Z., Lu, L., Bovik, A.C.: Video Quality Assessment Based on Structural Distortion Measurement. Signal Processing: Image Comm. 19, 121–132 (2004)
11. Kohonen, T.: The self-organizing map. Proceedings of the Institute of Electrical and Electronics Engineers 78, 1464–1480 (1990)
12. Haykin, S.: Neural Networks – A Comprehensive Foundation, 2nd edn. Prentice Hall, New Jersey (1999)
13. Kim, H.Y.: ProEikon – Library for Image Processing and Computer Vision, `http://www.lps.usp.br/~hae/software` (access at February 2011)

Restoration Model with Inference Capability of Self-Organizing Maps

Michiharu Maeda

Fukuoka Institute of Technology
3-30-1 Wajiro-higashi, Higashi-ku, Fukuoka, Japan
maeda@fit.ac.jp

Abstract. This paper presents a restoration model with inference capability of self-organizing maps. Self-organizing maps have been studied principally for the ordering process and the convergence phase of weight vectors. As a novel approach of self-organizing maps, a restoration model for a defective image is proposed. The model creates a map containing one unit for each pixel. Utilizing pixel values as input, the inference for lost pixels is conducted by self-organizing maps. The inference of an original image proceeds appropriately since any pixel is influenced by neighboring pixels corresponding to the neighboring setting. Consequentially, images with high quality are constituted by restoring lost pixels. Experimental results are presented in order to show that our approach is effective in quality for restoration of lost pixels.

Keywords: self-organizing maps, inference, restoration, lost pixel.

1 Introduction

Self-organizing maps realize the network with the local and topological ordering by utilizing the mechanism of the lateral inhibition among neurons. Neighboring neurons usually respond to neighboring inputs [1, 2]. For the localized inputs obviously, the outputs react locally. Huge amounts of information are locally represented and their expressions form a configuration with topological ordering. As an application of self-organizing maps, for example, there are the combinatorial optimization problem, pattern recognition, vector quantization, and clustering [3]. These are useful when there exists redundancy among input data. If there is no redundancy, it is difficult to find specific patterns or features in the data. Although a number of self-organizing maps exist, they differ with respect to the field of application. For self-organizing maps, the ordering and the convergence of weight vectors have been mainly argued [4]. The former is a topic on the formation of topology preserving map, and outputs are constructed in proportion to input characteristics [5, 6]. For instance, there is the traveling salesman problem as an application of feature maps, which is possible to obtain fine results by adopting the elastic-ring method with many weights compared to inputs [7, 8]. The latter is an issue on the approximation of pattern vectors, and the model

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 153–162.
DOI: 10.1007/978-3-642-35230-0_16

expresses enormous information of inputs to a few weights. It is especially an important problem for the convergence of weight vectors, and asymptotic distributions and quantitative properties for weight vectors have been mainly discussed when self-organizing maps are applied to vector quantization [9–12]. In the meantime, the proposed model is inspired by image restoration using self-organizing maps [13, 14] which infers an original image from a degraded image including random-valued impulse noise. For image restoration, the smoothing methods, such as the moving average filter and the median filter, have been well known as a plain and useful approach [15]. From the standpoint of distinct ground, the inference of original image has been conducted by the model of Markov random field formulated statistically, based on the concept that any pixel is affected by neighboring pixels [16, 17].

In this study, a restoration model with inference capability of self-organizing maps is described. Our model forms a map in which one element corresponds to each pixel. The inference for lost pixels is conducted by self-organizing maps using pixel values as input. As any pixel is influenced by neighboring pixels corresponding to neighboring setting, the inference of an original image is appropriately promoted. Consequentially, images with high quality are constituted by restoring lost pixels. Experimental results are presented in order to show that our approach is effective in quality for restoration of lost pixels.

2 Self-Organizing Maps

For self-organizing maps, Kohonen's algorithm exists and is known as a popular and utility learning. In this algorithm, the updating of weights is modified to involve neighboring relations in the output array. The algorithm is applied to the structure as shown in Fig. 1. In vector space R^n, input $\boldsymbol{x}$ which is generated on probability density function $p(\boldsymbol{x})$ is defined. Input $\boldsymbol{x}$ has the components x_1 to x_n. Output unit y_i is generally arranged in an array of one- or two-dimensional maps and is completely connected to inputs via w_{ij}.

Let $\boldsymbol{x}(t)$ be an input vector at step t and let $\boldsymbol{w}_i(0)$ be weight vectors at initial values in R^n space. For given input vector $\boldsymbol{x}(t)$, we calculate the distance between $\boldsymbol{x}(t)$ and $\boldsymbol{w}_i(t)$, and select the weight vector as winner c minimizing the distance. The process is written as follows:

$$c = \arg\min_i\{\|\boldsymbol{x} - \boldsymbol{w}_i\|\}, \tag{1}$$

where arg$(\cdot)$ gives the index c of the winner.

With the use of winner c, weight vector $\boldsymbol{w}_i(t)$ is updated as follows:

$$\Delta\boldsymbol{w}_i = \begin{cases} \alpha(t)\,(\boldsymbol{x} - \boldsymbol{w}_i) & (i \in N_c(t)), \\ \boldsymbol{0} & (\text{otherwise}), \end{cases} \tag{2}$$

where $\alpha(t)$ is the learning rate and is a decreasing function of time $(0 < \alpha(t) < 1)$. $N_c(t)$ has a set of indexes of topological neighborhoods for winner c at step t.

The adaptive learning algorithm evaluates unknown probability density function $p(\boldsymbol{x})$. Then weight vectors represent centroids of each clustering set.

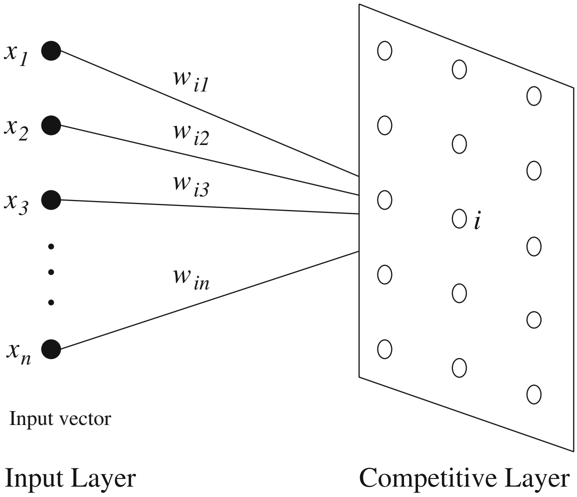


Fig. 1. Structure for self-organizing maps

Generally, with respect to the above learning for evaluating results, cost function H exists as follows:

$$H = \sum_{i=1}^{k} \int_{S_i} d(\boldsymbol{x}, \boldsymbol{w}_i) p(\boldsymbol{x}) d\boldsymbol{x}, \tag{3}$$

where k is the number of clustering set represented by partition space S_i and $d(\boldsymbol{x},\ \boldsymbol{w}_i)$ is the square error of the Euclidean distance between input vector $\boldsymbol{x}$ $= (x_1, x_2, \cdots, x_n)$ and weight vector $\boldsymbol{w}_i = (w_{1i},\ w_{2i},\ \cdots,\ w_{ni})$, i.e., $d(\boldsymbol{x},\ \boldsymbol{w}_i)$ $= \|\boldsymbol{x}-\boldsymbol{w}_i\|^2$.

For discrete data, for weight vectors which result from learning, a distortion error is calculated. Let D_i be the i-th partition error as the following equation.

$$D_i = \sum_{\boldsymbol{x} \in S_i} d(\boldsymbol{x}, \boldsymbol{w}_i). \tag{4}$$

Mean square error E is given continuously as follows:

$$E = \sum_{i=1}^{k} D_i. \tag{5}$$

Eq. (3) corresponds to Eq. (5) [18], as the sequence of input vectors $\boldsymbol{x}(t)$ becomes stationary and ergodic [19]. Here the dimension of input vector and the total number of input vectors are omitted in this study.

As described above, weights are adapted by self-organizing maps while they are affected with neighboring relations. Thus it is possible to form topology preserving map and to approximate pattern vectors. Furthermore cost function exists and is able to evaluate the accuracy for weights after learning.

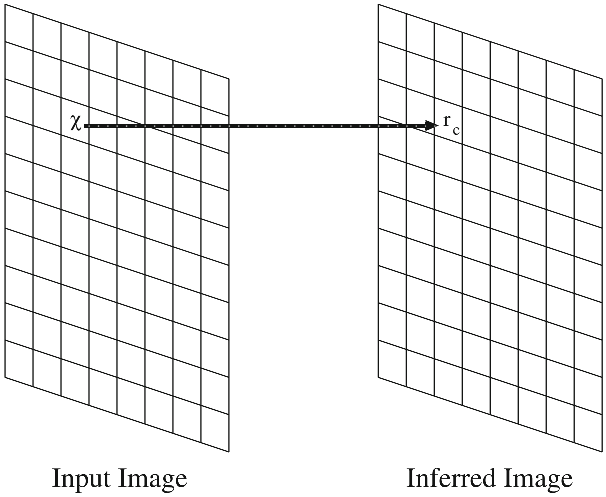


Fig. 2. Correspondence between input image and inferred image

3 Restoration of Lost Pixels

When self-organizing maps are adapted to the traveling salesman problem, many weights compared to inputs are used. By disposing an array of one-dimensional map for output units, fine solutions based on the position of weights after learning have been obtained approximately. In the meantime, when self-organizing maps apply to vector quantization, a few weights compared to inputs are utilized for the purpose of representing huge amounts of information, and a number of discussions have been made on asymptotic distributions and quantitative properties for weight vectors.

In this section, a learning algorithm of self-organizing maps for restoration of lost pixels is presented with the same number both of inputs and weights for inferring an original image from a yielded image. In order to restore a defective image, the proposed algorithm is inspired by image restoration using self-organizing maps [13, 14] which infers an original image from a degraded image including random-valued impulse noise. The purpose of this study is to infer the original image by restoring lost pixels. Here, input χ as the yielded image and weight r_i as the inferred image are defined. A map forms that one element reacts for each pixel, and image inference is executed by self-organizing maps using pixel values as input. We assume that the positions of lost pixels are already known.

To begin with, the value of r_i is randomly distributed near the central value of gray scale as initial value. Next, input image with $l \times m$ size is given with the positions of lost pixels. Input χ as the scalar value of gray scale is arbitrarily selected, except for lost pixels, and let r_c be a winner of the inferred image corresponding to χ. As shown in Fig. 2, both positions χ and r_c agree under

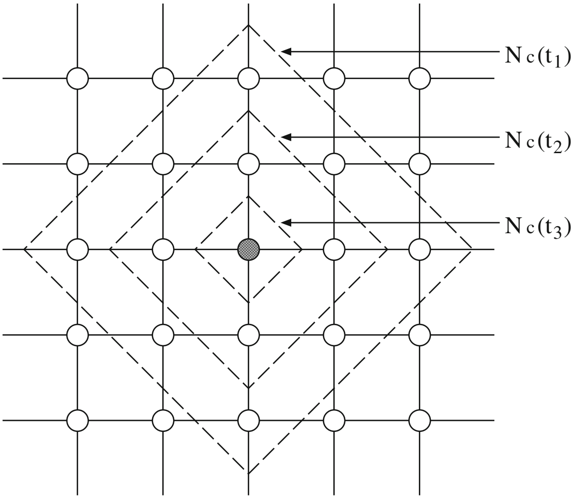


Fig. 3. Distribution of topological neighborhoods

the input image and the inferred image. Therefore, inferred image r_i is updated as follows:

$$\Delta r_i = \begin{cases} \alpha(t)(\chi - r_i) & (i \in N_c(t)), \\ 0 & (\text{otherwise}). \end{cases} \tag{6}$$

where the notations are changed from Eq. 2, because we treat the scalar values of gray scale.

Figure 3 shows an example of the arrangement of topological neighborhoods. The circle signifies the weight and the line which connects the circles denotes the topological neighborhood. In this figure, the black circle expresses the weight of winner c. As the set of topological neighborhoods changes $N_c(t_1)$, $N_c(t_2)$, and $N_c(t_3)$ when the time varies t_1, t_2, and t_3, respectively, it is shown that the number of topological neighborhoods decreases with time. By obtaining information of the neighboring pixels, it is possible to complement lost information about pixels.

Pixel restoration by self-organizing maps (PRSOM) algorithm is presented as follows.

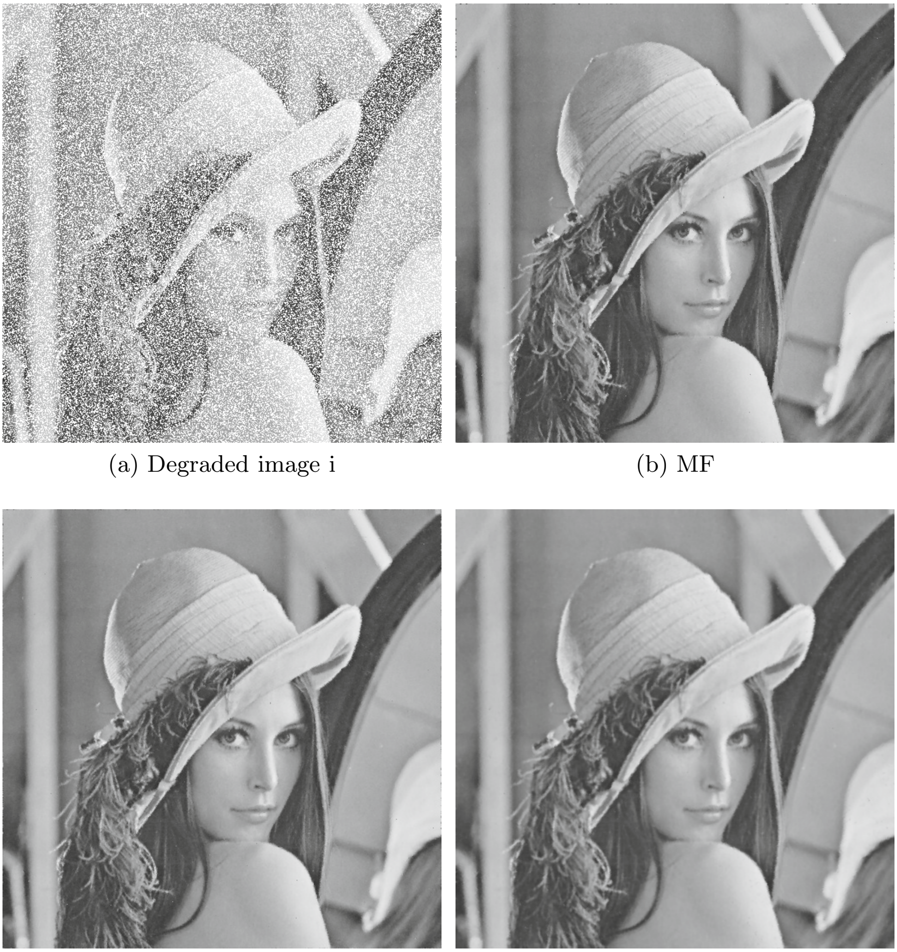
[*PRSOM algorithm*]

```
Step 1  Initialization:
        Give initial weights $\{r_1(0), r_2(0), \cdots, r_{lm}(0)\}$, input image $\{\chi_1, \chi_2, \cdots, \chi_{lm}\}$ including lost pixels, and maximum iteration $T_{max}$. Set $t \leftarrow 0$.
Step 2  Learning:
        (2.1) Choose input $\chi$ as gray scale at random, except for lost pixels
        (2.2) Select $r_c$ corresponding to input $\chi$.
        (2.3) Update $r_c$ and its neighborhoods according to Eq. (6).
        (2.4) Set $t \leftarrow t + 1$.
```

Step 3 Construction of restored image:
If the lost pixel is χ_i, the restored pixel is r_i, otherwise it is χ_i
Step 4 Condition:
If $t = T_{max}$, then terminate, otherwise go to Step 2.

In this study, a peak signal to noise ratio (PSNR) P is used as the quality measure after learning for restoration of lost pixels. PSNR P is presented as follows [20]:

$$P = 10 \log_{10}(\sigma/E) \quad [\mathrm{dB}] \tag{7}$$



(a) Degraded image i

(b) MF

(c) MA

(d) PRSOM

Fig. 4. Degraded image i with 512×512 size and 256 gray-scale, and examples of results for the median filter (MF), the moving average (MA), and the proposed approach (PRSOM)

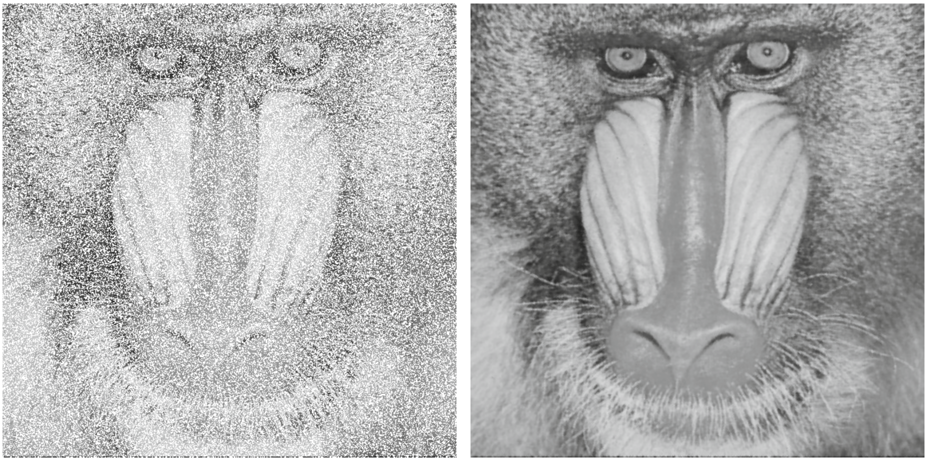
where σ and E are the square of gray-scale length, i.e., $\sigma = (Q-1)^2$ as a gray scale Q, and mean square error between the original image and the inferred image, respectively.

4 Numerical Experiments

In the numerical experiments, image restoration is performed to infer the original image with the size 512×512 and gray scale 256. The defective image contains 40% lost in comparison with the original image as shown in Fig. 4 (a). We assume that the positions of lost pixels are already known. Initial weights are randomly distributed near the central value of gray scale Q. Parameters are chosen as follows: $l = 512$, $m = 512$, $Q = 256$, $M = 100$, $T_{max} = Mlm$, and $N(t) = N_0 - \lfloor N_0 t / T_{max} \rfloor$.

For image restoration, Fig. 4 (b), (c), and (d) show examples of results for the median filter (MF), the moving average (MA), and the proposed approach (PRSOM), respectively. Here the smoothing methods of MF and MA have 3×3 mask. The smoothing methods restore using remained pixels and values resulted by these insert only in the lost pixels. As a result, MF, MA, and PRSOM are almost same in appearance as shown in the figures. According to the technique given in this study, the defective image is restorable for PRSOM.

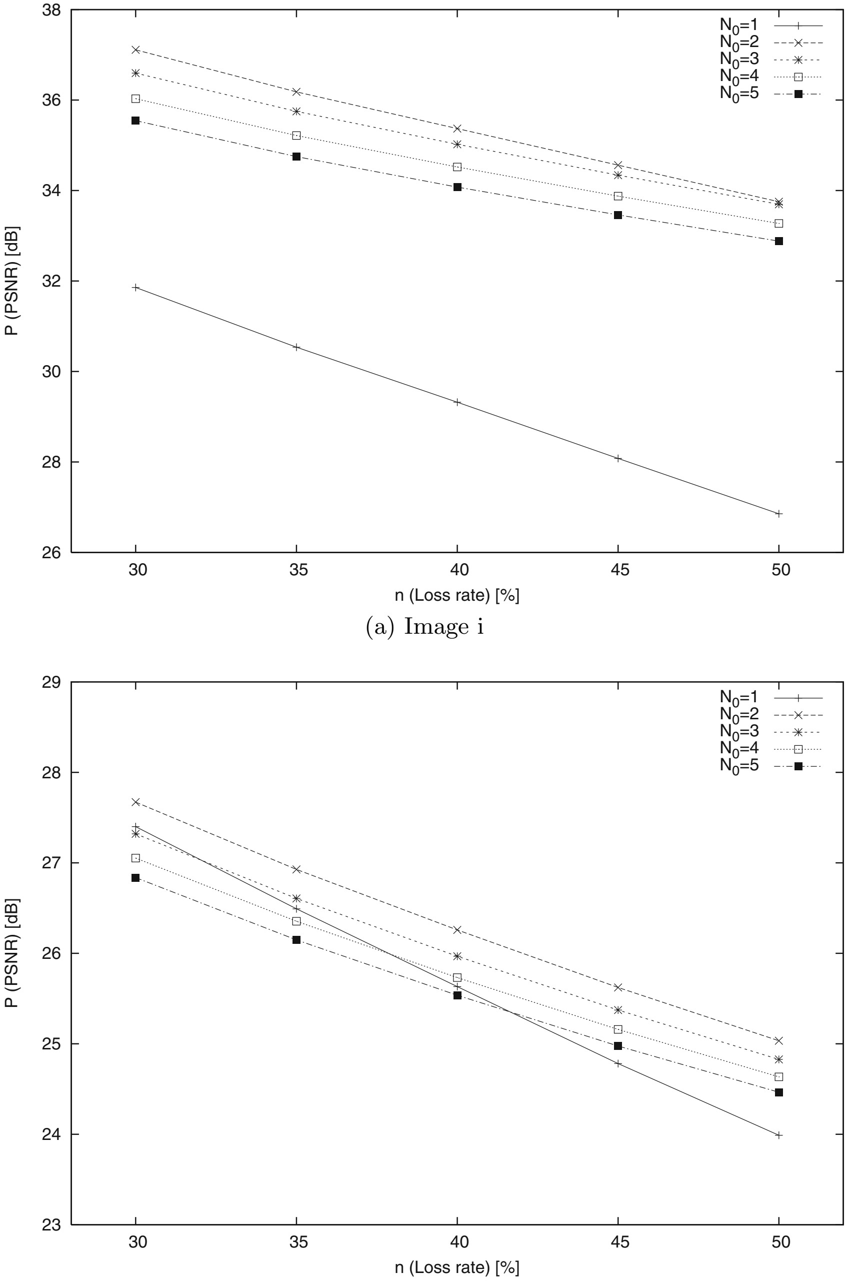
As an example of another image, Fig. 5 (a) shows the defective image. As well as the above-mentioned image, the defective image contains 40% lost compared to the original image. The condition of the computation is equal to that of the earlier description. According to the present algorithm, a result of PRSOM is shown in Fig. 5 (b). The initial neighborhood is $N_0 = 2$. It is proven that the defective image can be also restored in this case.



(a) Degraded image ii

(b) PRSOM

Fig. 5. Degraded image ii with 512×512 size and 256 gray-scale and example of result for the proposed approach (PRSOM)



(a) Image i

(b) Image ii

Fig. 6. PSNR and loss rate for each initial neighborhood

Figure 6 shows the effect of loss rate on accuracy in PSNR P for each of initial neighborhood $N_0 = 1, 2, 3, 4, 5$ for images i and ii. In this case, P yields the maximum when $N_0 = 2$ for both images i and ii. Figure 4 (d) was restored by this value.

Table 1 summarizes PSNR for results of the proposed approach (PRSOM) compared to the median filter (MF) and the moving average filter (MA). It is proven that PRSOM excel MF and MA for both images i and ii.

Table 1. PSNR for results of MF, MA and PRSOM. (Unit: dB)

Image	i					ii				
Loss rate	30%	35%	40%	45%	50%	30%	35%	40%	45%	50%
MF	35.7	34.7	33.7	32.7	31.8	26.5	25.6	24.9	24.1	23.5
MA	36.7	35.8	35.0	34.0	33.1	27.2	26.5	25.8	25.1	24.5
PRSOM	37.1	36.2	35.4	34.6	33.8	27.7	26.9	26.3	25.6	25.0

5 Conclusions

In this study, a restoration model with inference capability of self-organizing maps has been described and its validity has been shown through numerical experiments. Our model formed a map in which one element corresponds to each pixel. The inference of lost pixels was conducted by self-organizing maps using pixel values as input. As any pixel was influenced by neighboring pixels corresponding to neighboring setting, the inference of an original image was appropriately promoted. As a result, images with high quality were constituted by restoring lost pixels. Finally, for the future works, we will study more effective techniques of our algorithms.

References

1. Grossberg, S.: Adaptive Pattern Classification and Universal Recoding: I. Parallel Development and Coding of Neural Feature Detectors. Biol. Cybern. 23, 121–134 (1976)
2. Willshaw, D.J., Malsburg, C.: How Patterned Neural Connections Can Be Set up by Self-Organization. Proc. R. Soc. Lond. B 194, 431–445 (1976)
3. Hertz, J., Krogh, A., Palmer, R.G.: Introduction to the Theory of Neural Computation. Addison-Wesley (1991)
4. Kohonen, T.: Self-Organizing Maps. Springer, Berlin (1995)
5. Villmann, T., Herrmann, M., Martinetz, T.M.: Topology Preservation in Self-Organizing Feature Maps: Exact Definition and Measurement. IEEE Trans. Neural Networks 8, 256–266 (1997)
6. Maeda, M., Miyajima, H., Shigei, N.: Parallel Learning Model and Topological Measurement for Self-Organizing Maps. Journal of Advanced Computational Intelligence and Intelligent Informatics 11, 327–334 (2007)

7. Durbin, R., Willshaw, D.: An Analogue Approach to the Traveling Salesman Problem Using an Elastic Net Method. Nature 326, 689–691 (1987)
8. Angéniol, B., Vaubois, G., Texier, J.-Y.: Self-Organizing Feature Maps and the Traveling Salesman Problem. Neural Networks 1, 289–293 (1988)
9. Ritter, H., Schulten, K.: On the Stationary State of Kohonen's Self-Organizing Sensory Mapping. Biol. Cybern. 54, 99–106 (1986)
10. Ritter, H., Schulten, K.: Convergence Properties of Kohonen's Topology Conserving Maps, Fluctuations, Stability, and Dimension Selection. Biol. Cybern. 60, 59–71 (1988)
11. Maeda, M., Miyajima, H.: Competitive Learning Methods with Refractory and Creative Approaches. IEICE Trans. Fundamentals E82–A, 1825–1833 (1999)
12. Maeda, M., Shigei, N., Miyajima, H.: Adaptive Vector Quantization with Creation and Reduction Grounded in the Equinumber Principle. Journal of Advanced Computational Intelligence and Intelligent Informatics 9, 599–606 (2005)
13. Maeda, M.: A Relaxation Algorithm Influenced by Self-Organizing Maps. In: Kaynak, O., Alpaydın, E., Oja, E., Xu, L. (eds.) ICANN 2003 and ICONIP 2003. LNCS, vol. 2714, pp. 546–553. Springer, Heidelberg (2003)
14. Maeda, M., Shigei, N., Miyajima, H.: Learning Model in Relaxation Algorithm Influenced by Self-Organizing Maps for Image Restoration. IEE J. Trans. Electrical and Electronic Engineering 3, 404–412 (2008)
15. Gonzalez, R.C., Woods, R.E.: Digital Image Processing. Prentice Hall (2002)
16. Geman, S., Geman, D.: Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. IEEE Trans. Pattern Anal. Mach. Intel. 6, 721–741 (1984)
17. Maeda, M., Miyajima, H.: State Sharing Methods in Statistical Fluctuation for Image Restoration. IEICE Trans. Fundamentals E87-A, 2347–2354 (2004)
18. Reichl, L.E.: A Modern Course in Statistical Physics. University of Texas Press (1980)
19. Imai, H.: Information Theory, Shokodo (1984)
20. Gersho, A., Gray, R.M.: Vector Quantization and Signal Compression. Kluwer Academic Publishers (1992)

Improvements in the Visualization of Segmented Areas of Patterns of Dynamic Laser Speckle

Lucía I. Passoni, Ana Lucía Dai Pra, Adriana Scandurra, Gustavo Meschino,
Christian Weber, Marcelo Guzmán, Héctor Rabal, and Marcelo Trivi

Facultad de Ingeniería. Universidad Nacional de Mar del Plata, Mar del Plata, Argentina
Centro de Investigaciones Ópticas CIC-CONICET, La Plata, Argentina
{lpassoni,daipra,scandu,gmeschin,marcelo.guzman}@fi.mdp.edu.ar,
cweber@ciop.unlp.edu.ar, hrabal@ing.unlp.edu.ar,
marcelociop@yahoo.com.ar

Abstract. This paper proposes a method to visualize different regions into image of *biospeckle* patterns using Self-Organizing Maps. Images are obtained from sequences of laser speckle images of biological specimens. The dynamic speckle is a phenomenon that occurs when a beam of coherent light illuminates a sample in which there is some type of activity, not visible, which results in a variable pattern over time. Self-Organizing Maps have shown an efficient behavior for the identification of regions according to the activity of the phenomenon involved. In this paper we show results obtained in the segmentation of regions in corn seeds, particularly the detection of the floury zone.

Keywords: Dynamic Laser Speckle, Biospeckle, Self-Organizing Maps, Corn seed.

1 Introduction

The laser dynamic speckle is an optical phenomenon produced when a laser light is reflected from an illuminated surface undergoing some kind of activity. The activity is evident when the sample changes its properties due to diverse physical reasons. This behavior can be observed in biological process such as seeds viability [1], bacteria activity [2], fruits bruising [3], and non-biological processes such as drying of paints [4], and corrosion [5].

Dynamic laser speckle patterns have been used to assess issues of interest in different fields, like biology (seed analysis, animal sperm motility), medicine (capillary blood flow), industry (discovering bruising in fruits, painting drying, monitoring of ice cream melting, yeast bread, gels), etc. Different descriptors have been proposed to evaluate the activity; they are characterized by their computational cost and aptitude to detect activity in particular cases [6]. A novel Descriptor based in Fuzzy Granularity (FGD) has been proposed by Dai Pra *et al.* [7]. It exhibits low computational cost and good performance when applied to discover different dynamic phenomena [4].

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 163–171.
DOI: 10.1007/978-3-642-35230-0_17

Over the years, corn was finding different uses depending on the physico-chemical composition that defines the type of grain. The quality of maize grain is associated with both physical composition, which determines the texture and hardness, as to their chemical composition, which defines the nutritional and technological properties. The kernel of corn consists of four main parts, where the endosperm is 80-85%, 10-12% embryo, the pericarp 5-6% and 2-3 percent aleurone. The chemical composition of the endosperm is what sets different grain shapes and physical characteristics, which enable the commercial rates [8, 9].

There has been an explosion of interest among seed corn buyers about the differences in the type of starch found in hybrids. What these discussions are referring to is the amount of floury (also called *soft* or *dent*) endosperm versus vitreous (also called *hard* or *flinty*) endosperm [10].

The method commonly used to assess the proportion of hard endosperm is the flotation test [11]. An aqueous solution of sodium nitrate is used, achieving a specific gravity of 1.25 to water kept at a temperature of 35 °C. This method allows comparing the density of various batches of corn kernels; it is based on the principle that the hard grains are of greater density and therefore such grains float in lower proportion than the grains of lower density in the solution of sodium nitrate.

Quantifying the floating grains does not allow determine the amount of endosperm starchy endosperm and vitreous grains presenting a given sample. That is why this paper entered the optical field to determine the possibility of using the method of speckle in such a disquisition.

Computational Intelligence methodologies have been previously used for processing speckle image sequences. The design of decision models with Artificial Neural Networks, Fuzzy Granular Computation and Genetic Algorithms is addressed in [12]. Self-Organizing Maps (SOM) were used to characterize a chemotaxis assay in [13]; where regions were neatly differentiated according to the bacterial motility within the sample. In [14] SOMs were proposed as clustering methods, when the sensitivity of the activity measurement of dynamic speckle images needs to be improved, by using the mean energy of the wavelet coefficients of the intensity series as a set of descriptors .

In this work we propose the use of time domain descriptors together with SOMs to discriminate the speckle dynamic activity of the endosperm embryo. The aim is to provide a tool to be used jointly with digital image processing methodologies to determine areas of the corresponding endosperm fractions. The activity of the endosperm is focused on its two majority parties (floury and vitreous endosperm), and the issue of successfully automating the identification of these areas would be of potential importance for trade and industrialization.

2 Methodology

In this section we propose a SOM-based model that uses several descriptors (features) of laser speckle patterns to identify areas of corn seeds images.

2.1 Equipment Setup, Signal Acquirement, and Feature Extraction

Assays were performed in the laboratory belonging to the Center of Optical Research (CONICET-CIC CIOp). They were performed on maize grains known at industry as *flint,* on 10 specimens from the same sample. To carry out the measures, every grain of corn was wet for 12 hours, then cut them lengthwise and cut surface. They were illuminated with an expanded laser attenuated He-Ne (10 mW) at room temperature (approx. 20 °C). Using a CCD camera and a computer with digital image processor, a sequence of 300 images for each sample tested was recorded and filmed at 8 bits resolution and 400 x 400 pixels size, with sampling frequency of approximately 1 Hz. Fig. 1 shows a schematic of the experimental unit used.

Intensity from each image pixel of the image stack was converted into a time series (Time History Speckle Patterns, THSP) to be processed by computing different descriptors, as shown in Fig. 2. So, the feature extraction was performed over the time series of intensity level in a pixel wise basis, computing numerical descriptors for every pixel location. As stated by Trivi [16] the speckle is a stochastic effect and time series of laser speckle patterns are the measurable evidence of this stochastic process.

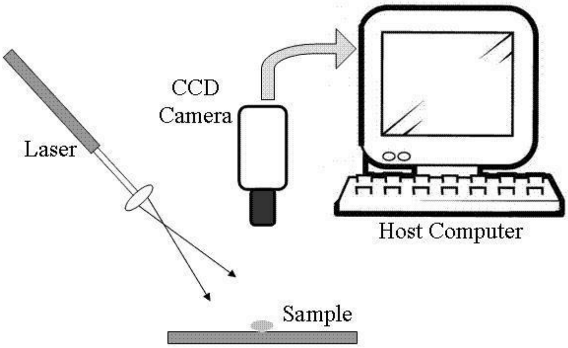


Fig. 1. Optical setup

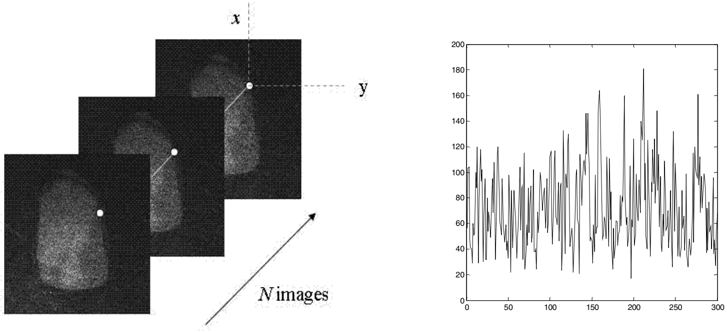


Fig. 2. Left: Image stack, where a time series of laser speckle pattern of the *(x,y)* pixel in N images is pointed. Right: Time series of the intensity variation as the dynamic speckle pattern of a *(x,y)* pixel.

There are many descriptors that have been developed to characterize *biospeckle* [1]. We propose the use of three descriptors that deal with the time domain, in order to reduce computational cost, compared with those processed signals in the frequency or time-frequency domains. They are the Average of Subtraction of Consecutive Images, the Dynamic Range Descriptor and the Fuzzy Granular Descriptor.

Subtraction Average of consecutive pixel intensities

One of the simplest descriptor is the Subtraction Average (SA) of two consecutive elements of the time speckle pattern [6].

$$SA = \sum_{k=1}^{N-1} \left| I_k(x,y) - I_{k+1}(x,y) \right| \Big/ N - 1, \qquad (1)$$

where (x, y) is the image pixel location and N is the amount of images stacked.

Dynamic Range Descriptor

Dynamic Range descriptor was computed as the difference between the maximum and the minimum value of the intensity in each evaluated time series. The potential of this feature lies in its speed and ability to discriminate regions of coarse different activity [8].

$$DR = \max_{k=1,N}\left\{ I\ (x,y)_k \right\} - \min_{k=1,N}\left\{ I\ (x,y)_k \right\} \qquad (2)$$

Fuzzy Granular Descriptor

The Fuzzy Granular algorithm is based on granular computing. It can be applied to both stationary and non-stationary cases, allowing monitoring the phenomenon in almost real time. According to the histogram of the image stack, different types of granules are identified; they are detected and counted, giving a descriptor that weights the series changes through the number of granules in a fixed time lapse [7].

The fuzzy sets theory, making reference to vague and overlapped concepts, allows defining granules with this property. To generate information granules several fuzzy sets are defined into the intensity values domain of the THSP. For intensity values $I(x, y)$, a fuzzy set is defined by a membership function $\mu(I(x, y))$ that takes gradual values in the real interval [0,1] (Eq. 3).

Trapezoidal functions μ_{dark} , μ_{medium} and μ_{light} with media overlapping are adopted, where:

$$\mu_c(I(x,y)) \in [0,1], \text{ with } c \in \{dark, medium, light\}. \qquad (3)$$

Each granule of $I(x, y)$ signal is defined as a continuous sequence of elements belonging to the same intensity concept. The Fuzzy Granular Descriptor is the result of applying Eq. 4 to each $I(x, y)$ signal.

$$Q_N = \left(\sum_{c=1}^{3} suc_{k,c} \left[\mu_c \left(I_k(x,y) \right) \right] \right) / N, \quad k = 1,2...,N$$
$$suc_{k,c} = \begin{cases} 1 & \text{if } \mu_c \left(I_{k-1}(x,y) \right) \neq 0 \wedge \mu_c \left(I_k(x,y) \right) = 0 \\ 0 & \text{else} \end{cases} \tag{4}$$

2.2 Pseudo-coloring by Self-Organizing Maps

The Self-Organizing Map proposed by Kohonen [17] is a popular non supervised neural network model. A SOM quantizes the data space of training data and simultaneously it performs a topology-preserving projection of the data space onto a regular neuron (or cell) grid.

SOM structure is usually a regular 2-dimensional grid of neurons, though they can be arranged in 1-dimensional (line) or 3-dimensional (space). Considering D-dimensional input data, each neuron *i*, is connected to the inputs by D weights. From another point of view, these weights can be seen as D-dimensional reference vectors contained into the cells. The set of reference vectors is called the SOM codebook. Neurons of the map are related to adjacent neurons only by a neighborhood functional definition. There are no weights connecting neurons each other.

During each training step, one sample vector from the input data set is taken randomly and a similarity measure is computed between the input vector and all the codebook vectors. The cell whose weight vector has the greatest similarity with the input sample is selected as the Best-Matching Unit (BMU). The similarity is usually defined by means of a distance measure, typically Euclidean distance.

After finding the BMU, the codebook is updated. The reference vectors of the BMU and its topological neighbors (according to the neighborhood function) are changed in order to be "closer" to the input vector in the input space. This adaptation procedure stretches the BMU and its topological neighbors towards the sample vector. The adaptation is given by:

$$W_j(n+1) \leftarrow W_j(n) + \eta(n)\, h_{ji}(n) \left[X(n) - W_j(n) \right], \tag{5}$$

where n is the iteration number, j is the neuron index that is considered in the current iteration, W_j is the prototype vector of cell j, $\eta(n)$ is a learning rate, $h_{ji}(n)$ is the neighborhood function defined centered on BMU, and $X(n)$ is the vector of the speckle patterns presented . Usually, both learning rate and the neighborhood function radius are decreasing as iterations progress.

Once trained, a SOM offers different ways to be visualized and analyzed. A matrix of distances between the codebook vectors of the cells and their neighbors is widely used [18]. Data samples can be projected onto the SOM by their BMU. Similar data will be projected in near cells.

In order to evaluate the quality of the map, two kinds of errors are considered: the quantization error and topographic error [19]. They tend to minimize when the map vectors perform an organized projection of the training pattern according to a similarity criterion.

Once the map is properly trained, colors can be assigned to cells according to the distance between prototype vectors with a palette generated heuristically. The color coding is such that topological nearby cells will have similar colors and those far according to this criterion will have distinct colors. Using this colored map, a color can be assigned to each input data according their BMUs.

To assign colors, we followed the next steps:

- Data are projected into a 2-dimensional space via the Principal Component Analysis (PCA).
- Codebook is projected according to the previous analysis.
- PCA codebook coordinates are scaled in the [0, 1] range.
- Colors are assigned to each cell of the PCA codebook coordinates using a RGB palette, selected to give a well differenced picture at most of LCD monitors.

The *k-means* clustering algorithm (using k-regions of interest) is performed to evaluate numerically the SOM colored regions [20]. In this way we achieve a colored map that will serve as a reference for coloring a new image based on where the descriptors of the temporal evolution of each (x, y) pixel impact as a result of consulting the SOM.

3 Results

Temporal descriptors were computed for all samples. Thus SA, DR and FGD descriptors were obtained for the stack of 300 images of 400x400 pixels, achieving 1,600,000 patterns vectors. In order to balance the type of information, part of the image background was trimmed, given that in the acquisition stage the background was rather oversized. Finally images of 300x300 pixels were computed. Fig. 3 shows images corresponding to descriptors of a specimen. Note that the discrimination ability in the four targeted areas (background, embryo, floury endosperm and vitreous endosperm) is not achieved with any of them.

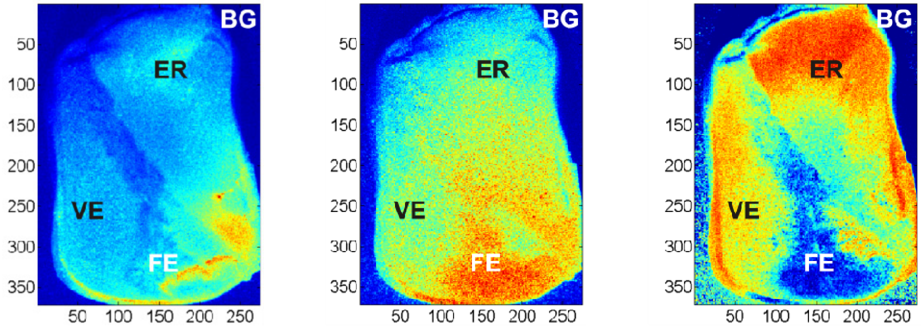


Fig. 3. Descriptor Images and the four region of interest: Vitreous Endosperm (VE), Floury Endosperm (FE), Embryo region (ER) and background (BG). Left: subtraction Average Descriptor; middle: dynamic Range Descriptor; right: Fuzzy Granular Descriptor.

The SOM toolkit for Matlab®, from the Laboratory of Computer and Information Science (CIS) at the Helsinki University of Technology, was used. In order to determine the SOM dimension, a growing configuration for increasing dimensions of the grid size was proposed, with a stopping criterion of minimizing the topographic error, obtaining a good projection of similar vectors in neighboring cells. Linear codebook initialization was performed. The dimension was determined as a 10x10 cell array. The number of iterations was set to 100 and the neighborhood Gaussian function was used. The learning function $\eta(n)$ ("inverse") was defined as in Eq 6, with $\eta_0=0.5$.

$$\eta(n) = \frac{\eta_0}{1+\frac{100n}{50}} , \tag{6}$$

The three inputs vectors components were the Subtraction Average, the Dynamic Range and the Fuzzy Granular Descriptor, all of them normalized according their variance.

Definition of map size was made disposing of all data generated by the 10 trials for a total of 90,000 vectors of three variables, generated by the images of 300x300 pixels. We carried out a scheme of training and test by a cross-validation process. Since there were 10 different experimental tests, we performed a leave-one-out scheme, with 10 runs, training with 9 cases and testing with the left out case. Each time, a cluster analysis was performed, with k-means algorithm (k=4, 4 targeted regions of interest) as shown in Fig. 4, achieving a good average of the Davies-Bouldin index. The 10x10 codebook vectors are colored according to the distance between prototype vectors with a RGB palette.

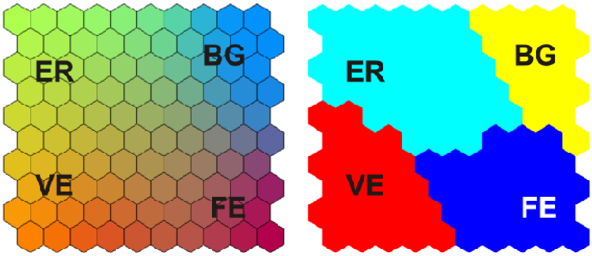


Fig. 4. Left: SOM colored map. Right: K-means partition labeled with regions of interest

Fig. 5 shows five corn seed images colored according with the labeled regions. To evaluate the goodness of the proposal we used expert's opinion, who considered, observing the prepared seeds and the pseudo-colored images of the processed speckle patterns that endosperm areas were properly identified. Also he noted that the methodology is considered valid to automate the calculation of the fractions of floury and vitreous endosperm of seeds.

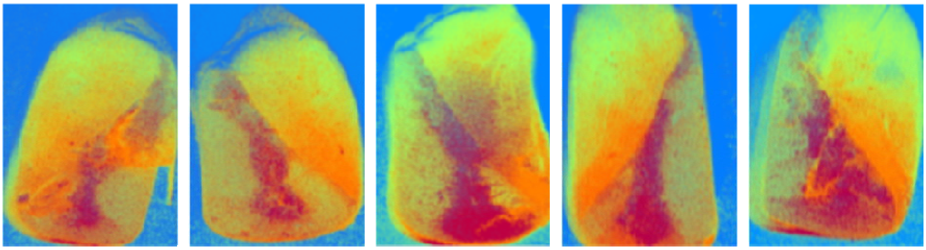


Fig. 5. Five specimens of flint corn seed, pseudo-colored according to the regions of interests given by the SOM codebooks

As shown in Fig. 5, pseudo coloring of corn flinty seeds enables differentiating the background region (blue color), that is highly spaced in the codebook from live parts. Endosperm starchy region is distinguished perfectly in a purple coloration, as also the region of the embryo (yellow-green zone) and the vitreous endosperm (orange zone). Thus, the four classes can be perfectly discriminated, a fact that was not possible using only one of the time domain descriptors.

4 Conclusions

In this work we propose the use of Self-Organizing Maps to visually identify diverse regions in corn seeds. The methodology proposed based in an optic approach is a new method, since the existing is an indirect estimation as a function of the capacity to float of the grains due to grain composition.

Particularly, it is addressed the segmentation of regions of the endosperm embryo with the aim to design an automated process to compute fractions of interest.

The results obtained by processing a set of specimens of flint corn are very encouraging. This proposal is novel in the field of agricultural technology and aims to provide a methodology for assessing the quality of corn based on the content of the endosperm. In this sense, the conjunction of an optical acquisition process based on the emission of coherent light and Computational Intelligence techniques have shown synergy in the creation of innovative processes applied to real problems of socio-economic impact.

References

1. Braga, R.A., Fabbro, I.M.D., Borem, F.M., Rabelo, G., Arizaga, R., Rabal, H.J., Trivi, M.: Assessment of seed viability by laser speckle techniques. Biosystems Engineering 86(3), 287–294 (2003), doi:10.1016/j.biosystemseng.2003.08.005
2. Sendra, H., Murialdo, S., Passoni, L.: Dynamic laser speckle to detect motile bacterial response of pseudomonas aeruginosa. Journal of Physics: Conference Series 90(1), 012064 (2007)
3. Pajuelo, M., Baldwin, G., Rabal, H., Cap, N., Arizaga, R., Trivi, M.: Biospeckle assessment of bruising in fruits. Optics and Lasers in Engineering 40(12), 13–24 (2003), doi:10.1016/S0143-8166(02)00063-5; <ce:title>Optics in Latin America part II</ce:title>

4. Dai Pra, A.L., Passoni, L.I., Rabal, H.J.: Fuzzy granular computing and dynamic speckle interferometry for the identification of different thickness of wet coatings. Infocomp, Journal of Computer Science 8(4), 45–51 (2009)
5. Fricke-Begemann, T., Gülker, G., Hinsch, K.D., Wolff, K.: Corrosion monitoring with speckle correlation. Appl. Opt. 38(28), 5948–5955 (1999), doi:10.1364/AO.38.005948.12
6. Rabal, H.J., Braga, R.A. (eds.): Dynamic Laser Speckle and Applications. CRC Press (2008)
7. Dai Pra, A.L., Passoni, L.I., Rabal, H.: Evaluation of laser dynamic speckle signals applying granular computing. Signal Processing 89(3), 266–274 (2009), doi:10.1016/j.sigpro.2008.08.012
8. Drury, S.M., Reynolds, T.L., Ridley, W.P., Bogdanova, N., Riordan, S., Nemeth, M.A., Sorbet, R., Trujillo, W.A., Breeze, M.L.: Composition of Forage and Grain from Second-Generation Insect-Protected Corn MON 89034 Is Equivalent to That of Conventional Corn (Zea mays L). J. Agric. Food Chem. 56(12), 4623–46302 (2008)
9. Bragachini, M.A., Casini, C., Ustarroz, F., Saavedra, A.E., Mendez, J.A., Errasquin, L.: La calidad del grano de Maíz. En: Maíz Cadena de Valor Agregado. E.E.A. INTA Balcarce PRECOP II. Actualización Técnica 54, 9–10 (2010)
10. Mahanna, B., Thomas, E.: (April 2012), `https://www.pioneer.com/home/site/us/menuitem.b8381b50868d5c8176f576f5d10093a0/`
11. Lepes, I.T., Miotto, R.M., Cedro, A.V., Ruegg, O.E.: Test de flotación en maíces duros argentinos. I Congreso Nacional de Maiz, Pergamino, Argentina, pp. 287–298 (1976)
12. Guzman, M., Meschino, G.J., Dai Pra, A.L., Trivi, M., Passoni, L.I., Rabal, H.: Dynamic laser speckle: decision models with computational intelligence techniques. Speckle 0001, 738717–738717-8 (2010)
13. Etchepareborda, P., Federico, A., Kaufmann, G.: Sensitivity evaluation of dynamic speckle activity measurements using clustering methods. Appl. Opt. 49, 3753–3761 (2010)
14. Meschino, G., Murialdo, S., Passoni, L., Rabal, H., Trivi, M.: Biospeckle image stack process based on artificial neural networks. In: 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), August 31-September 4, pp. 4056–4059 (2010), doi:10.1109/ IEMBS.2010.5627620
15. Braga, R.A., Silva, W.S., Sáfadi, T., Nobre, C.M.B.: Time history speckle pattern under statistical view. Optics Communications 281(9), 2443–2448 (2007) ISSN 0030-4018, doi:10.1016/j.optcom.2007.12.069
16. Trivi, M.: Dynamic Speckle in Dynamic Laser Speckle and Applications. In: Rabal, H.J., Braga, R.A. (eds.), pp. 21–51. CRC Press (November 2008)
17. Kohonen, T.: Self-Organizing Map. Springer (1995)
18. Vesanto, J., Sulkava, M.: Distance Matrix Based Clustering of the Self-Organizing Map. In: Dorronsoro, J.R. (ed.) ICANN 2002. LNCS, vol. 2415, pp. 951–956. Springer, Heidelberg (2002)
19. Kiang, M.Y.: Extending the Kohonen self-organizing map networks for clustering analysis. Computational Statistics Data Analysis 38, 161–180 (2001)
20. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11, 586–600 (2000)

Online Visualization of Prototypes and Receptive Fields Produced by LVQ Algorithms

David Nova and Pablo A. Estévez

Department of Electrical Engineering and Advanced Mining Technology Center,
University of Chile, Casilla 412-3, Santiago, Chile
dnovai@ug.uchile.cl, pestevez@ing.uchile.cl

Abstract. A new approach is proposed to visualize online the training of learning vector quantization algorithms. The prototypes and data samples associated to each receptive field are projected onto a two-dimensional map by using a nonlinear transformation of the input space. The mapping finds a set of projection vectors by minimizing a cost function, which preserves the local topology of the input space. The proposed visualization is tested on two datasets: image segmentation and pipeline. The usefulness of the method is demonstrated by studying the behavior of Generalized LVQ, Supervised Neural Gas and Harmonic to Minimum LVQ algorithms on high-dimensional datasets.

Keywords: Learning Vector Quantization, Supervised Neural Gas, Harmonic to Minimum, Data visualization, Data Projection, Topology preservation.

1 Introduction

Many methods have been proposed in pattern recognition such as self-organizing maps (SOM) and learning vector quantization (LVQ) methods [1]. These methods are prototype-based, being the former unsupervised and the latter supervised. An advantage of prototype based methods is the intuitive understanding of the clusterization or classification obtained. Another advantage, widely exploited by SOMs is the use of visualization schemes. However, there is a lack of visualization schemes for LVQ algorithms. The original LVQ is based on heuristic learning or a priori domain knowledge to minimize the classification error. In the last years, several algorithms have been proposed to enhance the original LVQ obtaining faster convergence, better approximation to the Bayesian decision border, and robustness to different initializations. A breakthrough model is the generalization of LVQ, the so-called generalized learning vector quantization (GLVQ) proposed in [2]. It provides a robust and efficient approximation of the Bayesian decision border, through a continuous and differentiable cost function, usually outperforming the original LVQ algorithm.

For high dimensional datasets, the adaptation of LVQ algorithms during training is usually monitored by using the misclassification rate as an indicator. In this paper, we propose an online visualization scheme appropriate for LVQ algorithms, in order to visualize the prototypes and their respective receptive fields. An online visualization scheme is useful for studying the behavior of the LVQ algorithms during training, either

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 173–182.
DOI: 10.1007/978-3-642-35230-0_18

for didactic purposes or for research. In addition, we can get a fine-tuned map of the final classification.

The remainder of this paper is organized as follows: In section 2, three modern LVQ algorithms are briefly introduced. In section 3, the proposed method is presented, and in section 4 the results are shown. Finally, in section 5 the conclusions are drawn.

2 Learning Vector Quantization Algorithms

A training dataset $X = \{(\mathbf{x}_i, l_i) \subset \mathbb{R}^D \times \{1, ..., C\} | i = 1, ..., N\}$; $\mathbf{x} = (x_1, ..., x_D) \in \mathbb{R}^D$ is assumed, where D is the data dimensionality and C the number of different classes. The network consists of a number of prototypes, which are characterized by their vectors in a feature space $\mathbf{w}_i \in \mathbb{R}^D$ and their class labels $c(\mathbf{w}_i) \in \{1, ..., C\}$. The classification scheme is based on the best matching unit (BMU) (winner-takes-all). The receptive field of each prototype $\mathbf{w}_i$ can be described as follows:

$$R^i = \{\mathbf{x} \in X | \forall \mathbf{w}_j (j \neq i \rightarrow d(\mathbf{w_i}, \mathbf{x}) < d(\mathbf{w_j}, \mathbf{x}))\}, \tag{1}$$

where $d(\mathbf{w}, \mathbf{x})$ is a distance measure. Learning aims at determining the weight vectors of prototypes, such that the given training dataset is mapped to their corresponding class labels. In what follows, we briefly describe three different modern LVQ methods.

2.1 Generalized Learning Vector Quantization

GLVQ [2] has an underlying cost function related to a maximization of the hypothesis margin of the classifier, as shown below

$$E_{GLVQ} = \sum_{i=1}^{l} \phi\left(\mu(\mathbf{x}_i)\right), \tag{2}$$

where $\phi(\cdot)$ is the logistic sigmoid function, $\mu(\mathbf{x}_i) = \frac{d_J(\mathbf{x}_i) - d_K(\mathbf{x}_i)}{d_J(\mathbf{x}_i) + d_K(\mathbf{x}_i)}$, $d_J(\mathbf{x}_i) = d(\mathbf{w}_J, \mathbf{x}_i)$ is the distance of data point $\mathbf{x}_i$ from its closest prototype $\mathbf{w}_J$ having the same class label y, and $d_K(\mathbf{x}_i) = d(\mathbf{w}_K, \mathbf{x}_i)$ is the distance from the closest prototype $\mathbf{w}_K$ having a class label different from y. Using stochastic gradient descent method, the following learning rules are obtained:

$$\begin{array}{l} \Delta \mathbf{w}_J = +2 \cdot \epsilon \cdot \phi'(\mu(\mathbf{x}_i)) \cdot \mu^+ \cdot (\mathbf{x}_i - \mathbf{w}_J) \\ \Delta \mathbf{w}_K = -2 \cdot \epsilon \cdot \phi'(\mu(\mathbf{x}_i)) \cdot \mu^- \cdot (\mathbf{x}_i - \mathbf{w}_K) \end{array}, \tag{3}$$

where $\mu^+ = \frac{2 \cdot d_K}{(d_K + d_J)^2}$, $\mu^- = \frac{2 \cdot d_J}{(d_K + d_J)^2}$ and $\epsilon \in]0, 1[$ is the learning rate.

2.2 Supervised Neural Gas

Supervised Neural Gas (SNG) [4] adds the idea of neighborhood cooperativity into GLVQ to avoid the dependency on the initialization. All prototypes of the respective class are adapted towards the data point according to their ranking. The SNG cost function is as follows:

$$E_{SNG} = \sum_{\mathbf{x} \in X} \sum_{\mathbf{w}_r \in \mathbf{W}_{c_{\mathbf{x}_i}}} \frac{h_\gamma(r, \mathbf{x}, \mathbf{W}_{c_\mathbf{x}}) \cdot \phi(\mu(\mathbf{x}))}{C(\gamma, K_{c_\mathbf{x}})} \tag{4}$$

where

$$h_\gamma(r, \mathbf{x}_i, \mathbf{W}) = \exp\left(-\frac{k_r(\mathbf{x}_i, \mathbf{W}_{c_\mathbf{x}})}{\gamma}\right), \tag{5}$$

denotes the degree of neighborhood cooperativity, $k_r(\mathbf{x}, \mathbf{W})$ is the ranking of prototype $\mathbf{w}_r$ given input $\mathbf{x}_i$, and $C(\gamma, K)$ is a normalization depending on the neighborhood size γ, K is the cardinality of $\mathbf{W}_{c_\mathbf{x}}$, $\mathbf{W}_{c_\mathbf{x}} = \{\forall \mathbf{w}_j \in \mathbb{R}^D | c(\mathbf{w}_i) = c(\mathbf{x}_i)\}$ and ϕ is a sigmoid function. For each $\mathbf{x}$, all prototypes $\mathbf{w}_J \in \mathbf{W}_{c_{\mathbf{x}_i}}$ are updated as follows:

$$\varDelta \mathbf{w}_J = +2 \cdot \epsilon \cdot \frac{\phi'(\mu(\mathbf{x}_i)) \cdot \mu^+ \cdot h_\gamma(r, \mathbf{x}, \mathbf{W}_{c_{\mathbf{x}_i}})}{C(\gamma, K_{c_\mathbf{x}})} \cdot (\mathbf{x} - \mathbf{w}_J) \tag{6}$$

and the closest prototype of a different class is adapted as follows:

$$\varDelta \mathbf{w}_K = -2 \cdot \epsilon \cdot \sum_{\mathbf{w}_J \in \mathbf{W}_{c_\mathbf{x}}} \frac{\phi'(\mu(\mathbf{x}_i)) \cdot \mu^- \cdot h_\gamma(r, \mathbf{x}_i, \mathbf{W}_{c_\mathbf{x}})}{C(\gamma, K_{c_{\mathbf{x}_i}})} \cdot (\mathbf{x}_i - \mathbf{w}_K), \tag{7}$$

where $\mu^+ = \frac{2 \cdot d_K}{(d_K + d_J)^2}$, $\mu^- = \frac{2 \cdot d_J}{(d_K + d_J)^2}$ and $\epsilon \in]0, 1[$ is the learning rate.

2.3 Harmonic to Minimum Learning Vector Quantization

In order to repair the sensitiveness to different initializations, the Harmonic to Minimum Learning Vector Quantization (H2MLVQ) was introduced in [5], based on the K-harmonic means algorithm [6]. Here all prototypes having the same class label than the current sample are updated. Likewise, all prototypes with a different label than the sample are adjusted. The objective function for H2MLVQ is:

$$E_{H2M-LVQ} = \sum_{i=1}^{l} \phi\left(\mu(\mathbf{x}_i)\right), \tag{8}$$

where $\mu(\mathbf{x}_i) = \frac{d_j^H - d_k^H}{d_j^H + d_k^H}$ and d_J^H, d_K^H are the harmonic average distances [6]. Using the gradient descent method the following update rules are obtained:

$$\varDelta \mathbf{w}_J = -2 \cdot \epsilon \cdot \phi'(\mu(\mathbf{x}_i)) \cdot \mu^+ \cdot \alpha_J \cdot (\mathbf{x}_i - \mathbf{w}_J), \tag{9}$$

$$\varDelta \mathbf{w}_K = +2 \cdot \epsilon \cdot \phi'(\mu(\mathbf{x}_i)) \cdot \mu^- \cdot \alpha_K \cdot (\mathbf{x}_i - \mathbf{w}_K), \tag{10}$$

where $\mu^+ = \frac{2 \cdot d_K^H}{(d_K^H + d_J^H)^2}$, $\mu^- = \frac{2 \cdot d_J^H}{(d_K^H + d_J^H)^2}$, $\epsilon \in]0, 1[$ is the learning rate, α_K and α_J are adaptive coefficients, for details see [5].

3 Online Visualization Method for LVQ Algorithms (OVI-LVQ)

The basic idea is to easily visualize the prototypes and samples during the training of LVQ algorithms on high-dimensional datasets. First, a visualization of the prototypes and samples in the 2D output space projections is performed online during the LVQ training. At the end of training, a recalling procedure is used to fine-tune the resulting map.

Let $\{\mathbf{x}_i : 1 \leq i \leq N\}$ and $\{\mathbf{w}_j : 1 \leq j \leq M\}$ be D-dimensional input samples and prototypes (codebook vectors), respectively. In addition, $\{\mathbf{y}_i : 1 \leq i \leq N\}$ and $\{\mathbf{z}_j : 1 \leq j \leq M\}$ are the corresponding A-dimensional output samples and codebook positions (prototype projections) in the output space, respectively, with $A \ll D$. Let $d_{j,k}$ and $D_{j,k}$ be the pairwise Euclidean distances in the input and output spaces, respectively. These distances are defined as follows:

$$d_{j,k} = ||\mathbf{w}_j - \mathbf{w}_k||,\, D_{j,k} = ||\mathbf{z}_j - \mathbf{z}_k||. \tag{11}$$

3.1 Online Visualization of Prototypes

In the online visualization phase, the OVI-NG algorithm proposed in [3] is used to project the prototypes. In the next subsection we will explain our proposed method to project samples using the prototypes as references. A global cost function is defined as follows:

$$E = \frac{1}{2}\sum_{j=1}^{N}\sum_{k\neq j}(D_{j,k} - d_{j,k})^2 = \frac{1}{2}\sum_{j=1}^{N}\sum_{k\neq j}E_{j,k}, \tag{12}$$

where the function F is defined as:

$$F(f) = e^{-\left(\frac{f}{\sigma(t)}\right)},\ \text{where}\ \sigma(t) = \sigma_0\left(\frac{\sigma_f}{\sigma_0}\right)^{\left(\frac{t}{T_{max}}\right)}, \tag{13}$$

and $\sigma(t)$ is the width of the neighborhood which decreases with the number of iterations as shown in eq. (13).

A simplified version of gradient descent is used, where the codebook position associated with the winner unit, $\mathbf{z}_{j^*}$, is fixed, and the $N-1$ remaining positions are moved towards the winner's position. Therefore, the ranking in the output space s_{j,j^*} takes values in the range $\{1, ..., N-1\}$, where $s = 1$ corresponds to the nearest codebook position with respect to the winner's position. In order to minimize eq. (12). The following update rule for the codebook positions is used:

$$\mathbf{z}_j(t+1) = \mathbf{z}_j + \alpha F(s_{j,j^*})\frac{(D_{j,j^*} - d_{j,j^*})}{D_{j,j^*}} \times (\mathbf{z}_{j^*} - \mathbf{z}_j), \tag{14}$$

where α is the learning rate.

This online visualization method for projecting prototypes can be added to most LVQ algorithms. We can imagine that a virtual link exists between prototypes (codebook vectors) in the input space and their respective codebook positions in the output space. The initial topology of the network is a set of M prototypes. Each prototype j has associated a D-dimensional codebook vector, $\mathbf{w}_j$ in the input space, and a two-dimensional codebook position, $\mathbf{z}_j$ in the output space. The algorithm is as follows:

1. Initialize the codebook vectors associated to the prototypes, $\mathbf{w}_j$, and the codebook positions, $\mathbf{z}_j$, randomly.
2. Present an input vector, $\mathbf{x}_i(t)$, to the LVQ network $(i = 1, ..., N)$ at iteration t.
3. Find the winner prototype j^* which belongs to the same class than $\mathbf{x}_i(t)$, by using:

$$j^* = \underset{\forall j | c(\mathbf{w}_j) = c(\mathbf{x}_i(t))}{argmin} \|\mathbf{x}_i(t) - \mathbf{w}_j(t)\| \tag{15}$$

4. Update the prototype (codebook) vectors according to the update rules of the corresponding LVQ algorithm version.
5. Generate the ranking in output space $s_{j,j^*} = s(z_{j^*}(t), z_j(t)) \in \{1, ..., M-1\}$ for each codebook position $\mathbf{z}_j(t)$ with respect to the codebook position associated with the winner unit $\mathbf{z}_{j^*}(t),\ j \neq j^*$.
6. Update the codebook positions using eq. (14).
7. If $t < t_{max}$, go back to step 2.

3.2 Sample Recalling Procedure

In order to project samples a recalling procedure is proposed. We assume that the codebook position vectors have already been adjusted using the procedure described in section 3.1. These codebook positions are used as references for the projection of samples in the output space. The sample recalling can also be done online, although for the sake of computational time only a subset of samples may be projected during training. At the end of training, after convergence a fine-tuned projection is obtained using all training samples.

The receptive field R^i associated with i-th each codebook vector obtained by LVQ training is considered. Because virtual links exist between the input space and output space, there would be also a receptive field associated with each codebook position in the output space. The proposed cost function is as follows:

$$\hat{E} = \sum_{i}^{M} \left((1-\lambda)E_{local} + \lambda E_{global}\right), \tag{16}$$

where:

$$E_{local} \equiv \sum_{j \in R^i} \sum_{k \neq j} E_{j,k} = \sum_{j \in R^i} \sum_{k \neq j} (D_{j,k} - d_{j,k})^2 \cdot F(s_{j,k}), \tag{17}$$

$$E_{global} \equiv \sum_{j \in R^i} \sum_{m=1}^{M} E_{j,m} = \sum_{j \in R^i} \sum_{m=1}^{M} (D_{j,m} - d_{j,m})^2 \cdot F(s_{j,m}). \tag{18}$$

The parameter λ in eq. (16) is used to control the trade-off between the local and global topology preservation. Eq. (17) deals with pairwise distances of elements belonging to the same receptive field both in input and output spaces. This term contributes to preserving the inner local topology. On the other hand, eq. (18) deals with the pairwise distances between samples associated with a given receptive field and all the prototypes.

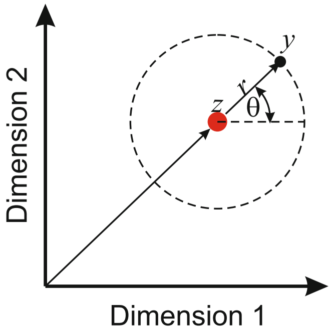


Fig. 1. Scheme representing a two-dimensional output space. The nearest codebook position **z** (*solid red dot*) is used as a reference to initialize the output sample **y** (*solid black dot*), where r and θ represent the distance and the angle of orientation, respectively.

This term contributes to preserving the global topology of the data. By using stochastic gradient descent, we get:

$$\frac{\partial E_{local}}{\partial \mathbf{y}_j} = \sum_{k \neq j} F(s_{j,k}) \cdot \frac{(D_{j,k} - d_{j,k})}{D_{j,k}} \times (\mathbf{y}_k(t) - \mathbf{y}_j(t)), \tag{19}$$

$$\frac{\partial E_{global}}{\partial \mathbf{y}_j} = \sum_{m=1}^{M} F(s_{j,m}) \cdot \frac{(D_{j,m} - d_{j,m})}{D_{j,m}} \times (\mathbf{z}_m(t) - \mathbf{y}_j(t)), \tag{20}$$

where F is defined in eq. (13). The update rule for sample projection is the following:

$$\mathbf{y}_j(t+1) = \mathbf{y}_j(t) + \alpha \cdot \left[(1-\lambda) \frac{\partial E_{local}}{\partial \mathbf{y}_j} + \lambda \frac{\partial E_{global}}{\partial \mathbf{y}_j} \right], \tag{21}$$

where α is the learning rate.

Using a priori knowledge that each sample belongs to a receptive field, an initialization strategy is proposed. The distance and orientation of a sample with respect to its nearest prototype is considered, as shown in fig. 1. The initial position of the sample projections in the output space is set as follows:

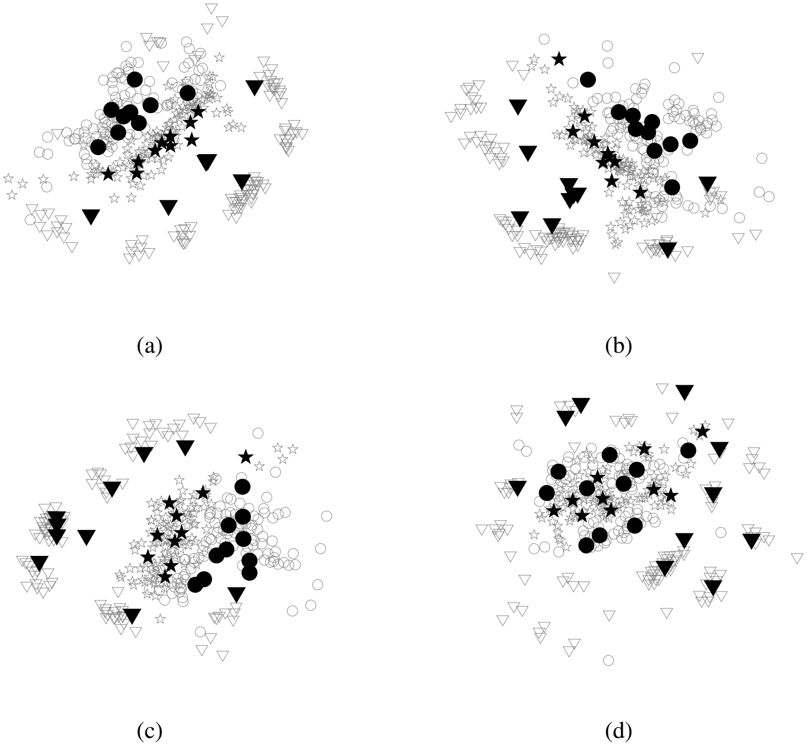
$$\mathbf{y}_j(t=0) = \begin{bmatrix} z_{i,1} + r_j \cdot \cos(\theta_j) \\ z_{i,2} + r_j \cdot \sin(\theta_j) \end{bmatrix}, \; \theta_j = \min_{\theta \in [0, 2\pi]} \left(\sum_{i=1}^{M} \| D_{i,j} - d_{i,j} \| \right), \tag{22}$$

where $z_{i,1}$ and $z_{i,2}$ are the components of the codebook position vector in the output space (assumed fixed in the previous prototype projection phase), which is associated with the i-th receptive field, r is the a priori known distance in the input space between the i-th codebook vector and j-th input sample. In this initialization procedure, we assume that the absolute value of pairwise distances are the same in the input and output spaces, therefore only the angle is unknown. The angle θ_j is determined by minimizing the pairwise distance between the input sample and all prototype vectors such as in eq. (22), both in the input and output spaces.

The sample recalling procedure is as follows:

1. Initialize all the output samples $\mathbf{y}_j$ using eq. (22).
2. Select the receptive field associated to the i-th prototype.
3. Present all the output samples $\mathbf{y}_j$ which belong to the i-th prototype, and update their positions using the update rule defined in eq. (21).
4. If $t < t_{max}$, go back to step 2.

The topology preservation measure q_m defined in [7] is used as a quality performance measure of the mapping. The range of the q_m measure is between 0 and 1, where $q_m = 0$ indicates poor neighborhood preservation between the input and output spaces, and $q_m = 1$ indicates a perfect neighborhood preservation. We use as a reference the projections obtained with classical unsupervised projection methods such as CCA [8] and Sammon [9]. Another quality measurement is the classification rate obtained using the 2D map and the receptive fields obtained.


(a)

(b)

(c)

(d)

Fig. 2. 2-D visualizations of the pipeline dataset obtained using OVI-LVQ projection (a)-(c) and CCA projection (d). The plots correspond to different LVQ algorithms: (a) GLVQ, (b) SNG, (c) and (d) H2MLVQ. Bolds marks denote the position of the prototype projections for the different classes.

4 Experiments

The OVI-LVQ method was used with GLVQ, SNG and H2MLVQ algorithms to visualize the prototypes and receptive fields obtained with high-dimensional datasets. In all experiments the parameters of the OVI-LVQ were set to: $\epsilon = 0.05$, $\alpha = 0.05$, $T_{max} = 100$ and $\lambda = 0.5$.

Two datasets were used: Image Segmentation [10] and Pipeline[1]. Image segmentation dataset consists of 19 dimensional feature vectors and seven classes. The parameters for this dataset were set as follow: 5 prototypes per class, $\sigma_0 = 3500$, $\sigma_f = 700$. The recalling parameters were set as follows: $\sigma_0 = 20$ and $\sigma_0 = 2$. The training and test set consist of 210 and 2100 samples, respectively. Pipeline dataset consists of 13 atributes and three classes. For pipeline dataset, the following parameters were used: 10 prototypes per class, $\sigma_0 = 4000$, $\sigma_f = 800$. The recalling parameters were set as follows: $\sigma_0 = 30$ and $\sigma_0 = 3$. The training and test set consist of 300 and 700 samples, respectively.

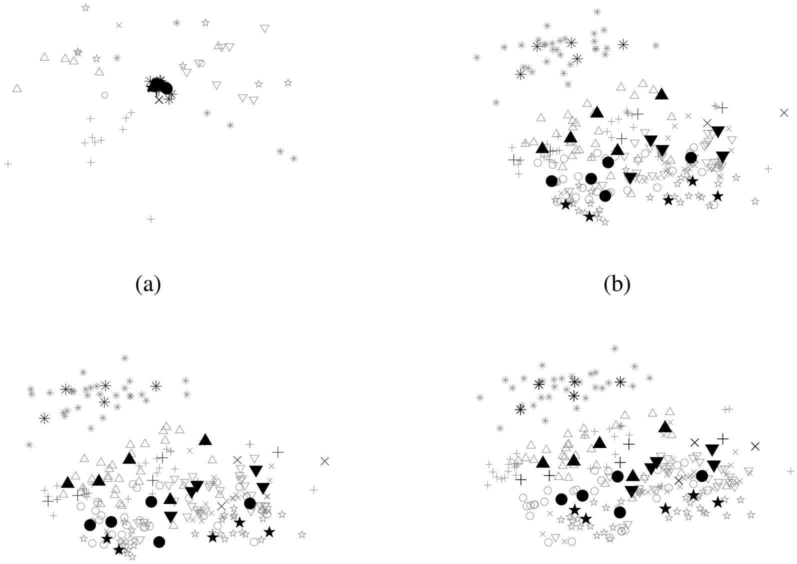


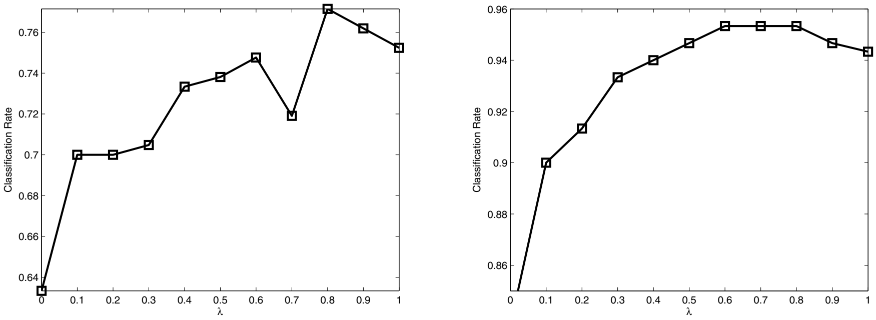
Fig. 3. 2D visualizations of the image segmentation dataset training evolution for H2MLVQ. (a) Initialization, (b) after 50 epochs, (c) after 200 epochs and (d) final sample recalling visualization. Bold marks denote prototype projections for the different classes.

Fig. 2 (a)-(c) shows 2D visualizations obtained with OVI-LVQ for the pipeline dataset. Fig. 2 (d) shows a projection using CCA, for comparison purposes. The different classification rates obtained for each LVQ methods (see Table 1) are reflected in the distribution of prototypes for each map. GLVQ and SNG get trapped in local minima

[1] Non-linearity and Complexity Research Group, www1.aston.ac.uk/ncrg/.

Table 1. Percentage of correct classifications on the test set for pipeline and image segmentation datasets. Results shown on columns OVI-LVQ, Sammon and CCA correspond to classification rates on the output space.

	Pipeline Dataset				Image Segmentation Dataset			
	Input Space	OVI-LVQ	Sammon	CCA	Input Space	OVI-LVQ	Sammon	CCA
GLVQ	94.32	86.33	82.33	84.67	70.28	68.57	37.14	50.48
SNG	98.14	89.50	72.00	87.33	80.80	70.00	63.81	68.33
H2MLVQ	98.94	94.67	81.67	86.67	86.61	73.81	70.00	72.38



(a) (b)

Fig. 4. Classification rate as a function of λ in the output space using H2MLVQ algorithm on: (a) Image segmentation dataset, (b) Pipeline dataset.

due to the initialization sensitiveness problem associated with the original GLVQ. It can be seen in the final maps in fig. 2 that there are prototypes with empty receptive fields. On the contrary, H2MLVQ gets rid of the initialization problem, and all its prototypes are active and well located. From the topology preserving point of view, the q_m values are within $\pm 3.5\%$ in average of those obtained using Sammon and CCA projections.

Fig. 3 shows the 2D visualizations obtained using OVI-LVQ method for the H2M LVQ algorithm trained on the image segmentation dataset. Figs. 3(a)-(c) show online projections after 0, 50 and 200 training epochs. Fig. 3(d) depicts the final map after fine-tuning by using all samples. Table 1 shows the classification rates obtained with the three LVQ methods for the pipeline and image segmentation datasets on the test set. For the latter dataset GLVQ gets stuck in a local minima and can not achieve a good quantization. A correct quantization is achieved with H2MLVQ and SNG. The sensitiveness initialization problem associated to GLVQ is reflected in that this method achieved in general a lower classification performance in comparison with the H2MLVQ and SNG algorithms. Furthermore, as shown in Table 1, the proposed OVI-LVQ projection method outperformed both Sammon and CCA projections. The topology preservation measurement q_m for the OVI-LVQ projection of the image segmentation dataset was computed. For all neighborhood sizes, the q_m value was within $\pm 2.6\%$ in average of the values obtained with Sammon and CCA projections. This result indicates that OVI-LVQ preserves well the local topology. Fig. 4 shows how the classification rate in the output space varies with the parameter λ. The maximum classification rate is obtained for $\lambda = 0.8$ (77.14%) for the image segmentation dataset and for $\lambda = 0.6 - 0.8$ (95.33%) for the pipeline dataset. This means that for classification purposes we have to weight more the global error versus the local error.

5 Conclusions

A new visualization method ad-hoc for LVQ algorithms has been proposed. The OVI-LVQ method allows us to visualize online the evolution of prototypes and receptive fields during training of LVQ algorithms. This could be useful for didactic purposes, but also for visualizing the results of LVQ methods including their errors. Another possible aplication is the visualization of trajectories of sequential data. The receptive field concept used in our method allows obtaining a good quality mapping and at the same time reduce the computational cost due to less distance calculations. The proposed initialization method enables a good starting position for the output samples which can be further refined by the recalling of samples procedure. The parameter λ associated to the trade-off between global and local topology preservation helps to increase the classification rate based on the 2D map, which indicates that the receptive fields of the prototypes are preserved in the projection.

Acknowledgements. This research was supported by Conicyt-Chile under grant Fondecyt 1110701.

References

1. Kohonen, T.: Self-organizing maps. Springer-Verlag New York, Inc., Secaucus (1997)
2. Sato, A., Yamada, K.: Generalized Learning Vector Quantization. In: Advances in Neural Information Processing Systems, vol. 8, pp. 423–429 (1996)
3. Estévez, P.A., Figueroa, C.J.: Online data visualization using the neural gas network. Neural Networks 19(67), 923–934 (2006)
4. Hammer, B., Strickert, M., Villmann, T.: Supervised neural gas with general similarity measure. Neural Processing Letters 21(1), 21–44 (2005)
5. Qin, A.K., Suganthan, P.N.: Initialization insensitive LVQ algorithm based on cost-function adaptation. Pattern recognition 38(5), 773–776 (2005)
6. Zhang, B., Hsu, M., Dayal, U.: K-harmonic means-a data clustering algorithm. Hewllet-Packard Research Laboratory Technical Report HPL-1999-124 (1999)
7. König, A.: Interactive visualization and analysis of hierarchical neural projections for data mining. IEEE Transactions on Neural Networks 11(3), 615–624 (2000)
8. Demartines, P., Herault, J.: Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets. IEEE Transactions on Neural Networks 8(1), 148–154 (1997)
9. Sammon, J.W.: A Nonlinear Mapping Structure Analysis. IEEE Transactions on Computers 100(5), 401–409 (1969)
10. Frank, A., Asuncion, A.: UCI Machine Learning Repository. School of Information and Computer Science. University of California, Irvine (2010), http://archive.ics.uci.edu/ml

Efficient Approximations of Kernel Robust Soft LVQ

Daniela Hofmann, Andrej Gisbrecht, and Barbara Hammer

CITEC Center of Excellence, Bielefeld University, Germany
{dhofmann,agisbrec,bhammer}@techfak.uni-bielefeld.de

Abstract. Robust soft learning vector quantization (RSLVQ) constitutes a probabilistic extension of learning vector quantization (LVQ) based on a labeled Gaussian mixture model of the data. Training optimizes the likelihood ratio of the model and recovers a variant similar to LVQ2.1 in the limit of small bandwidth. Recently, RSLVQ has been extended to a kernel version, thus opening the way towards more general data structures characterized in terms of a Gram matrix only. While leading to state of the art results, this extension has the drawback that models are no longer sparse, and quadratic training complexity is encountered. In this contribution, we investigate two approximation schemes which lead to sparse models: k-approximations of the prototypes and the Nyström approximation of the Gram matrix. We investigate the behavior of these approximations in a couple of benchmarks.

1 Introduction

Due to its very intuitive training and classification behavior which is often matched by excellent classification accuracy, learning vector quantization (LVQ) as proposed by Kohonen [9] more than 20 years ago still constitutes a very popular and widely used classification scheme. In the last years, quite a few variants have been proposed which accompany the original scheme by formal cost functions based on which powerful extensions towards metric learning, for example, can be derived [14,16]. In this contribution, we will focus on the approach robust soft LVQ (RSLVQ) as proposed in [16] since it offers a very intuitive representation of data in terms of a mixture of labeled Gaussians. Training takes place by means of an optimization of the likelihood ratio, leading to updates similar to LVQ2.1 in the limit of small bandwidth. Being a prototype-based approach, LVQ offers a very intuitive interface for the applicant: she can directly inspect the prototypes in the same way as data. Regarding the crucial impact of interpretability of the given models in many fields, this fact constitutes an important benefit of LVQ classifiers [17].

In many application areas, data sets are becoming more and more complex and additional structural information is often available. Examples include chemical structures, biological networks, social network data, graph structures, dedicated images, etc. Often, dedicated similarity measures have been developed to compare such data. This holds for bioinformatics sequences, graphs, or tree structures as they occur in linguistics, time series data, functional data arising in mass spectrometry, relational data stored in relational databases, etc. These

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 183–192.
DOI: 10.1007/978-3-642-35230-0_19

data are no longer explicitly represented as Euclidean vectors, rather, pairwise similarities are available. Hence LVQ classifiers cannot be used directly.

There exists a couple of techniques which can deal with more general data structures of similarity measures, see e.g. the approaches [4,5]. In particular, several popular prototype-based algorithms have been extended to deal with more general data: Exemplar based variants, for example, restrict the location of prototypes to exemplars, where dissimilarities are well defined, see e.g. the approaches [10,3]. These techniques, however, have the drawback that a smooth adaptation of prototypes is no longer possible and problems can occur especially if the given data are spare. More general smooth adaptation is offered by relational extensions such as relational neural gas or relational learning vector quantization [6]. Kernelization constitutes another possibility such as proposed for neural gas, self-organizing maps, or different variants of learning vector quantization [1,12]. Recently, a kernel variant of RSLVQ has been proposed which matches the classification performance of support vector machines (SVM) in a variety of benchmarks [7]. By formalizing the interface to the data as a general similarity or dissimilarity matrix, complex structures can be dealt with, relying on dedicated structure kernels or an explicit Gram matrix, for example [11,5].

Kernel RSLVQ, unlike RSLVQ, represents prototypes implicitly by means of a linear combination of data in kernel space. This has two drawbacks: on the one hand, prototypes are no longer directly interpretable, since the vector of linear coefficients is usually not sparse. Hence, in theory, all data points can contribute to the prototype. On the other hand, an adaptation step does no longer scale linearly with the number of data points, rather, quadratic complexity is required. This makes the technique infeasible if large data sets are considered. In this contribution, we propose two different approximation schemes and we investigate the effect of these techniques in a variety of benchmarks. First, we consider the Nyström approximation of Gram matrices which has been proposed in the context of SVMs in [18]. It constitutes a low rank approximation of the matrix based on a small subsample of the data. Assuming a fixed size of the subsample, a linear adaptation technique results. This approximation technique accounts for an efficient update, but prototypes are still distributed. As an alternative, we investigate an approximation of prototypes in terms of their k closest exemplars after training. This way, sparse models are obtained, albeit the technique still displays quadratic complexity. The effects of these approximations on the accuracy are tested in a couple of benchmarks.

Now we first review RSLVQ and its kernel variant. We explain the Nyström approximation and its incorporation into kernel RSLVQ. Afterwards, we explain the k-approximation. We test the performance using benchmarks similar to [2].

2 Kernel Robust Soft Learning Vector Quantization

Learning vector quantization (LVQ) constitutes a prototype based classification algorithm proposed by Kohonen [9]. Its model complexity is controlled by the number of prototypes which act as a compressed representation of the given data. This feature has been used e.g. in the context of life-long learning models, see e.g. [8]. Basic LVQ learning is directly based on Hebbian learning. One of the first proposals of a cost function of LVQ can be found in [13]. An alternative which is based on a probabilistic model of the data has been proposed in [16]: RSLVQ. This method models data by a mixture of Gaussians and derives learning thereof by means of a maximization of the log likelihood ratio of the given data. In the limit of small bandwidth, a learning rule which is similar to LVQ2.1 is obtained.

Assume data $\xi_k \in \mathbb{R}^n$ are labeled with labels y_k. A RSLVQ network represents a mixture distribution characterized by m prototypes $w_j \in \mathbb{R}^n$. The labels of prototypes $c(w_j)$ are fixed. σ_j denote the bandwidths. Mixture component j induces the probability

$$p(\xi|j) = \text{const}_j \cdot \exp(f(\xi, w_j, \sigma_j^2))$$

with normalization constant const_j and function f

$$f(\xi, w_j, \sigma_j^2) = -\|\xi - w_j\|^2/\sigma_j^2 \,.$$

The probability of data point ξ is defined as mixture

$$p(\xi|W) = \sum_j P(j) \cdot p(\xi|j)$$

with prior $P(j)$ and parameters W of the model. The probability of a data point ξ and a given label y is

$$p(\xi, y|W) = \sum_{c(w_j)=y} P(j) \cdot p(\xi|j) \,.$$

Learning aims at an optimization of the log likelihood ratio

$$L = \sum_k \log \frac{p(\xi_k, y_k|W)}{p(\xi_k|W)} \,.$$

A stochastic gradient ascent yields the following update rules, given a data point (ξ_k, y_k)

$$\Delta w_j = \alpha \cdot \begin{cases} (P_y(j|\xi_k) - P(j|\xi_k)) \cdot \text{const}_j \cdot \partial f(\xi_k, w_j, \sigma_j^2)/\partial w_j & \text{if } c(w_j) = y_k \\ -P(j|\xi_k) \cdot \text{const}_j \cdot \partial f(\xi_k, w_j, \sigma_j^2)/\partial w_j & \text{if } c(w_j) \neq y_k \end{cases}$$

$\alpha > 0$ is the learning rate. The probabilities are defined as

$$P_y(j|\xi_k) = \frac{P(j)\exp(f(\xi_k, w_j, \sigma_j^2))}{\sum_{c(w_j)=y_j} P(j)\exp(f(\xi_k, w_j, \sigma_j^2))}$$

and

$$P(j|\xi_k) = \frac{P(j)\exp(f(\xi_k, w_j, \sigma_j^2))}{\sum_j P(j)\exp(f(\xi_k, w_j, \sigma_j^2))} \,.$$

If the standard Euclidean distance is used, class priors are equal, and small bandwidth is present, a learning rule similar to LVQ2.1 results.

Given a novel data point ξ, its class label is the most likely label y corresponding to a maximum value $p(y|\xi, W) \sim p(\xi, y|W)$. For typical settings, this rule can be approximated by a simple winner takes all rule, i.e. ξ is mapped to the label $c(w_j)$ of the closest prototype w_j.

RSLVQ is restricted to Euclidean vectors. A kernelization of the method makes the technique applicable for more general data sets which are characterized in terms of a Gram matrix only. We assume that a kernel k is fixed corresponding to a feature map Φ. Then, the equation

$$k_{kl} := k(\xi_k, \xi_l) = \Phi(\xi_k)^t \Phi(\xi_l)$$

holds for all data points ξ_k, ξ_l. We assume that prototypes are represented by linear combinations of data

$$w_j = \sum_m \gamma_{jm} \Phi(\xi_m)$$

where the coefficients γ_{jm} are non-negative and sum up to 1. The cost function of RSLVQ becomes

$$L = \sum_k \log \frac{\sum_{c(w_j)=y_k} P(j) p(\Phi(\xi_k)|j)}{\sum_j P(j) p(\Phi(\xi_k)|j)} \,.$$

We assume equal bandwidth $\sigma^2 = \sigma_j^2$, for simplicity; more complex adjustment schemes based on the data have been investigated in [15], for example, usually leading to only a minor increase of accuracy. Note that the position of prototypes is not clear a priori, such that a prior adaptation of the bandwidth according to the data density is not possible. Further, we assume constant prior $P(j)$ and mixture components induced by normalized Gaussians. These can be computed in the data space based on the Gram matrix because of the identity

$$\|\Phi(\xi_i) - w_j\|^2 = \|\Phi(\xi_i) - \sum_m \gamma_{jm} \Phi(\xi_m)\|^2 = k_{ii} - 2 \cdot \sum_m \gamma_{jm} k_{im} + \sum_{s,t} \gamma_{js} \gamma_{jt} k_{st}$$

where the distance in the feature space is referred to by $\| \cdot \|^2$. Thus the update rules become $\Delta w_j = \sum_m \Delta \gamma_{jm} \Phi(\xi_m) =$

$$\alpha \cdot \text{const}_j \cdot \begin{cases} (P_y(j|\Phi(\xi_k)) - P(j|\Phi(\xi_k)))\,(\Phi(\xi_k) - \sum_m \gamma_{jm} \Phi(\xi_m)) & \text{if } c(w_j) = y_k \\ -P(j|\Phi(\xi_k))\,(\Phi(\xi_k) - \sum_m \gamma_{jm} \Phi(\xi_m)) & \text{if } c(w_j) \neq y_k \end{cases}$$

A stochastic gradient ascent yields the following adaptation rules for γ_{jm}:

$$\Delta \gamma_{jm} = \alpha \cdot \text{const}_j \cdot \begin{cases} -(P_y(j|\Phi(\xi_k)) - P(j|\Phi(\xi_k)))\gamma_{jm} & \text{if } \xi_m \neq \xi_k, c(w_j) = y_k \\ (P_y(j|\Phi(\xi_k)) - P(j|\Phi(\xi_k)))(1 - \gamma_{jm}) & \text{if } \xi_m = \xi_k, c(w_j) = y_k \\ P(j|\Phi(\xi_k))\gamma_{jm} & \text{if } \xi_m \neq \xi_k, c(w_j) \neq y_k \\ -P(j|\Phi(\xi_k))(1 - \gamma_{jm}) & \text{if } \xi_m = \xi_k, c(w_j) \neq y_k \end{cases}$$

This adaptation performs exactly the same updates as RSLVQ in the feature space if prototypes are in the convex hull of the data. To guarantee non-negativity and normalization, a correction takes place after every adaptation step. As an alternative, barrier techniques could be used, or the restrictions could be dropped entirely allowing more general linear combinations as solutions.

Note that, unlike RSLVQ, prototypes are represented implicitly in terms of linear combinations. The inspection of a prototype thus requires to inspect the coefficients representing the prototype γ_j and all data, the latter usually being characterized in terms of pairwise similarities only. Further, an adaptation step has squared complexity caused by the distributed representation of prototypes. Thus, the method does no longer directly give interpretable results, and it is no longer applicable for large data sets.

3 Nyström Approximation of the Gram Matrix

The Nyström technique has been presented in [18] in the context of SVMs. It allows to approximate a Gram matrix by a low rank approximation. For many kernel-based approaches, this approximation can be integrated into the learning rules in such a way that updates with linear complexity result. We shortly review the main idea behind this approach in the following.

By the Mercer theorem kernels $k(\xi_j, \xi_l)$ can be expanded by orthonormal eigenfunctions ϕ_i and non negative eigenvalues λ_i in the form $k(\xi_j, \xi_l) = \sum_{i=1}^{\infty} \lambda_i \phi_i(\xi_j) \phi_i(\xi_l)$. The eigenfunctions and eigenvalues of a kernel are the solution of an integral equation $\int k(\xi_j, \xi) \phi_i(\xi) p(\xi) d\xi = \lambda_i \phi_i(\xi_j)$ which can be approximated based on the Nyström technique by sampling ξ i.i.d. according to p, denoting the sampled values as $\xi_1, \ldots, \xi_m$ after possible reenumeration: $\frac{1}{m} \sum_{l=1}^{m} k(\xi_j, \xi_l) \phi_i(\xi_l) \approx \lambda_i \phi_i(\xi_j)$. We denote the submatrix corresponding to the m sampled points of the Gram matrix by $\mathbf{K}_{m,m}$. We denote eigenvalues and eigenvectors of this matrix by $\mathbf{U}^{(m)}$ and $\mathbf{\Lambda}^{(m)}$, respectively, characterized by the eigenvalue equation $\mathbf{K}_{m,m} \mathbf{U}^{(m)} = \mathbf{U}^{(m)} \mathbf{\Lambda}^{(m)}$. These solutions allow us to approximate the eigenfunctions and eigenvalues $\lambda_i \approx \frac{\lambda_i^{(m)}}{m}, \quad \phi_i(\xi_l) \approx \frac{\sqrt{m}}{\lambda_i^{(m)}} \mathbf{k}_{\xi_l} \mathbf{u}_i^{(m)}$ where $\mathbf{u}_i^{(m)}$ is the ith column of $\mathbf{U}^{(m)}$ and we use the vector of kernel values $\mathbf{k}_{\xi_l} = (k(\xi_1, \xi_l), ..., k(\xi_m, \xi_l))^T$.

This allows us to approximate a given full Gram matrix $\mathbf{K}$ by a low-rank counterpart, since we can use these approximations in the kernel expansion. Subsampling corresponds to a choice of m rows and columns of the matrix, the corresponding submatrix is denoted by $\mathbf{K}_{m,m}$ as before. The corresponding m rows and columns, respectively, are denoted by $\mathbf{K}_{m,n}$ and $\mathbf{K}_{n,m}$, respectively. These are transposes of each other, since the matrix is symmetric. The approximation as introduced above leads to the following approximation of the kernel expansion by orthonormal eigenfunctions $\tilde{\mathbf{K}} = \sum_{i=1}^{m} 1/\lambda_i^{(m)} \cdot \mathbf{K}_{n,m} \mathbf{u}_i^{(m)} (\mathbf{u}_i^{(m)})^T \mathbf{K}_{m,n}$ where $\lambda_i^{(m)}$ and $\mathbf{u}_i^{(m)}$ correspond to the $m \times m$ eigenproblem as above. In the case that some $\lambda_i^{(m)}$ are zero, we replace the corresponding fractions with zero. Thus we get, $\mathbf{K}_{m,m}^{-1}$ denoting the Moore-Penrose Pseudoinverse,

$$\tilde{\mathbf{K}} = \mathbf{K}_{n,m} \mathbf{K}_{m,m}^{-1} \mathbf{K}_{m,n} \,.$$

For a given matrix $\mathbf{K}$ with rank m, this approximation is exact, if the m chosen m-dimensional points are linearly independent.

This observation allows us to approximate the full Gram matrix as used in kernel RSLVQ by a low rank approximation: this equation for $\tilde{\mathbf{K}}$ can directly be integrated into the computation of the Gaussians using the identity

$$\|\Phi(\xi_i) - w_j\|^2 = \mathbf{e}_i^t \mathbf{K} \mathbf{e}_i - 2 \cdot \mathbf{e}_i^t \mathbf{K} \gamma_j + \gamma_j^t \mathbf{K} \gamma_j$$

where $\mathbf{e}_i$ denotes the i th unit vector. Using $\tilde{\mathbf{K}}$ instead, linear complexity results if the matrix vector multiplications are computed first.

4 *k*-Approximation of the Prototypes

Kernel RSLVQ yields prototypes which are implicitly represented as linear combinations of data points

$$w_j = \sum_m \gamma_{jm} \Phi(\xi_m) \, .$$

Since the training algorithm and classification depends on pairwise distances only, simple linear algebra allows us to compute the distance of a data point and a prototype based on the pairwise similarity of the data point and all training data only, i.e. the given Gram matrix, as specified above. However, direct interpretability and sparseness of the prototype is lost this way.

Here we propose to use a simple approximation of the prototypes to maintain interpretability and flexibility of the clustering. As already proposed in the context of relational approaches for streaming data, a prototype is approximated by the k nearest exemplars in the given training set [6]. These exemplars can easily be computing as follows: we determine the distance of the prototype and the data points and choose the respective k nearest exemplars based on these values. This way, a small number of exemplars represent the classifier by means of a nearest neighbor classification rule. Since the exemplars can be inspected in the same way as data points, this allows insights into the model by experts in the field. Note that it would be reasonable to choose an adaptive number of exemplars to represent a prototype instead of a fixed number k depending on the resulting classification results; this will be the subject of future research.

5 Experiments

We compare RSLVQ and its Nyström and k-approximation, respectively, with different values of k on a variety of benchmarks as introduced in [2]. For the Nyström approximation, we use a subsample of 10% in all cases. The data sets consist of similarity matrices which are, in general, non-Euclidean. Non-Euclideanity can be quantified by the signature of the data set, i.e. the number of positive, negative, and zero eigenvalues of the similarity matrix. The matrices are symmetrized and normalized before processing. In general, the given similarity matrices do not constitute a valid kernel such that a probabilistic representation using the above formulas is no longer well-defined due to potentially negative distances. There exist standard preprocessing tools which transfer a given similarity matrix into a valid kernel, as presented e.g. in [2,11]. Typical corrections are:

- *Spectrum clip:* set negative eigenvalues of the matrix to 0. This can be realized as a linear projection and directly transfers to out-of-sample extensions.
- *Spectrum flip:* negative eigenvalues are substituted by their positive values. Again, this can be realized by means of a linear transformation.

These transforms which turn a given similarity matrix into a valid Gram matrix are tested for kernel RSLVQ with according preprocessing. We use training data in analogy to [2].

- *Amazon47*: This data set consists of 204 books written by 47 different authors. The similarity is determined as the percentage of customers who purchase book j after looking at book i. The signature is $(192, 12, 0)$. The class label of a book is given by the author.
- *Aural Sonar*: This data set consists of 100 wide band solar signals corresponding to two classes, observations of interest versus clutter. Similarities are determined based on human perception, averaging over 2 random probands for each signal pair. The signature is $(62, 38, 0)$. Class labeling is given by the two classes: target of interest versus clutter.
- *Face Rec*: 945 images of faces of 139 different persons are recorded. Images are compared using the cosine-distance of integral invariant signatures based on surface curves of the 3D faces. The signature is $(794, 151, 0)$. The labeling corresponds to the 139 different persons.
- *Patrol*: 241 samples representing persons in seven different patrol units are contained in this data set. Similarities are based on responses of persons in the units about other members of their groups. The signature is $(117, 124, 0)$. Class labeling corresponds to the seven patrol units.
- *Protein*: 213 proteins are compared based on evolutionary distances comprising four different classes according to different globin families. The signature is $(171, 42, 0)$. Labeling is given by four classes corresponding to different globin families.
- *Voting*: Voting contains 435 samples with categorical data compared by means of the value difference metric. Class labeling into two classes is present. The signature is $(226, 209, 0)$.

Note that the rank of the Gram matrix is given by the number of positive eigenvalues if clip is used as preprocessing, and the sum of non-negative eigenvalues if the original data or flip are used.

Prototypes are initialized by means of normalized random coefficients γ_{jm}. Coefficient m is set to zero if the label of point ξ_m does not coincide with the prototype label $c(w_j)$. The number of prototypes is taken as a small multiple of the number of classes. Other meta-parameters are optimized on the data sets using cross-validation. The results for RSLVQ and its Nyström approximation are reported in Tab. 1. Classification accuracy is thereby evaluated in a 20-fold cross-validation. Note that a decomposition of a data set characterized by a similarity matrix into training and test set corresponds to a selection of a set of indices I. The submatrix formed by $(k_{ij})_{i,j \in I}$ characterizes the training set, distances of prototypes to test points for a classification of the test set can be computed based on $(k_{ij})_{i \in I, j \notin I}$. Similar experiments show the performance of a sparse approximation of the result using different numbers k to represent a prototype by k exemplars in Tab. 2.

Table 1. Results of kernel RSLVQ and a Nyström approximation of the Gram matrix using 10% of the data. The mean classification error and standard deviation obtained in a 20-fold cross-validation are reported. Results for k-NN and SVM are taken from [2].

	k-NN	SVM	kernel RSLVQ	Nyström	prototypes
Amazon47	16.95 (4.85)	75.98 (7.33)	15.37 (0.36)	64.15 (0.81)	94
clip	17.68 (4.75)	81.34 (4.77)	15.37 (0.41)	64.15 (0.33)	
flip	17.56 (4.91)	84.27 (4.33)	16.34 (0.42)	65.73 (0.30)	
Aural Sonar	17.00 (7.65)	14.25 (7.46)	11.50 (0.37)	21.25 (2.05)	10
clip	14.00 (6.82)	13.00 (5.34)	11.25 (0.39)	15.00 (0.63)	
flip	12.75 (6.42)	13.25 (5.31)	11.75 (0.35)	16.25 (0.84)	
Face Rec	4.23 (1.43)	3.92 (1.29)	3.78 (0.02)	3.52 (0.02)	139
clip	4.15 (1.32)	4.18 (1.25)	3.84 (0.02)	3.47 (0.02)	
flip	4.15 (1.32)	4.18 (1.32)	3.60 (0.02)	3.52 (0.02)	
Patrol	11.88 (4.42)	40.73 (5.95)	17.50 (0.25)	61.77 (0.63)	24
clip	11.56 (4.54)	38.75 (4.81)	17.40 (0.29)	47.50 (0.78)	
flip	11.67 (4.24)	47.29 (5.90)	19.48 (0.34)	45.94 (0.66)	
Protein	29.88 (9.96)	2.67 (2.97)	26.98 (0.37)	28.60 (1.63)	20
clip	30.35 (9.71)	5.35 (4.60)	4.88 (0.17)	12.21 (0.36)	
flip	31.28 (9.63)	1.51 (2.36)	1.40 (0.05)	8.02 (0.38)	
Voting	5.80 (1.83)	5.52 (1.77)	5.46 (0.04)	5.23 (0.04)	20
clip	5.29 (1.80)	4.89 (2.05)	5.34 (0.04)	5.17 (0.03)	
flip	5.23 (1.80)	4.94 (2.03)	5.34 (0.03)	5.34 (0.04)	

Table 2. Results of kernel RSLVQ and its k-approximation for $k \in \{1, \ldots, 4\}$. The classification error in % and standard deviation in parenthesis are given.

	kernel RSLVQ	1-approx	2-approx	3-approx	4-approx
Amazon47	15.37 (0.36)	36.83 (0.35)	29.27 (0.42)	29.65 (0.53)	30.97 (0.44)
clip	15.37 (0.41)	31.65 (0.26)	26.29 (0.48)	28.06 (0.43)	29.96 (0.42)
flip	16.34 (0.42)	31.28 (0.25)	28.91 (0.35)	29.85 (0.39)	30.95 (0.40)
Aural Sonar	11.50 (0.37)	25.13 (1.15)	20.62 (1.56)	21.62 (0.96)	21.62 (0.79)
clip	11.25 (0.39)	24.75 (0.78)	21.75 (1.06)	18.50 (0.66)	17.00 (0.83)
flip	11.75 (0.35)	24.75 (0.99)	21.50 (0.48)	21.00 (0.57)	21.62 (0.53)
Face Rec	3.78 (0.02)	3.70 (0.02)	5.92 (0.02)	8.99 (0.03)	11.70 (0.04)
clip	3.84 (0.02)	3.76 (0.02)	6.00 (0.02)	8.95 (0.03)	11.90 (0.04)
flip	3.60 (0.02)	3.33 (0.02)	5.64 (0.03)	8.58 (0.04)	11.57 (0.03)
Patrol	17.50 (0.25)	54.94 (0.96)	46.69 (1.02)	39.33 (0.77)	35.46 (0.49)
clip	17.40 (0.29)	32.46 (0.90)	21.89 (0.34)	22.36 (0.43)	20.28 (0.31)
flip	19.48 (0.34)	37.42 (0.85)	26.36 (0.45)	22.27 (0.25)	21.96 (0.27)
Protein	26.98 (0.37)	55.12 (0.67)	49.97 (0.77)	49.57 (0.75)	47.38 (0.92)
clip	4.88 (0.17)	22.44 (0.51)	25.81 (0.98)	28.20 (0.92)	29.42 (0.89)
flip	1.40 (0.05)	23.26 (0.26)	22.77 (0.34)	22.56 (0.47)	23.37 (0.52)
Voting	5.46 (0.04)	8.56 (0.06)	8.71 (0.07)	8.59 (0.07)	8.56 (0.05)
clip	5.34 (0.04)	8.65 (0.07)	9.22 (0.09)	9.08 (0.09)	8.82 (0.09)
flip	5.34 (0.03)	7.84 (0.04)	7.82 (0.03)	8.13 (0.03)	8.56 (0.04)

The results obtained with kernel RSLVQ are generally good and reach state-of-the art accuracy as reported in [2]. In general, preprocessing using spectrum clip or flip can be beneficial. Surprisingly, a naive application of kernel RSLVQ for the (non-euclidean) similarity matrix already yields surprisingly good results. The results of a linear time Nyström approximation as well as a k-approximation are heterogeneous.

While the classification accuracy obtained for the data sets Aural Sonar, Face Rec, Protein, and Voting using a Nyström approximation are still acceptable, they are considerably worse for Amazon47 and Patrol. Similarly, for some of the data sets, the k-approximation is generally acceptable and yields results comparable to kernel RSLVQ itself (Voting). For others, the 1-approximation yields worse results with a decrease of the accuracy by more than 10%, but a sufficient number k yields acceptable results (Aural Sonar, Patrol). Interestingly, there is also the opposite case, a 1-approximation yielding acceptable results, but larger values k leading to more than 10% loss of accuracy (FaceRec). For Amazon47 and Protein, the classification results of all approximations are significantly worse as compared to direct kernel RSLVQ.

In consequence, it is worthwhile to consider these approximations for some data sets, leading to greatly enhanced sparsity and computational performance, respectively, but it is not clear a priori whether the good classification accuracy of kernel RSLVQ can be preserved. Reasons for this behavior can be manifold, such as the intrinsic rank of the Gram matrix, the number of classes, etc. Note that the Nyström approximation acts in two ways, which makes a prior prediction of the result complicated:

- The Nyström approximation substitutes the original Gram matrix by a low rank approximation. If the rank is kept (because the original one has small rank or many small eigenvalues), Nyström does not lead to loss of information and it is guaranteed that the approximation is exact.
- On the other hand, the Nyström approximation can help to suppress noise which is not relevant for the classification task in a similar way as clip can have a beneficial effect.

6 Discussion

We have investigated kernel robust soft LVQ and the possibility to obtain efficient approximations by means of a Nyström approximation and k-approximation, respectively. These approximations aim at an improved computational performance of the technique or an improved sparsity of the classifier, respectively. While kernel RSLVQ generally yields very good results comparable to SVM, the situation is less clear for the approximations. In some cases, a high classification accuracy is maintained, while the classification accuracy is decreased by more than 15% for others. It is the subject of ongoing work to investigate formal properties of the Gram matrix such as its rank or geometric properties which allow to judge the suitability of the approximations in advance. In [19], it is proposed to evaluate the approximation quality based on the Pearson correlation, and the evaluation result is used to predict the performance of the Nyström approximation for unsupervised classification tasks. Currently, we are investigating this possibility in the context of classification.Archive Further, it is subject of ongoing work to test the combined effect of these approximations.

Acknowledgement. This work has been supported by the DFG under grant number HA2719/7-1 and by the CITEC center of excellence.

References

1. Boulet, R., Jouve, B., Rossi, F., Villa, N.: Batch kernel SOM and related Laplacian methods for social network analysis. Neurocomputing 71(7-9), 1257–1273 (2008)
2. Chen, Y., Garcia, E.K., Gupta, M.R., Rahimi, A., Cazzanti, L.: Similarity-based classification: Concepts and algorithms. JMLR 10, 747–776 (2009)
3. Cottrell, M., Hammer, B., Hasenfuss, A., Villmann, T.: Batch and median neural gas. Neural Networks 19, 762–771 (2006)
4. Frasconi, P., Gori, M., Sperduti, A.: A general framework for adaptive processing of data structures. IEEE TNN 9(5), 768–786 (1998)
5. Gärtner, T.: Kernels for Structured Data. PhD thesis, Univ. Bonn (2005)
6. Hammer, B., Hasenfuss, A.: Topographic mapping of large dissimilarity datasets. Neural Computation 22(9), 2229–2284 (2010)
7. Hofmann, D., Hammer, B.: Kernel Robust Soft Learning Vector Quantization. In: Mana, N., Schwenker, F., Trentin, E. (eds.) ANNPR 2012. LNCS, vol. 7477, pp. 14–23. Springer, Heidelberg (2012)
8. Kirstein, S., Wersing, H., Gross, H.-M., Körner, E.: A life-long learning vector quantization approach for interactive learning of multiple categories. Neural Networks 28, 90–105 (2012)
9. Kohonen, T.: Self-Oganizing Maps, 3rd edn. Springer (2000)
10. Kohonen, T., Somervuo, P.: How to make large self-organizing maps for nonvectorial data. Neural Networks 15(8-9), 945–952 (2002)
11. Pekalska, E., Duin, R.P.: The Dissimilarity Representation for Pattern Recognition. Foundations and Applications. World Scientific (2005)
12. Qin, A.K., Suganthan, P.N.: A novel kernel prototype-based learning algorithm. In: Proc. of the 17th International Conference on Pattern Recognition (ICPR 2004), Cambridge, UK (August 2004)
13. Sato, A., Yamada, K.: Generalized Learning Vector Quantization. In: NIPS (1995)
14. Schneider, P., Biehl, M., Hammer, B.: Distance learning in discriminative vector quantization. Neural Computation 21, 2942–2969 (2009)
15. Schneider, P., Biehl, M., Hammer, B.: Hyperparameter learning in probabilistic prototype-based models. Neuromputing 73(7-9), 1117–1124 (2009)
16. Seo, S., Obermayer, K.: Soft learning vector quantization. Neural Computation 15, 1589–1604 (2003)
17. Vellido, A., Martin-Guerroro, J.D., Lisboa, P.: Making machine learning models interpretable. In: ESANN 2012 (2012)
18. Williams, C.K.I., Seeger, M.: Using the Nyström method to speed up kernel machines. Advances in Neural Information Processing Systems 13, 682–688 (2001)
19. Zhu, X., Gisbrecht, A., Schleif, F.-M., Hammer, B.: Approximation techniques for clustering dissimilarity data. Neurocomputing 90, 72–84 (2012)

Gradient Based Learning in Vector Quantization Using Differentiable Kernels

Thomas Villmann*, Sven Haase, and Marika Kästner

Computational Intelligence Group,
University of Applied Sciences Mittweida, 09648 Mittweida, Germany
thomas.villmann@hs-mittweida.de, {villmann,haase,kaestner}@hs-mittweida.de

Abstract. Supervised and unsupervised prototype based vector quantization frequently are proceeded in the Euclidean space. In the last years, also non-standard metrics became popular. For classification by support vector machines, Hilbert space representations are very successful based on so-called kernel metrics. In this paper we give the mathematical justification that gradient based learning in prototype-based vector quantization is possible by means of kernel metrics instead of the standard Euclidean distance. We will show that an appropriate handling requires *differentiable universal kernels* defining the kernel metric. This allows a prototype adaptation in the original data space but equipped with a metric determined by the kernel. This approach avoids the Hilbert space representation as known for support vector machines. Moreover, we give prominent examples for differentiable universal kernels based on information theoretic concepts and show exemplary applications.

1 Introduction

Prototype based vector quantization is an ongoing topic of research with applications in unsupervised and supervised data modeling. Famous unsupervised models applied in data clustering or visualization are the self-organizing map (SOM,[21]), neural gas (NG, [26]) or respective fuzzy variants like fuzzy-c-means (FCM, [3,4]). Supervised approaches comprise the family of learning vector quantizers (LVQ, [21]) as well as support vector machines (SVM,[41]). LVQ models generate class typical prototypes whereas SVMs determine prototypes (support vectors) defining the class borders. Both paradigms are margin classifiers [11]. Recent developments in the field address the utilization of non-standard metrics to improve the model performance for domain specific problems like processing of functional data, e.g. spectra, time series, etc. [20,29,47], or better interpretability of the adapted models (relevance/matrix learning, [16,42]).

One of the most challenging ideas in classification learning is the kernel trick realized in SVMs. According to this idea, the data as well as the prototypes are implicitly mapped into a high-dimensional (infinite) feature mapping Hilbert space (FMHS) uniquely determined by the kernel, but the dissimilarities still

[*] Corresponding author.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 193–204.
DOI: 10.1007/978-3-642-35230-0_20

are calculated using the original data whereas model adaptation is processed in the dual space of the FMHS. This implicit mapping frequently offers a great flexibility and good separation possibility. This advantage, however, makes it more difficult to interpret the model because the prototypes in these models are given as infinite-dimensional representations in the FMHS. Moreover, the support vectors are not typical representatives of the classes, as mentioned before. Several variants of LVQ were established also integrating the kernel mapping concept in those models to keep the idea of class-typical prototypes (Kernel GLVQ, KGLVQ) [35,34]. Yet, these models also have to the infinity of the mapping space. Usually, it is approximated by a finite one using the Nystrøm-approximation technique [40], which obviously leads to an information loss in general.

In this paper we offer an alternative for the integration of kernels in prototype based vector quantization. For this purpose, we consider *differentiable* universal kernels determining a new differentiable metric in the data space to be used in the vector quantization model. Thus gradient based learning becomes available whereby the topological structure of the new metric space is isomorphic to the FMHS.

The paper is structured as follows: First we briefly review the idea and justification of kernel mapping into FHMS. Thereafter, we present the theoretical justification of the differentiable kernel online vector quantization approach. Subsequently, we present information theoretic kernels. Sample applications and concluding remarks complete the contribution.

2 Reproducing Kernels for Hilbert Spaces and Kernel Mapping

We start with a brief review of the kernel theory. For that we assume the data space as a compact metric space (V, d_V), i.e. a vector space V equipped with a metric d_V. A function

$$\kappa_\Phi : \ V \times V \to \mathbb{C} \tag{1}$$

is a kernel, if there exists a *Hilbert space* $\mathcal{H}$ and a map

$$\Phi : V \ni \mathbf{v} \longmapsto \Phi(\mathbf{v}) \in \mathcal{H} \tag{2}$$

with

$$\kappa_\Phi(\mathbf{v}, \mathbf{w}) = \langle \Phi(\mathbf{v}), \Phi(\mathbf{w}) \rangle_{\mathcal{H}} \tag{3}$$

for all $\mathbf{v}, \mathbf{w} \in V$ and $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ is the inner product of this Hilbert space. As a consequence the kernel is Hermitian, i.e. $\kappa_\Phi(\mathbf{v}, \mathbf{w}) = \overline{\kappa_\Phi(\mathbf{w}, \mathbf{v})}$ and, therefore, sesquilinear. The mapping Φ is called feature map and $\mathcal{H}$ the feature space of V. Without further restrictions on the kernel κ_Φ both $\mathcal{H}$ and Φ are not unique. A function $f : V \longrightarrow \mathbb{C}$ is *induced by* κ_Φ if there exists an element $g \in \mathcal{H}$ with $f(\mathbf{w}) = \langle g, \Phi(\mathbf{w}) \rangle_{\mathcal{H}}$. The following important Lemma is shown in [46]:

Lemma 1. *Let κ_Φ be a kernel of a metric space (V, d_V) and Φ a corresponding feature map into a Hilbert space $\mathcal{H}$. Then κ_Φ is continuous iff Φ does. In this case*

$$d_{\kappa_\Phi}(\mathbf{v}, \mathbf{w}) = \|\Phi(\mathbf{v}) - \Phi(\mathbf{w})\|_{\mathcal{H}} \tag{4}$$

defines a semi-metric[1] *on V and the identity map Ψ between the different metric spaces over the vector space V*

$$\Psi : (V, d_V) \longrightarrow (V, d_{\kappa_\Phi}) \tag{5}$$

is continuous. If the feature map Φ is injective d_{κ_Φ} is even a metric.

We have to to state the following important remark:

Remark 1. In the proof of this lemma the inner product property (3) of the kernel is never used. Only the norm properties of Hilbert spaces and their completeness are required. Hence, the lemma is also valid if Φ would map into a Banach space $\mathcal{B}$ with metric d_{κ_Φ}.

To ensure the separability of the feature map Φ the kernel has to be *universal* [46]. Further, Steinwart has also proofed that continuous universal kernel imply the injectivity of the corresponding feature map Φ. Again, we have to emphasize that the proof of this theorem does not utilize the inner product property (3) of the kernel. Only, the semi-metric properties of the corresponding metric are needed, which would remain valid also regarding Banach spaces instead of Hilbert spaces.

An important role in feature mapping play positive definite kernels, which *uniquely* correspond to Hilbert spaces $\mathcal{H}$ in a canonical manner according to the Mercer-theorem [1,27]. The kernel κ_Φ is said to be (strictly) positive definite if for all finite subsets $V_n \subseteq V$ with cardinality $\#V_n = n$, the Gram-Matrix

$$\mathbf{G}_n = [\kappa(\mathbf{v}_i, \mathbf{v}_j) : i, j = 1 \ldots n] \tag{6}$$

is (strictly) positive semi-definite [1]. In that case, the Hilbert space $\mathcal{H}$ is a so-called *reproducing kernel Hilbert space* (RKHS), i.e. the kernel function $\kappa_\Phi(\mathbf{v}, \cdot) \in \mathcal{H}$ and for each $\mathbf{v} \in V$ and all $f \in \mathcal{H}$ and $\mathbf{w} \in V$ the relation $f(\mathbf{w}) = \langle f, \kappa_\Phi(\mathbf{w}, \cdot)\rangle_{\mathcal{H}}$ is valid according to the Riesz representation theorem [1,22]. Here, κ_Φ is denoted as a *reproducing kernel* obviously being symmetric, real and, hence, bi-linear. The space $\mathcal{I}_{\kappa_\Phi}$ of kernel induced functions is given as the set

$$\mathcal{I}_{\kappa_\Phi} = \{\kappa_\Phi(\mathbf{w}, \cdot) | \mathbf{w} \in V\} \tag{7}$$

with $\mathcal{I}_{\kappa_\Phi} \subseteq \mathcal{H}$. For positive kernels the associated inner product implies a norm $\|\Phi(\mathbf{v})\|_{\mathcal{H}} = \sqrt{\langle \Phi(\mathbf{v}), \Phi(\mathbf{v})\rangle_{\mathcal{H}}}$ and, hence, also a metric $d_{\mathcal{H}}(\Phi(\mathbf{v}), \Phi(\mathbf{w})) = \|\Phi(\mathbf{v}) - \Phi(\mathbf{w})\|_{\mathcal{H}}$. Hence, the positive semi-definiteness of the kernel ensures the metric properties in comparison to the semi-metric (4) obtained for general kernels. Because κ_Φ is a kernel, the metric $d_{\mathcal{H}}(\Phi(\mathbf{v}), \Phi(\mathbf{w}))$ can be rewritten as

$$d_{\mathcal{H}}(\Phi(\mathbf{v}), \Phi(\mathbf{w})) = \sqrt{\kappa_\Phi(\mathbf{v}, \mathbf{v}) - 2\kappa_\Phi(\mathbf{v}, \mathbf{w}) + \kappa_\Phi(\mathbf{w}, \mathbf{w})} \tag{8}$$

using the bi-linearity and the symmetry of the positive kernel.

[1] Note, for a semi-metric the triangle inequality does not hold [32].

Remark 2. Obviously, the semi-metric d_{κ_Φ} from (4) coincides with $d_{\mathcal{H}}$ on $\mathcal{I}_{\kappa_\Phi}$ for positive kernels.

This last remark allows an important conclusion regarding the mapping Ψ from (5) in relation to a given positive continuous kernel κ_Φ:

Lemma 2. *Let (V, d_V) be a compact metric space, $\kappa_\Phi : V \times V \to \mathbb{R}$ a continuous positive kernel with the feature map $\Phi : V \longrightarrow \mathcal{H}$, and the kernel determining a metric $d_{\mathcal{H}}$ in $\mathcal{H}$ by (8). If the space of the induced functions $\mathcal{I}_{\kappa_\Phi}$ is dense in the space of continuous functions $\mathcal{C}(V)$, then the metric space $(V, d_{\mathcal{H}})$ is topologically equivalent to induced space $\mathcal{I}_{\kappa_\Phi} \subseteq \mathcal{H}$ with the metric $d_{\mathcal{H}}$. Moreover, both spaces are isometric, and, hence, $(V, d_{\mathcal{H}})$ is a Hilbert space, too. In consequence, the generally non-linear mapping Ψ from (5) is an bijective, separable and continuous mapping. The result of the Lemma 2 is visualized in Fig.1.*

$$\begin{tikzcd}
(V, d_V) \arrow[r, "\Phi, \kappa_\Phi"] \arrow[d, "\Psi"'] & \mathcal{I}_{\kappa_\Phi} \subseteq (\mathcal{H}, d_{\mathcal{H}}) \\
(V, d_{\mathcal{H}}) \arrow[ur, leftrightarrow, "\Phi \circ \Psi^{-1}"'] &
\end{tikzcd}$$

Fig. 1. Visualization of Lemma 2: For universal kernels κ_Φ the metric spaces $(V, d_{\mathcal{H}})$ and $\left(\mathcal{I}_{\kappa_\oplus}, d_{\mathcal{H}}\right)$ are topologically equivalent and isometric by means of the continuous bijective mapping $\Phi \circ \Psi^{-1}$

Proof. The kernel κ_Φ is assumed to be positive, continuous and generating a space of induced functions $\mathcal{I}_{\kappa_\Phi}$, which is dense in the space of continuous functions $\mathcal{C}(V)$. Hence, κ_Φ is universal and, therefore, the uniquely corresponding feature map $\Phi : V \longrightarrow \mathcal{H}$ is injective according to [46]. Hence, it is bijective for $\Phi : V \longrightarrow \mathcal{I}_{\kappa_\Phi} \subseteq \mathcal{H}$, whereby $\mathcal{H}$ is equipped with the Hilbert space metric $d_{\mathcal{H}}$. Because (V, d_V) is compact and the bijective mapping Φ is continuous, it follows immediately that $\mathcal{I}_{\kappa_\Phi}$ is a subspace of $\mathcal{H}$ and, therefore, a Hilbert space itself. Moreover, it follows from Lemma 1 that Φ is also continuous as well as the obviously bijective identity map $\Psi : (V, d_V) \longrightarrow (V, d_{\mathcal{H}})$ from (5). Hence, the map $\Phi\left(\Psi^{-1}(\mathbf{v})\right) = \Phi \circ \Psi^{-1}(\mathbf{v})$ with $\mathbf{v} \in (V, d_{\mathcal{H}})$ is bijective and continuous. Therefore, $(V, d_{\mathcal{H}})$ and $\mathcal{I}_{\kappa_\Phi}$ are isomorphic and, according to the Remark 2, also isometric. The separability of Ψ follows immediately from the separability property of Φ. ■

It is well known that the Gaussian kernel $\kappa_\Phi(\mathbf{u}, \mathbf{v}) = \exp\left(\frac{-\|\mathbf{u}-\mathbf{v}\|_E^2}{2\sigma^2}\right)$, the Student-type Gaussian kernel $\kappa_\Phi(\mathbf{u}, \mathbf{v}) = \left(\beta + \frac{\|\mathbf{u}-\mathbf{v}\|_E^2}{\sigma^2}\right)^{-\alpha}$ with $\alpha, \beta > 0$ and the exponential kernel $\kappa_{\Phi}(\mathbf{u},\mathbf{v}) = \exp\left(\langle \mathbf{u},\mathbf{v}\rangle_E\right)$ are universal on every compact subset of $\mathbb{R}^n$. Other universal kernel can be found in [28,43,46]. At this point we remark that these kernels are also differentiable, which becomes important in Sect. 4.

Another class of kernels are *information theoretic kernels* based on divergences [25,33]. This class is investigated in the light of universality in the next subsection. The relation of universal kernels to *characteristic kernels* is addressed in [45].

3 Universal Kernels Based on Divergences

Information theoretic kernels based on divergences are considered in many applications [8,23,25,33]. Here we relate them to universal differentiable kernels, such that the diagram in Fig.1 holds also for those kernels. For this purpose, we introduce the class of *radial kernels* $\kappa_r : \mathbb{R}^m \times \mathbb{R}^m \longrightarrow \mathbb{R}$ [19,41,43]. These kernels are defined as

$$\kappa_r(\mathbf{u},\mathbf{v}) = g\left(d(\mathbf{u},\mathbf{v})\right) \tag{9}$$

where $d(\mathbf{u},\mathbf{v})$ is a metric and g is a function on $\mathbb{R}_0^+ = \{x \in \mathbb{R} | x \geq 0\}$. Equivalently, $d(\mathbf{u},\mathbf{v})$ could be a norm of the difference $(\mathbf{u}-\mathbf{v})$. One important point to be emphasized here is that the argument of a radial kernel is required to be a metric or, equivalently, a norm. Radial kernels stand out due to its close relation to universal kernels. The following lemma holds for radial kernels [45]:

Lemma 3. *If the radial kernel is strictly positive definite then it is also universal.*

If we want to obtain a differentiable universal kernel based on divergences, we have, hence, to ensure that the divergence is differentiable, metric, and that the corresponding radial kernel is positive definite. Generally, divergences are not symmetric and, therefore, cannot serve as a metric [9,10,14]. Yet, there exist some special divergences for vectorized data, which are metrics at the same time under the assumption that the data vectors represent probability densities or at least positive functions [47]. For example, the Euclidean distance is a so-called η-divergence belonging to the class of Bregman-divergences with parameter $\eta = 2$ [30]. Österreicher and Vajda considered a subset of Csiszár's f-divergences to be metric [31,47]. To this class belongs the subclass of f_β-divergences, a prominent member of which is the squared *Hellinger distance*

$$D_H(\mathbf{u}\|\mathbf{v}) = \sum_{i=1}^{m} \left(\sqrt{u_i} - \sqrt{v_i}\right)^2 \tag{10}$$

obtained for the value $\beta = \frac{1}{2}$. Another example is the *Jensen-Shannon-divergence*

$$D_{JS}(\mathbf{u}\|\mathbf{v}) = \frac{D_{KL}(\mathbf{u}\|\mathbf{w}) + D_{KL}(\mathbf{v}\|\mathbf{w})}{2} \tag{11}$$

obtained for $\beta = 1$ with $\mathbf{w} = \frac{\mathbf{u}+\mathbf{v}}{2}$ and

$$D_{KL}\left(\mathbf{u}\|\mathbf{w}\right) = \sum_{i=1}^{m} u_i \log \frac{u_i}{v_i} \tag{12}$$

being the *Kullback-Leibler-divergence* [24]. It can be calculated based on the *Shannon-entropy*

$$H\left(\mathbf{v}\right) = -\sum_{i=1}^{m} v_i \log v_i \tag{13}$$

as

$$D_{JS}\left(\mathbf{u}\|\mathbf{v}\right) = H\left(\frac{\mathbf{u}+\mathbf{v}}{2}\right) - \left(\frac{H\left(\mathbf{u}\right) + H\left(\mathbf{v}\right)}{2}\right) \tag{14}$$

as shown in [25,44].

An analog divergence can be installed using the *Rényis α-entropy*

$$H_{\alpha}\left(\mathbf{v}\right) = \frac{1}{1-\alpha} \log\left(\sum_{i=1}^{m} \left(v_i\right)^{\alpha}\right) \tag{15}$$

defined for $\alpha > 0$ [36,37]. In the limit $\alpha \to 1$ $H_{\alpha}\left(\mathbf{v}\right)$ converges to the Shannon-entropy $H\left(\mathbf{v}\right)$ from (13). Based on the Rényi-entropy (15) the *Jensen-Rényi-α-divergence* is defined as

$$D^{\alpha}_{JR}\left(\mathbf{u}\|\mathbf{v}\right) = H_{\alpha}\left(\frac{\mathbf{u}+\mathbf{v}}{2}\right) - \left(\frac{H_{\alpha}\left(\mathbf{u}\right) + H_{\alpha}\left(\mathbf{v}\right)}{2}\right) \tag{16}$$

in complete analogy to (14) [2]. It turns out that both, $\sqrt{D_{JS}\left(\mathbf{u}\|\mathbf{v}\right)}$ and $\sqrt{D^{\alpha}_{JR}\left(\mathbf{u}\|\mathbf{v}\right)}$, are metrics [25] or, more precisely, they are Hilbertian metrics [17]. Moreover it is shown in the paper [25] by Martin et al. that the following lemma holds:

Lemma 4. *The kernels*

1. $\kappa^1_{JS}\left(\mathbf{u},\mathbf{v}\right) = \exp\left(-t \cdot D_{JS}\left(\mathbf{u}\|\mathbf{v}\right)\right)$, $t > 0$,
2. $\kappa^1_{JR}\left(\mathbf{u},\mathbf{v},\alpha\right) = \exp\left(-t \cdot D^{\alpha}_{JR}\left(\mathbf{u}\|\mathbf{v}\right)\right)$, $t > 0$,
3. $\kappa^2_{JS}\left(\mathbf{u},\mathbf{v}\right) = \left(t + D_{JS}\left(\mathbf{u}\|\mathbf{v}\right)\right)^{-1}$, $t > 0$ *and*
4. $\kappa^2_{JR}\left(\mathbf{u},\mathbf{v},\alpha\right) = \left(t + D^{\alpha}_{JR}\left(\mathbf{u}\|\mathbf{v}\right)\right)^{-1}$, $t > 0$

are strictly positive definite. For the kernels κ^1_{JR} and κ^2_{JR} the additional condition of $\alpha \in [0,1]$ has to be fulfilled for positive definiteness.

Therefore, we can finally state the following corollary for divergence based kernels:

Corollary 1. *The kernels given in Lemma 4 based on the Jensen-Shannon-divergence (14) and the Jensen-Rényi-α-divergence (16) are universal.*

Proof. This property follows immediately from Lemma 4 together with the Lemma 3. ∎

Last but not least we remark again that the kernels defined in Lemma 4 are differentiable [47], which relates them to the considerations in Sect. 4.

4 Differentiable Kernel and Gradient Based Vector Quantization

Vector quantization can be distinguished into unsupervised and supervised approaches. The main task for unsupervised models is to minimize some reconstruction error E for a given data set $V \subseteq \mathbb{R}^n$ of vectors $\mathbf{v}$ with respect to set of prototypes $W = \{\mathbf{w}_k\}_{k\in A}$, where A is a finite index set. Prominent examples are the self-organizing map (SOM,[21]), neural gas (NG, [26]), whereby for the SOM the variant of HESKES is taken [18]. For those models, the reconstruction error is given in terms of the dissimilarity measure $d(\mathbf{v}, \mathbf{w}_k)$ between data and prototypes, which is assumed to be differentiable. Adaptation for these models is frequently realized as a stochastic gradient descent. In that case, the gradient $\partial E/\partial \mathbf{w}_k$ contains the derivative $\partial d(\mathbf{v}, \mathbf{w}_k)/\partial \mathbf{w}_k$ originating from the chain rule of differentiation. For example, the cost function of the Heskes variant of SOM is

$$E_{\mathrm{SOM}} = \int P(\mathbf{v}) \sum_{\mathbf{r}\in A} \delta_{\mathbf{r}}^{s(\mathbf{v})} \sum_{\mathbf{r}'\in A} \frac{h_\sigma^{SOM}(\mathbf{r},\mathbf{r}')}{2K(\sigma)} d(\mathbf{v}, \mathbf{w}_{\mathbf{r}'}) d\mathbf{v} \tag{17}$$

with the so-called neighborhood function $h_\sigma^{SOM}(\mathbf{r},\mathbf{r}') = \exp\left(-\frac{\|\mathbf{r}-\mathbf{r}'\|_A}{2\sigma^2}\right)$ and $\|\mathbf{r}-\mathbf{r}'\|_A$ is the distance in the SOM-lattice A according to its topological structure [18]. Further, $P(\mathbf{v}$ is the data density and the Kronegger symbol $\delta_{\mathbf{r}}^{s(\mathbf{v})}$ assigns a data vector $\mathbf{v}$ to the winning unit $s(\mathbf{v})$. $K(\sigma)$ is a normalization constant depending on the neighborhood range σ. Then the stochastic gradient prototype update for all prototypes is given as [18]:

$$\triangle \mathbf{w}_{\mathbf{r}} = -\varepsilon h_\sigma^{SOM}(\mathbf{r}, s(\mathbf{v})) \frac{\partial d(\mathbf{v}, \mathbf{w}_{\mathbf{r}})}{\partial \mathbf{w}_{\mathbf{r}}}. \tag{18}$$

depending on the derivatives of the used dissimilarity measure d, which allows the application of differentiable kernel metrics.

Prototype based classification in the context of learning vector quantization models (LVQ, [21]) was renewed by the idea of SATO&YAMADA to approximate the non-differentiable classification error C by a differentiable function E_C referred as *Generalized* LVQ (GLVQ,[39,38]). As in unsupervised vector quantization, E_C depends on the underlying dissimilarity measure $d(\mathbf{v}, \mathbf{w}_k)$ according to

$$E_C(W) = \frac{1}{2} \sum_{\mathbf{v}\in V} f(\mu(\mathbf{v})) \text{ with } \mu(\mathbf{v}) = \frac{d^+(\mathbf{v}) - d^-(\mathbf{v})}{d^+(\mathbf{v}) + d^-(\mathbf{v})}\ . \tag{19}$$

with $d^+(\mathbf{v}) = d(\mathbf{v}, \mathbf{w}^+)$ denoting the distance between the data point $\mathbf{v}$ and the nearest prototype $\mathbf{w}^+$, belonging to the same class as the presented data point $\mathbf{v}$. Analogously, $d^-(\mathbf{v})$ is defined as the distance to the best matching prototype of all other classes. The function $\mu(\mathbf{v})$ is the classifier function. Like in SOMs, $d(\mathbf{v}, \mathbf{w})$ in (19) is required to be some differentiable dissimilarity measure with respect to $\mathbf{w}$. Then the cost function can be minimized by gradient descent learning based on the (stochastic) derivatives

$$\frac{\partial_s E}{\partial \mathbf{w}^+} = \frac{2d^- \cdot f'(\mu(\mathbf{v}))}{(d^+ + d^-)^2} \frac{\partial d^+}{\partial \mathbf{w}^+}, \quad \frac{\partial_s E}{\partial \mathbf{w}^-} = -\frac{2d^+ \cdot f'(\mu(\mathbf{v}))}{(d^+ + d^-)^2} \frac{\partial d^-}{\partial \mathbf{w}^-} \quad , \tag{20}$$

where we used the abbreviations d^+ for $d^+(\mathbf{v})$ for simplicity and d^+, analogously.

Thus, stochastic gradient learning in supervised and unsupervised vector quantization can be seen as a gradient descent learning of an error function in the metric space $(V, d(\mathbf{v}, \mathbf{w}_k))$. Obviously, under gentle conditions on V (continuous, local convex, ...) it can be assumed that $\partial d(\mathbf{v}, \mathbf{w}_k) / \partial \mathbf{w}_k \in V$ is valid. Yet, the choice of the metric is free except the necessary differentiability. Hence, metrics determined by differentiable kernel are applicable [15]. Obviously, the kernels presented in Sec.2 and 3 are differentiable (for the latter kernels, see [47] for differentiability of the respective divergences). If such a metric is obtained from an universal kernel κ_Φ for RKHS, respectively, the Lemma 2 ensures the topological and isometric equivalence to the respective FMHS. Hence, the algorithm operates in the same structural space as SVMs do and, therefore, can profit from its richness in shape, which frequently delivers excellent performance. At this point we empasize the following essential drawn from the Lemma 2:

Remark 3. The take home message of the Lemma 2 in context of gradient based online learning is: Assume a set of prototypes W', which has to be learned in the induced image space $\mathcal{I}_{\kappa_\Phi} \subseteq \mathcal{H}$. Because $\mathcal{I}_{\kappa_\Phi}$ is a subspace of $\mathcal{H}$ any linear combinations of prototypes belongs to $\mathcal{H}$. Further, if the corresponding universal kernel κ_Φ is continuous and differentiable, it is sufficient to train prototypes W by gradient descent learning in the isomorph-isometric space $(V, d_\mathcal{H})$ induced by the mapping Ψ. Lemma 2 ensures the unique equivalence. An analogous statement obviously holds also for the Banach space problem.

More properties of differentiable Mercer-like kernels and their reproducing properties can be found in [12,48].

5 Exemplary Applications

In this section we briefly give results from exemplary applications for classification problems. We compare the GLVQ with differentiable kernels (DK-GLVQ) with several state-of-the-art prototype based classification algorithms including SVMs using Gaussian kernels based on an Extreme Learning Kernel (ELM,[13]) and improved GLVQ variants. For the latter we consider standard GLVQ with Euclidean metric, and the powerful variant based on matrix learning (GMLVQ, [42]) as a generalization of the relevance learning approach [16]. The GMLVQ uses the distance $d(\mathbf{v}, \mathbf{w}) = (\Omega(\mathbf{v} - \mathbf{w}))^2$ with a here squared matrix Ω, which is automatically adapted during learning for optimal classification performance. Moreover, we include the recently proposed kernel GLVQ (KGLVQ) based on a Nystrøm-approximation [40]. For the DK-GLVQ we applied two variants: The first one used a Gaussian kernel with self-adapting kernel-with σ. The second one uses the kernel $\kappa_\Phi(\mathbf{v}, \mathbf{w}) = \exp\left(-(\Omega(\mathbf{v} - \mathbf{w}))^2\right)$ with a self-adapting non-degenerating (squared) matrix for comparison with GMLVQ. We refer to this variant as DK-GMLVQ. This variant is much more stable than the σ-adjusting variant which may be addressed to the regularizing properties of matrix learning known from GMLVQ [5,6].

We applied these algorithms to two standard benchmark data sets taken from UCI repository [7]. Both data sets are two-class problems to to establish compatibility with SVMs. The first one is breast cancer data set (WDBC) consisting of 569 samples with 32 dimensions. The second data set is a diabetes study (PIMA) with 768 eight-dimensional samples. All experiments were performed by three-fold cross-validation. For the GLVQ variants we used one prototype for each class. The SVM resulted in 512 and 691 support vectors for both problems, respectively. The results are depicted in Tab. 1

Table 1. Classification accuracies in % together with their variances for the several algorithms and datasets (PIMA and WDBC). Results are obtained by three-fold cross-validation.

Dataset	GLVQ	KGLVQ	DK-GLVQ	GMLVQ	DK-GMLVQ	SVM-ELM
PIMA	75.1(±0.062)	71.1 (±0.031)	76.2 (±0.031)	77.7 (±0.016)	**78.3** (±0.025)	76.4 (±0.042)
WDBC	93.49(±0.016)	92.3(±0.034)	92.2(±0.009)	94.7 (±0.020)	95.4 (±0.025)	**97.7** (±0.014)

We observe a good performance of both kernel GLVQ variants using differentiable kernels. These are significantly improved compared to KGLVQ, which uses approximation techniques. Hence, we can conclude that the Nystrøm-approximation leads to a significant loss in accuracy. Further, comparison to GLVQ and GMLVQ also shows clear improvements, although standard GMLVQ achieved high performance. Last but not least, comparison to the SVM demonstrates that differentiable kernel are an excellent alternative to SVM. In particular we emphasize the drastically reduced model complexity taking only two prototypes compared to hundreds of support vectors while achieving similar accuracies.

6 Conclusion

In this paper we considered the theoretical framework of differentiable kernels for application in gradient based learning in supervised and unsupervised prototype based vector quantization. We show that utilization of a data metric determined by universal kernels as known from support vector machines leads to an optimization space equivalent and isometric to a reproducing kernel Hilbert space. Hence, gradient based vector quantization schemes with differentiable universal kernels can benefit from this property. The main results of topological and isometric equivalence is the Lemma 2. An extension of this theory for reproducing kernel Banach spaces can be found in [48], which assume weaker restrictions and, therefore, offer greater flexibility [49]. Last but not least we provide some examples of differentiable universal kernels based on divergences as fundamental information theoretic concepts. Further, we demonstrated abilities of GLVQ using differentiable kernel for exemplary datasets, which show high performance also compared to SVMs but with lower model complexity.

Otherwise, the presented approach cannot deal with arbitrary kernels such as structure kernels. So the method trades increased efficiency by reduced flexibility in kernel choice.

References

1. Aronszajn, N.: Theory of reproducing kernels. Transactions of the American Mathematical Society 68, 337–404 (1950)
2. Ben-Hamza, A., Krim, H.: Jensen-Rényi divergence measure: theoretical and computational perspectives. In: Proceedings of the IEEE International Symposium on Information Theory, pp. 257–257 (2003)
3. Bezdek, J.: A convergence theorem for the fuzyy ISODATA clustering algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence 2(1), 1–8 (1980)
4. Bezdek, J.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York (1981)
5. Biehl, M., Bunte, K., Schleif, F.-M., Schneider, P., Villmann, T.: Large margin discriminative visualization by matrix relevance learning. In: Abbass, H., Essam, D., Sarker, R. (eds.) Proc. of the International Joint Conference on Neural Networks (IJCNN), Brisbane, pp. 1873–1880. IEEE Computer Society Press, Los Alamitos (2012)
6. Biehl, M., Hammer, B., Schleif, F.-M., Schneider, P., Villmann, T.: Stationarity of matrix relevance learning vector quantization. Machine Learning Reports 3(MLR-01-2009), 1–17 (2009) ISSN:1865-3960, http://www.uni-leipzig.de/~compint/mlr/mlr_01_2009.pdf
7. Blake, C., Merz, C.: UCI repository of machine learning databases. Dep. of Information and Computer Science, University of California, Irvine (1998), http://www.ics.edu/mlearn/MLRepository.html
8. Chan, A., Vasconcelos, N., Moreno, P.: A family of probabilistic kernels based on information divergence. Technical Report SVCL-TR 2004/01, Statistical Visual Computing Laboratory (SVCL) at Universit of California, San Diego (2004)
9. Cichocki, A., Amari, S.-I.: Families of alpha- beta- and gamma- divergences: Flexible and robust measures of similarities. Entropy 12, 1532–1568 (2010)
10. Cichocki, A., Cruces, S., Amari, S.-I.: Generalized alpha-beta divergences and their application to robust nonnegative matrix factorization. Entropy 13, 134–170 (2011)
11. Crammer, K., Gilad-Bachrach, R., Navot, A., Tishby, A.: Margin analysis of the LVQ algorithm. In: Becker, S., Thrun, S., Obermayer, K. (eds.) Advances in Neural Information Processing (Proc. NIPS 2002), vol. 15, pp. 462–469. MIT Press, Cambridge (2003)
12. Ferreira, J., Menegatto, V.: Reproducing properties of differentiable Mercer-like kernels. Mathematische Nachrichten 285 (in press, 2012)
13. Frénay, B., Verleysen, M.: Parameter-free kernel in extreme learning for non-linear support vector regression. Neurocomputing 74(16), 2526–2531 (2011)
14. Gretton, A., Herbrich, R., Smola, A., Bousquet, O., Schölkopf, B.: Kernel methods for measuring independence. Journal of Machine Learning Research 6, 2075–2129 (2005)
15. Hammer, B., Strickert, M., Villmann, T.: Supervised neural gas with general similarity measure. Neural Processing Letters 21(1), 21–44 (2005)

16. Hammer, B., Villmann, T.: Generalized relevance learning vector quantization. Neural Networks 15(8-9), 1059–1068 (2002)
17. Hein, M., Bousquet, O.: Hilbertian metrics and positive definite kernels on probability measures. Technical report, Max Planck Institute for Biological Cybernetics (2004)
18. Heskes, T.: Energy functions for self-organizing maps. In: Oja, E., Kaski, S. (eds.) Kohonen Maps, pp. 303–316. Elsevier, Amsterdam (1999)
19. Hoffmann, T., Schölkopf, B., Smola, A.: Kernel methods in machine learning. The Annals of Statistics 36(3), 1171–1220 (2008)
20. Kästner, M., Hammer, B., Biehl, M., Villmann, T.: Functional relevance learning in generalized learning vector quantization. Neurocomputing 90(9), 85–95 (2012)
21. Kohonen, T.: Self-Organizing Maps. Springer Series in Information Sciences, vol. 30. Springer, Heidelberg (1995) (Second Extended Edition 1997)
22. Kolmogorov, A., Fomin, S.: Reelle Funktionen und Funktionalanalysis. VEB Deutscher Verlag der Wissenschaften, Berlin (1975)
23. Kulis, B., Sustik, M., Dhillon, I.: Low-rank kernel learning with Bregman matrix divergences. Journal of Machine Learning Research 10, 341–376 (2009)
24. Kullback, S., Leibler, R.: On information and sufficiency. Annals of Mathematical Statistics 22, 79–86 (1951)
25. Martin, A., Smith, N., Xing, E., Aguiar, P., Figueiredo, M.: Nonextensive information theoretic kernels on measures. Journal of Machine Learning Research 10, 935–975 (2009)
26. Martinetz, T.M., Berkovich, S.G., Schulten, K.J.: 'Neural-gas' network for vector quantization and its application to time-series prediction. IEEE Trans. on Neural Networks 4(4), 558–569 (1993)
27. Mercer, J.: Functions of positive and negative type and their connection with the theory of integral equations. Philosophical Transactions of the Royal Society, London, A 209, 415–446 (1909)
28. Micchelli, C., Xu, Y., Zhang, H.: Universal kernels. Journal of Machine Learning Research 7, 26051–22667 (2006)
29. Mwebaze, E., Schneider, P., Schleif, F.-M., Aduwo, J., Quinn, J., Haase, S., Villmann, T., Biehl, M.: Divergence based classification in learning vector quantization. Neurocomputing 74(9), 1429–1435 (2011)
30. Nielsen, F., Nock, R.: Sided and symmetrized Bregman centroids. IEEE Transaction on Information Theory 55(6), 2882–2903 (2009)
31. Österreicher, F., Vajda, I.: A new class of metric divergences on probability spaces and its applicability in statistics. Annals of the Institute of Statistical Mathematics 55(3), 639–653 (2003)
32. Pekalska, E., Duin, R.: The Dissimilarity Representation for Pattern Recognition: Foundations and Applications. World Scientific (2006)
33. Principe, J.: Information Theoretic Learning. Springer, Heidelberg (2010)
34. Qin, A., Suganthan, P.: A novel kernel prototype-based learning algorithm. In: Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004), vol. 4, pp. 621–624 (2004)
35. Qin, A.K., Suganthan, P.N.: Kernel neural gas algorithms with application to cluster analysis. In: ICPR (4), pp. 617–620 (2004)
36. Rényi, A.: On measures of entropy and information. In: Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability. University of California Press (1961)
37. Rényi, A.: Probability Theory. North-Holland Publishing Company, Amsterdam (1970)

38. Sato, A., Yamada, K.: Generalized learning vector quantization. In: Touretzky, D.S., Mozer, M.C., Hasselmo, M.E. (eds.) Proceedings of the 1995 Conference Advances in Neural Information Processing Systems, vol. 8, pp. 423–429. MIT Press, Cambridge (1996)
39. Sato, A.S., Yamada, K.: Generalized learning vector quantization. In: Tesauro, G., Touretzky, D., Leen, T. (eds.) Advances in Neural Information Processing Systems, vol. 7, pp. 423–429. MIT Press (1995)
40. Schleif, F.-M., Villmann, T., Hammer, B., Schneider, P.: Efficient kernelized prototype based classification. International Journal of Neural Systems 21(6), 443–457 (2011)
41. Schölkopf, B., Smola, A.: Learning with Kernels. MIT Press (2002)
42. Schneider, P., Hammer, B., Biehl, M.: Adaptive relevance matrices in learning vector quantization. Neural Computation 21, 3532–3561 (2009)
43. Scovel, C., Hush, D., Steinwart, I., Theiler, J.: Radial kernels and their reproducing kernel Hilbert spaces. Journal of Complexity 26, 641–660 (2010)
44. Shannon, C.: A mathematical theory of communication. Bell System Technical Journal 27, 379–432 (1948)
45. Sriperumbudur, B., Fukumizu, K., Lanckriet, G.: Universality, characteristic kernels, and RKHS embedding of measures. Journal of Machine Learning Research 12, 2389–2410 (2011)
46. Steinwart, I.: On the influence of the kernel on the consistency of support vector machines. Journal of Machine Learning Research 2, 67–93 (2001)
47. Villmann, T., Haase, S.: Divergence based vector quantization. Neural Computation 23(5), 1343–1392 (2011)
48. Villmann, T., Haase, S.: A note on gradient based learning in vector quantization using differentiable kernels for Hilbert and Banach spaces. Machine Learning Reports 6(MLR-02-2012), 1–29 (2012) ISSN:1865-3960, http://www.techfak.uni-bielefeld.de/~fschleif/mlr/mlr$_$02$_$2012.pdf,
49. Zhang, H., Xu, Y., Zhang, J.: Reproducing kernel banach spaces for machine learning. Journal of Machine Learning Research 10, 2741–2775 (2009)

Nonlinear Time Series Analysis by Using Gamma Growing Neural Gas

Pablo A. Estévez and Jorge R. Vergara

Dept. Electrical Engineering and Advanced Mining Technology Center, University of Chile,
Casilla 412-3, Santiago, Chile
{pestevez,jorgever}@ing.uchile.cl

Abstract. In this paper, we investigate the properties of the Gamma Growing Neural Gas (γ-GNG) model for the analysis of nonlinear time series. This model includes a temporal context descriptor based on a short term memory structure called Gamma memory. It is shown that γ-GNG can approximately reconstruct the space-state, and filter out additive noise. Simulation results on two data sets are presented: Lorenz system and NH3-FIR Laser time series.

1 Introduction

Several variants of self-organizing feature maps (SOMs) [1] can deal with processing data sequences that are temporally or spatially connected, such as words, DNA sequences, time series, etc. [2], [3]. As fully recursive models are rather expensive, other models simply add context vectors to the conventional weight vectors (prototypes). A context vector represents the temporal context in the previous time steps. In Merge SOM (MSOM) [4], the context is described by a linear combination of the weight and the context of the last winner neuron. As this type of context does not depend on the lattice architecture, it can be combined with other self-organizing neural networks such as Neural Gas (NG) [5] and Growing Neural Gas (GNG) [6], to produce the Merge Neural Gas (MNG) [7] and the Merge Growing Neural Gas (MGNG) [8], respectively.

In our previous work we have added Gamma filters [9] to SOM, NG and GNG, yielding the γ-SOM [10], γ-NG [11], and γ-GNG models [12], respectively. It has been shown that the gamma filter variants of SOM, NG and GNG are generalizations that include as particular examples the MSOM, MNG and MGNG models, when the filter order is set to one. In general, performance of the gamma models are better than those of merge models under the temporal quantization error metric.

In this paper, we investigate the properties of γ-GNG for analysis of nonlinear time series. We show that γ-GNG can reconstruct the state-space approximately, as well as filter out noise. Results are shown on two data sets: the Lorenz system and the NH3-FIR Laser time series.

2 Delay Coordinate Embedding

The state space is the set of all possible states of a deterministic dynamical system. Embedding theorems, e.g. Takens [13], Sauer [14], allow us to reconstruct the internal

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 205–214.
DOI: 10.1007/978-3-642-35230-0_21

dynamics of an n-dimensional state space starting from a one-dimensional time series, $x_i(t)$, which corresponds to a sequence of scalar measurements of the state space or a single state variable. To embed a time series, the following delay coordinate vector, whose components consists of time-delayed versions of the samples, is constructed:

$$s(t) = [x_i(t), x_i(t-\tau), \cdots, x_i(t-(m-1)\times\tau)] \tag{1}$$

where τ and m are the embedding parameters. Takens theorem guarantees, under certain conditions, a one-to-one correspondence between the reconstructed dynamics and the true dynamics of the system. A correct embedding requires $m \geq 2 \times D$, where D is the dimension of the internal dynamics, e.g. an attractor. Takens theorem does not provide a way to estimate the embedding dimension m and the delay τ, but many heuristics methods have been proposed in the literature. The parameter τ is usually estimated by seeking for the delay that provides the first minimum of the average mutual information [15]. The false nearest neighbor algorithm [16] computes an upper bound on the parameter m. Estimating τ and m is computationally complex and numerically sensitive [17], reason why nonlinear time series analysis techniques that do not require these parameters are extremely attractive.

3 Gamma Context Model

In the 90's gamma filters were studied in the context of focused multilayer perceptron neural networks by Principe et al. [9,19]. The gamma filter is defined in the time domain as follows:

$$y(n) = \sum_{k=0}^{K} w_k c_k(n)$$

$$c_k(n) = \beta c_k(n-1) + (1-\beta)c_{k-1}(n-1) \tag{2}$$

where $c_0(n) \equiv x(n)$ is the input signal and $y(n)$ is the filter output, and $w_0, \cdots, w_K$ are the weight parameters of the filter. The parameter $\beta \in (0,1)$ allow us to decouple depth (D) and resolution (R) from the filter order, K. Depth measures how far into the past the memory stores information, and resolution indicates the degree to which information about each individual element of the input sequence is preserved. The average memory depth for a Gamma memory of order-K becomes [19],

$$D = \frac{K}{(1-\beta)} \tag{3}$$

and its resolution is

$$R = 1 - \beta.$$

By increasing the filter order, K, the Gamma filter can achieve an increasing memory depth without compromising resolution.

Let $\mathcal{N} = \{1, \ldots, M\}$ be a set of neurons. Each neuron has associated a weight vector $w^i \in \Re^d$, for $i = 1, \ldots, M$, and also a set of context vectors $\mathcal{C} = \left\{c_1^i, c_2^i, \ldots, c_K^i\right\}$,

$c_k^i \in \Re^d$, $k = 1, \ldots, K$, where K is the Gamma filter order. Given a sequence s, the context set $\mathcal{C}$ is initialized at zero values.

Given an input, $x(n)$, the best matching unit, I_n, is the neuron that minimizes the following distance criterion,

$$d_i(n) = \alpha_w \left\| x(n) - w^i \right\|^2 + \sum_{k=1}^{K} \alpha_k \left\| c_k(n) - c_k^i \right\|^2 \tag{4}$$

where the parameters α_w and $\alpha_k,\ k \in \{1, 2, \ldots, K\}$ control the relevance of the different elements. To compute the recursive distance (4) every context descriptor in the different filtering stages is required. Formally, the K context descriptors of the current unit are defined as gamma memories:

$$c_k(n) = \beta c_k^{I_{n-1}} + (1-\beta)\, c_{k-1}^{I_{n-1}} \quad \forall k = 1, \ldots, K \tag{5}$$

where $c_0^{I_{n-1}} \equiv w^{I_{n-1}}$ and at $n = 0$ the initial conditions $c_k^{I_0}, \forall k = 1, \ldots, K$ are set randomly. When $K = 1$, the context used in the merge family of models is recovered.

Because the context of order k is constructed recursively, depending on the context of order $k-1$, it is recommended that $\alpha_w > \alpha_1 > \alpha_2 > \cdots > \alpha_K > 0$, otherwise errors in the early filter stages would propagate through higher-order contexts.

3.1 γ-GNG Algorithm

Assuming a univariate time series, a Gamma filtered embedding of the time series is constructed as follows:

$$u(t) = [w^i(t), c_1^i(t), \cdots, c_K^i(t)] \tag{6}$$

where w^i is the weight scalar, and c_k^i, for $k = 1, \cdots, K$, are K temporal contexts associated to the $i-th$ neuron, for $i = 1, \cdots, M$. In other words each neuron is represented by a (K+1)-dimensional vector composed of the weight scalar and K temporal contexts. In the following the $gamma$-GNG algorithm is described.

1. Initialize randomly two weights w^i, and set to zero their respective contexts, c_k^i, for $k = 1, \cdots, K$, $i = 1, 2$. Connect them with a zero age edge and set to 0 their respective winner counters, $wcount_i$.
2. Present input vector, $x(n)$, to the network
3. Calculate context descriptors $c_k(n)$ using eq. (5)
4. Find best matching unit (BMU), I_n, and the second closest neuron,J_n, using eq. (4)
5. Update the BMUs local winner count variable: $wcount_{I_n} = wcount_{I_n} + 1$
6. Update the BMU's weight and contexts using the following rule

$$\begin{aligned} \triangle \mathbf{w}^i &= \epsilon_w(n) \cdot \left(\mathbf{x}(n) - \mathbf{w}^i \right) \\ \triangle \mathbf{c_k}^i &= \epsilon_w(n) \cdot \left(\mathbf{c_k}(n) - \mathbf{c_k}^i \right) \end{aligned} \tag{7}$$

   Update neighboring units (i.e. all nodes connected to the BMU by an edge) using step-size $\epsilon_c(n)$ instead of $\epsilon_w(n)$ in eq. (7).

7. Increment the age of all edges connecting the BMU and their topological neighbors, $a_j = a_j + 1$.
8. If the BMU and the second closest neuron are connected by an edge, then set the age of that edge to 0. Otherwise create an edge between them.
9. If there are any edges with an edge larger than a_{max} then remove them. If after this operation, there are nodes with no edges remove them.
10. If the current iteration n is an integer multiple of λ, and the maximum node count has not been reached, then insert a new node. The parameter λ controls the number of iterations required before inserting a new node. Insertion of a new node, r, is done as follows:
    (a) Find node u with the largest winner count.
    (b) Among the neighbors of u, find the node v with the largest winner count
    (c) Insert the new node r between u and v as follows,

$$\begin{aligned} w^r &= 0.75w^u + 0.25w^v \qquad (8) \\ {c_k}^r &= 0.75{c_k}^u + 0.25{c_k}^v \end{aligned}$$

    (d) Create edges between u and r, and v and r, and remove the edge between u and v
    (e) Decrease the winner count variables of nodes u and v by a factor $1 - \tilde{\alpha}$, and set the winner count of node r as follows,

$$\begin{aligned} wcount_u &= (1 - \tilde{\alpha}) \times wcount_u \qquad (9) \\ wcount_v &= (1 - \tilde{\alpha}) \times wcount_v \qquad (10) \\ wcount_r &= wcount_u \end{aligned}$$

11. Decrease winner count variables of all nodes, $j = 1, \cdots, J$ by a factor $1 - \tilde{\beta}$,

$$wcount_j = (1 - \tilde{\beta}) \times wcount_j$$

    Typically, $\tilde{\alpha} = 0.5$ and $\tilde{\beta} = 0.0005$.
12. Set $n \to n + 1$
13. If $n < L$ go back to step 2, where L is the cardinality of the data set.

3.2 Selecting the Best Model

The performance of the γ-GNG algorithm depends on the Gamma filter parameters β and K. The parameter β was varied from 0 to 1 with 0.1 steps, and the number of filter stages K was varied from 1 to 15, giving a total of 165 different models for each dataset. The temporal quantization error (TQE) [2] is used as a performance criterion to select the best models. TQE measures the average standard deviation of signals within the receptive field of each neuron in the grid for a certain past input. This generates a curve of quantization error versus the index of past inputs. This curve can be averaged to yield the TQE of the whole map, which we use as an indicator of quality of the spatio-temporal quantization. With the aim of measuring the TQE performance, a time window of 30 past events was used. This number is related to the average depth, D, of the Gamma memory as defined in eq. (3), e.g. if $K = 10$ and $\beta = 0.7$ then $D = 33$. The size of this window does not affect the function of the models, and it is used only for computing the TQE metric.

In practice, we noticed that several maps obtained by using γ-GNG can have very similar TQE values, although their dynamics look quite different. For this reason the top 10 TQE values were selected for further analysis, and an additional criterion was used to discriminate among these peaks. Remember that our goal is to reconstruct the original state-space. Notice from eq. (6) that γ-GNG renders a $(K+1)$-dimensional representation of the time series when using K contexts. As a result of this spatio-temporal quantization, the original univariate time series is transformed to a $(K+1)$-dimensional time series, by simply assigning to each point of the time series a neuron represented by a (K+1)-dimensional vector. This $(K + 1)$-dimensional representation can be projected onto a one-dimensional or a two-dimensional space by using principal component analysis (PCA), giving 1D-PCA or 2D-PCA projections, respectively. We searched for the 1D-PCA projection that has maximal mutual information with the original time series. Mutual information has the property of being invariant to scale, which is an advantage in this case because the dynamics of nonlinear time series, e.g. an attractor or fixed point, is also invariant to scale. However, 1D-PCA projections may be out of phase with the original time series, because the contexts contain information of the past inputs. Notice that the dynamics of a nonlinear time series is also invariant to a shift of the whole time series. Therefore, to select the best model among the top 10 peaks of the TQE measurement, we searched for the maximal mutual information between the original time series and delayed 1D-PCA projections. For obtaining the right delay, we started with the 1D-PCA time series without delay, and then increased the delay in unit steps until getting the maximum mutual information with the original time series (i.e., we looked for the first maximum of the mutual information as a function of the number of delays). The mutual information was computed using Fraser's algorithm [15].

4 Experiments

Experiments were carried out with two data sets: Lorenz system and NH3-FIR laser time series. The Lorenz system corresponds to a set of 3 differential equations for modeling convective fluid dynamics [20]. For certain parameters the Lorenz system dynamics contains a strange attractor. We used 4000 samples with a sampling interval of 0.005[s] of the state variable x(t), in order to reconstruct the state-space, see Fig. 1a). The Far-Infrared (FIR) laser time series corresponds to data set A[1] in the Santa Fe time series competition. This is a univariate time series, containing 1000 measurements from a FIR-Laser in a chaotic state. Fig. 1b) plots the laser time series. The laser NH3 time series is a sequence of 8-bit integer numbers, and the measurement error is at least the digitalization error [18]. It has been shown that the laser chaotic pulsations follow approximately the theoretical Lorenz model [21]. Both datasets have been widely used in the literature to reconstruct the state-space by using delay coordinate embedding [18].

[1] Available at `http://www-psych.stanford.edu/~andreas/Time-Series/SantaFe.html`

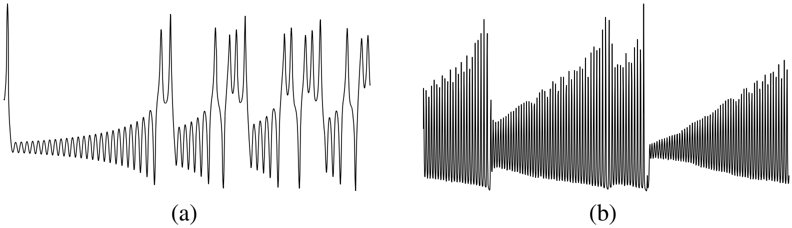


Fig. 1. a) Lorenz x(t) time series, and b) NH3-FIR laser time series

Training in γ-GNG is done in a single stage, during 200 epochs for each dataset using 100 neurons. Parameters α_i are fixed, and decayed linearly with the context order as follows:

$$\alpha_i = \frac{K+1-i}{\sum_{k=0}^{K}(k+1)},\ i = 0 \ldots K \tag{11}$$

with $\alpha_w \equiv \alpha_0$. The parameters used in (7) were set as $\epsilon_w = 0.05$, $\epsilon_c = 0.0006$.

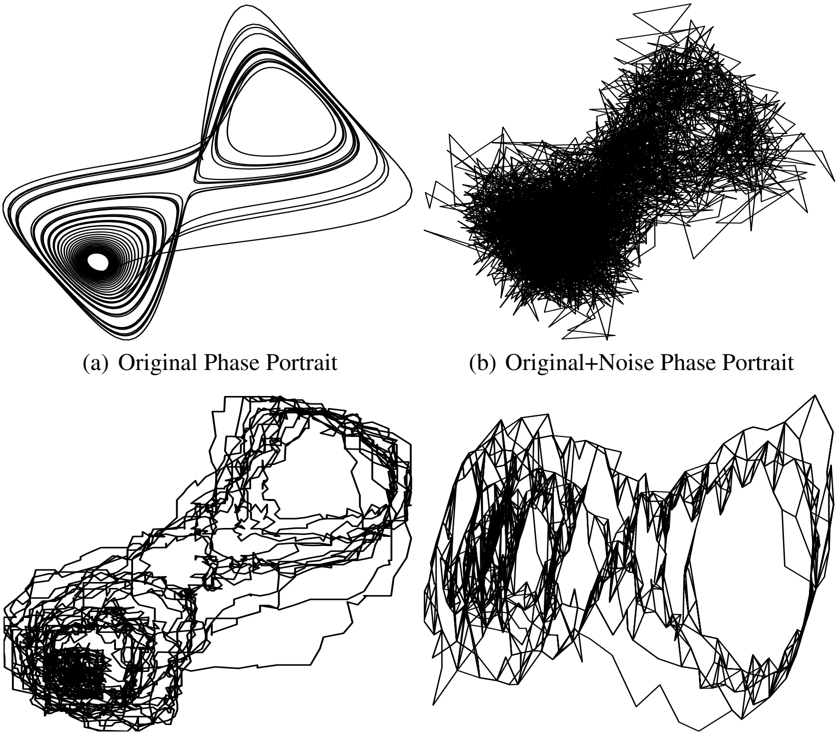
4.1 Lorenz System

The Lorenz time series $x(t)$ was contaminated with additive Gaussian white noise with standard deviations $\sigma = 0, 0.5, 1.0, 2.0, 2.5, 3.0$. Table 1 shows the Gamma filter parameters (β, K) corresponding to the best models found for different levels of noise. Remember that we do first a grid search of (β, K) values as described in Section 3.2, in order to find the top ten TQE values. Next, for these ten cases, we compute the mutual information (MI) between the original time series and the 1D-PCA projection using different delays, in order to align the time series. Table 1 shows the maximum MI values and the delays obtained. As expected the TQE values increase with the level of noise, while the MI values diminish.

Fig. 2a) shows the phase portrait of the original Lorenz time series $x(t)$ versus $x(t-\tau)$, where $\tau = 17$ delays (corresponding to $0.09[s]$). This value of τ was computed by using the traditional method of seeking the first minimum of the mutual information described in section 2. Fig. 2b) shows the phase portrait of the original Lorenz time

Table 1. Best γ-GNG models (β,K) obtained for the Lorenz x(t) time series with different standard deviations (σ) of additive Gaussian white noise. The table shows the temporal quantization error (TQE), maximal mutual information value (MI) and delay obtained.

σ Noise	β	K	TQE	MI	Delay
0.0	0.4	4	0.7099	5.0550	3
0.5	0.5	6	0.9837	4.0384	6
1.0	0.2	12	1.2809	3.4469	7
1.5	0.4	11	1.6389	3.0123	8
2.0	0.5	9	1.9250	2.8712	8
2.5	0.4	12	2.2117	2.5837	9
3.0	0.3	13	2.3864	2.5708	9



(a) Original Phase Portrait

(b) Original+Noise Phase Portrait

(c) γ-GNG 1D-PCA Phase Portrait

(d) γ-GNG 2D-PCA projection

Fig. 2. Results for Lorenz x(t) time series: a) Phase portrait of original time series b) Phase portrait of original time series plus additive white noise of $\sigma = 3.0$, c) Phase portrait of 1D-PCA projection obtained with γ-GNG using the noisy time series as input, and d) 2D-PCA projection obtained with γ-GNG using the noisy time series as input

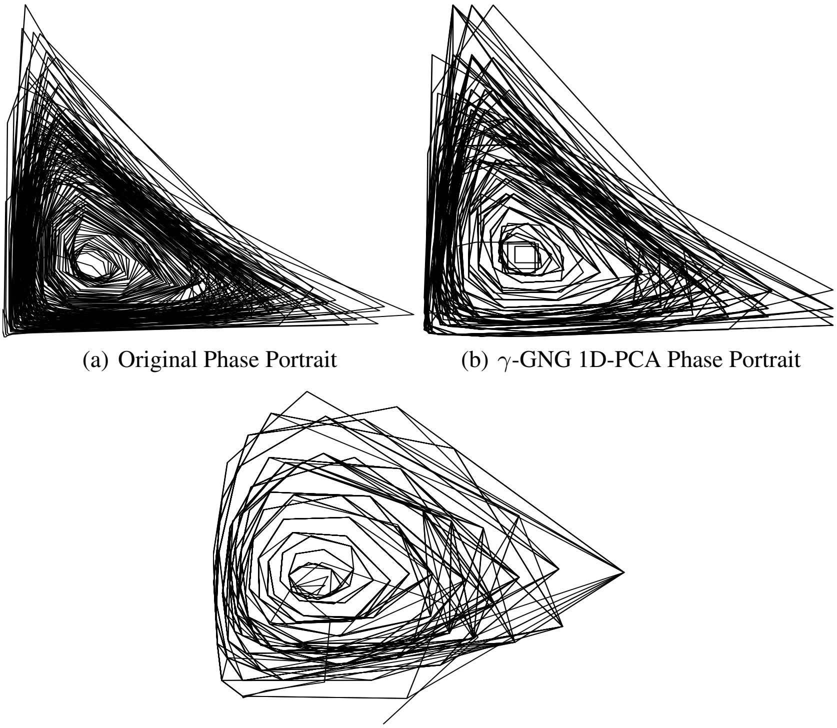
Table 2. Best γ-GNG model (β,K) obtained for the NH3-FIR laser time series. The table shows the temporal quantization error (TQE), maximal mutual information value (MI) and delay obtained.

β	K	TQE	MI	Delay
0.6	8	16.8211	3.2046	0

series $x(t)$ with additive white noise of standard deviation $\sigma = 3.0$, using $\tau = 17$ delays. Fig. 2c) shows the phase portrait of the 1D-PCA projection of the best model obtained with γ-GNG applied on the Lorenz $x(t)$ time series with additive white noise of standard deviation $\sigma = 3.0$, using $\tau = 17$ delays. Fig. 2d) shows directly a 2D-PCA projection corresponding to the best model obtained with γ-GNG for the noise level of $\sigma = 3.0$. In this 2D-PCA projection the dots correspond to neurons, and the links are created by connecting the closest neurons. It can be clearly observed by comparing figs. 2b) and 2c) with fig. 2a), that the phase portrait using the 1D-PCA projection obtained with γ-GNG is able to capture the dynamics of the strange attractor, filtering out most of the white noise. Fig. 2d) shows an alternative way of visualizing the strange attractor dynamics without using coordinate delay embedding. This plot is different than the phase portraits of figs. 2a-c), because the latter depend on the embedding parameter τ.

4.2 NH3-FIR Laser Time Series

Table 2 shows the Gamma filter parameters (β, K) corresponding to the best model found for the NH3-FIR laser time series. In this case the maximal mutual information between the original time series and the 1D-PCA projection of the model obtained with γ-GNG was found without any delay. Fig. 3a) shows the phase portrait of the original Lorenz time series $x(t)$ versus $x(t - \tau)$, where $\tau = 2$ was computed by seeking for the first minimum of the mutual information. Fig. 3b) shows the phase portrait of the 1D-PCA projection of the model obtained with γ-GNG applied on the laser time series, using $\tau = 2$. Fig. 3c) shows a 2D-PCA projection corresponding to the best model obtained with γ-GNG. In this projection the dots correspond to neurons, and the links are created by connecting the closest neurons. It can be observed by comparing fig. 3a) with fig. 3b), that the phase portrait using the 1D-PCA projection obtained with γ-GNG



(a) Original Phase Portrait

(b) γ-GNG 1D-PCA Phase Portrait

(c) γ-GNG 2D-PCA projection

Fig. 3. Results for Laser time series: a) Phase portrait of original time series b) Phase portrait of 1D-PCA projection obtained with γ-GNG , and c) 2D-PCA projection obtained with γ-GNG (direct state space reconstruction)

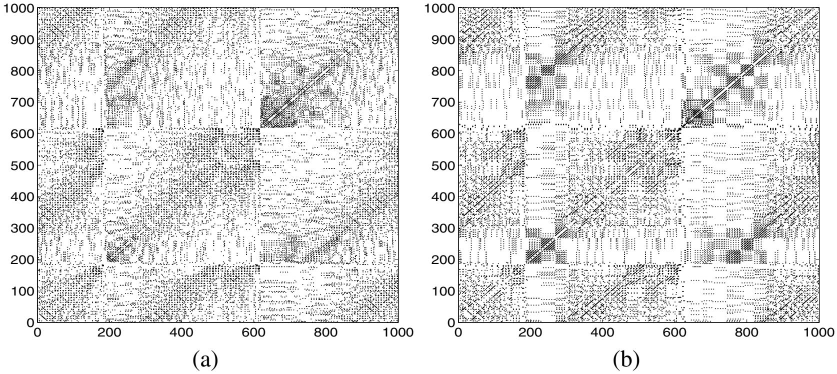


Fig. 4. Recurrent plots for Laser time series: a) Original time series, and b) 1D-PCA projection obtained with γ-GNG

is able to capture correctly the dynamics of the strange attractor. Fig. 3c) shows an alternative way of visualizing the strange attractor by using Gamma filtered embedding instead of delay coordinate embedding. Recurrence plots (RPs) is a method for visualizing the recurrences in dynamical systems [22]. Fig. 4a) shows the RP obtained with the original laser time series (1000 points), where the error distance has been setup to show only 2.5% of the points. Fig. 4b) shows the RP obtained with the 1D-PCA time series obtained with the γ-GNG model. By comparing Figs. 4a) and b) it can be seen that plot b) has less noisy isolated points (more white areas) than plot a), and it also shows more structure around transition points 200 and 600.

5 Conclusion

We have investigated the properties of the γ-GNG model for nonlinear time series analysis. We have shown that γ-GNG can capture the invariant properties of nonlinear dynamics, such as strange attractors. Another important property of the γ-GNG is its capacity to filter out noise as shown in the two time series analyzed. The proposed model built a kind of embedding by using Gamma filters instead of delay coordinates. The (β,K) parameters of the γ-GNG model may be related somehow to the $(\tau,\ m)$ embedding parameters. We have proposed a method to determine the (β,K) parameters based on the TQE minimization and mutual information. The method proposed here can easily be extended to the γ-SOM and γ-NG variants. In the near future, we plan to do research on time series prediction by using γ self-organizing neural networks.

Acknowledgement. This research was supported by Conicyt-Chile under grant Fondecyt 1110701.

References

1. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (1995)
2. Voegtlin, T.: Recursive Self-Organizing Maps. Neural Networks 15, 979–991 (2002)

3. Hammer, B., Micheli, A., Sperduti, A., Strickert, M.: Recursive Self-Organizing Network Models. Neural Networks 17, 1061–1085 (2004)
4. Strickert, M., Hammer, B.: Merge SOM for Temporal Data. Neurocomputing 64, 39–72 (2005)
5. Martinetz, T.M., Berkovich, S.G., Schulten, K.J.: "Neural-gas" Network for Vector Quantization and its Application to Time-Series Prediction. IEEE Transactions on Neural Networks, 558–569 (1993)
6. Fritzke, B.: A Growing Neural Gas Learns Topologies. In: Tesauro, G., Touretzky, D.S., Leen, T.K. (eds.) Neural Information Processing Systems (NIPS), pp. 625–632. MIT Press, Cambridge (1995)
7. Strickert, M., Hammer, B.: Neural Gas for Sequences. In: Yamakawa, T. (ed.) Proceedings of the Workshop on Self-Organizing Networks (WSOM), Kyushu, Japan, pp. 53–58 (2003)
8. Andreakis, A., Hoyningen-Huene, N.v., Beetz, M.: Incremental Unsupervised Time Series Analysis Using Merge Growing Neural Gas. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 10–18. Springer, Heidelberg (2009)
9. De Vries, B., Principe, J.C.: The Gamma Model- A New Neural Model for Temporal Processing. Neural Networks 5, 565–576 (1992)
10. Estévez, P.A., Hernández, R.: Gamma SOM for Temporal Sequence Processing. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 63–71. Springer, Heidelberg (2009)
11. Estévez, P.A., Hernández, R., Perez, C.A., Held, C.M.: Gamma-filter Self-organizing Neural Networks for Unsupervised Sequence Processing. Electronics Letters 47(8), 494–496 (2011)
12. Estévez, P.A., Hernández, R.: Gamma-Filter Self-Organizing Neural Networks for Time Series Analysis. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 151–159. Springer, Heidelberg (2011)
13. Takens, F.: Detecting Strange Atractors in Turbulence. Lecture Notes in Math., vol. 898. Springer, New York (1981)
14. Sauer, T.: Times series prediction using delay coordinate embedding. In: Weigend, A.S., Gershenfeld, N.A. (eds.) Time Series Prediction: Forecasting the Future and Understanding the Past, pp. 175–193. Addison-Wesley, FL (1994)
15. Fraser, A.M., Swinney, H.L.: Independent Coordinates for Strange Attractors from Mutual Information. Physical Review A 33, 1134–1140 (1986)
16. Kennel, M.B., Brown, R., Abarbanel, H.D.I.: Determining Embedding Dimension for Phase-Space Reconstruction Using a Geometrical Construction. Physical Review A 45, 3403–3411 (1992)
17. Bradley, E.: Analysis of Time Series. In: Berthold, M., Hand, D.J. (eds.) Intelligent Data Analysis. Springer, Berlin (1999)
18. Kantz, H., Schreiber, T.: Nonlinear Time Series Analysis. Cambridge University Press, New York (2004)
19. Principe, J.C., Giuliano, N.R., Lefebvre, W.C.: Neural and Adaptive Systems. John Wiley & Sons, Inc., New York (1999)
20. Lorenz, E.N.: Deterministic non-periodic flow. J. Atmos. Sci. 20, 130 (1963)
21. Huebner, U., Weiss, C.O., Abraham, N.B., Tang, D.: Lorenz-like Chaos in NH3-FIR Lasers (Data Set A). In: Weigend, A.S., Gershenfeld, N.A. (eds.) Time Series Prediction: Forecasting the Future and Understanding the Past, pp. 73–104. Addison-Wesley, FL (1994)
22. M.N., Romano, M.C., Thiel, M., Kurths, J.: Recurrence Plots for the Analysis of Complex Systems. Physics Reports 438, 237–329 (2007)

Robust Regional Modeling for Nonlinear System Identification Using Self-Organizing Maps

Amauri H. de Souza Junior[1], Francesco Corona[2], and Guilherme A. Barreto[1]

[1] Federal University of Ceará, Department of Teleinformatics Engineering
Av. Mister Hull, S/N - Campus of Pici, Center of Technology, Fortaleza, Ceará, Brazil
[2] Aalto University, Department of Information and Computer Science
Konemiehentie 2, Espoo, Finland

Abstract. Global modeling is a common approach to the problem of learning dynamical input-output mappings. It consists in fitting a single regression model, starting from the whole set of input and output measurements. On the other side of the spectrum, the local modeling approach segments the input space into several localized partitions (usually, Voronoi cells) and a number of specialized regression models are fit over each partition. Regional modeling stands in between the global and local approach. Firstly, the input space is indeed divided into partitions (as in local modeling), then partitions are merged into larger regions over which the regression models are built. In this paper, we extend the regional modeling approach through the use of robust regression, a statistical framework that better handles outliers and deviation of residuals from gaussianity. The approach is validated using two benchmark problems in system identification and its performance compared to those achieved by standard global and local models.

1 Introduction

Modern industrial plants have been the source of challenging tasks in dynamical system identification and control [13]. Designing control systems to achieve the level of quality demanded by current industry standards requires building accurate models of the plant being controlled. Building accurate models requires reliable data, usually in the form of input and output time series. Once such data are available, they can be used for obtaining direct or inverse models of nonlinear systems (e.g., using neural network architectures [11,3]).

Although several techniques have been proposed and applied to the modeling and control of dynamical systems [5,14], they can be roughly categorized in two main approaches: global and local modeling. Global models implement a single structure, such as a linear regression or a neural network model, that approximates the whole input-output mapping of the system being identified. Global models constitute the mainstream approach in system identification and control [12]. Local models have been a source of much interest because they have the ability to fit to the local shape of an arbitrary surface, which is particularly difficult when the dynamical system characteristics vary considerably throughout the state space. The input space is divided into partitions, each one being

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 215–224.
DOI: 10.1007/978-3-642-35230-0_22

associated with a specialized model. To estimate the system output at a given time, a single model is chosen from the pool of available local models according to some criteria defined on the current input data. Within the neural network literature, local modeling techniques have been implemented mostly through the use of the Self-Organizing Map (SOM) [6,4,14] as valuable alternatives to global models based on supervised neural network architectures.

One of the main problems with local models is the description of the off-equilibrium dynamics. This is due to the possible lack of measured data in the partitions away from equilibrium, which may render the local regression problem ill-posed [2]. Moreover, it is not straightforward how to select the appropriate number of local models beforehand, without any *a priori* information; an inappropriate selection may cause the over- or under-identification of the original system dynamics [18]. Stemming from the clustering of the SOM [16], the recently proposed Regional modeling approach (RM)[15] is a SOM-based effort to overcome such a shortcoming. The definition of more populated partitions, hereafter called regions, is obtained from the clustering of the SOM prototypes. In that sense, the RM approach stands in between global and local modeling.

As with conventional local modeling, in the regional modeling approach any regression model can be built over such regions to estimate the system's dynamics. In a previous study [15], we have investigated the performance of RM using Ordinary Least Squares (OLS) regression and the Extreme Learning Machine, ELM [8]. This work extends the regional modeling approach using such regression techniques through the application of the M-estimates proposed by Huber [9]. Based on our experimental results, we argue that we can develop models able to offer protection against outliers in each operating regime without any loss of accuracy when compared to standard regional models.

The remainder of the paper is organized as follows. In Section 2, the inverse modeling problem is introduced and the fundamentals of regional modeling are then presented. In Section 3, we describe the application of M-Estimates into Regional Modeling approach. Computer simulations and performance analysis of the proposed approach on two benchmarking problems are presented in Section 4. The main conclusions and futher works are presented in Section 5.

2 Regional Modeling

Let us assume that the dynamical system under investigation is approximated by a nonlinear autoregressive model with exogenous inputs, NARX [12]:

$$y(t) = f\left[y(t-1), \ldots, y(t-p); u(t), u(t-1), \ldots, u(t-q+1)\right] + \varepsilon(t), \qquad (1)$$

where $f(\cdot)$ is the unknown single-input single-output mapping between the system's input $u(t) \in \mathbb{R}$ and the system's output $y(t) \in \mathbb{R}$ at time t. The noise term $\varepsilon(t)$ (or error) denotes an unobserved scalar random variable. The input- and output-memory orders are denoted by $q \geq 1$ and $p \geq 1$, with $q \leq p$, respectively.

In this paper, we are interested in the inverse system identification problem; that is, the estimation of the input $u(t)$ from previous input and output observations:

$$\begin{aligned} u(t) &= f^{-1}\left[u(t-1), \ldots, u(t-q); y(t-1), \ldots, y(t-p)\right] + \epsilon(t), \qquad (2)\\ &= f^{-1}\left[\mathbf{x}(t)\right] + \epsilon(t), \end{aligned}$$

where $\epsilon(t)$ denotes an unobserved scalar random variable. Thus, for this task, the estimation problem is conventionally approached through the design of a regression model where the system's input is the target $u(t) \in \mathbb{R}$ and the explanatory variables are defined from the regression vector $\mathbf{x}(t) \in \mathbb{R}^{p+q}$.

As mentioned in the introduction, a global regression model may not be able to capture the dynamics of regions with high curvatures, thus smoothing out some of the details. Local models, on the other hand, may be too good at capturing such details but they may suffer from under-utilization, because some of them may be associated to unimportant regions of the system dynamics. Regional modeling [15] comes out as an alternative modeling approach specifically designed to find a reliable trade-off between global and local techniques.

The basic idea behind regional models consists in dividing the space of the explanatory variables, the input space hereafter, into a number of partitions and merging them into larger ones, called regions, where the system dynamics are modelled. In its basic formulation, the partitioning of the input space is achieved using the SOM, and the K-means algorithm applied to SOM's weight-vectors is used to merge them. The system dynamics in each region are then reconstructed using independent regression models. In our experiments, we used OLS regression and the Extreme Learning Machine.

The first step for building a regional model consists in learning a SOM on a set of input data $\mathbf{X} = \{\mathbf{x}_l\}_{l=1}^{N}$, $\mathbf{x}_l \in \mathbb{R}^{p+q}$ that maps them onto a low-dimensional lattice of M neurons which are usually arranged as a rectangular 2-dimensional array. The set of weight vectors $\mathbf{W} = \{\mathbf{w}_m\}_{m=1}^{M}$, with $\mathbf{w}_m \in \mathbb{R}^{p+q}$, and their corresponding coordinates $\mathbf{r}_m \in \mathbb{R}^2$ in the lattice characterize the trained SOM.

Once the SOM is trained, the second step consists in finding L partitions $\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_L$ (with $L \leq M$) of the input data by assigning a region l to each neuron of the SOM. This is carried out using the K-means algorithm over the SOM weight vectors $\mathbf{W}$ and the optimal number L of clusters, and thus also regions, is selected by minimizing the Davies-Bouldin index[1], (DB) [10]:

$$L = \underset{K=1,\ldots,M}{\operatorname{argmin}} \; DB(\mathbf{W}, \mathbf{P}^K), \qquad (3)$$

where $\mathbf{P}^K = \{\mathbf{p}_k\}_{k=1}^{K}$ with $\mathbf{p}_k \in \mathbb{R}^{p+q}$ denotes the set of K-means prototypes. Once the optimal L is determined, the l-th region is comprised of all weight vectors $\mathbf{w}_m$ that are associated with the prototype $\mathbf{p}_l$, that is

$$\mathbf{W}_l = \{\mathbf{w}_m \;\mid\; \|\mathbf{w}_m - \mathbf{p}_l\| < \|\mathbf{w}_m - \mathbf{p}_j\|, \quad \forall j \neq l, j = 1, \ldots, L\}, \qquad (4)$$

and the set $\mathbf{X}_l$ of input vectors whose closest SOM weight vector belongs to $\mathbf{W}_l$.

[1] The smallest DB index value is considered the best choice based on the criterion of constructing clusters with low intra-cluster distances and high inter-cluster distances.

The third and last step consists in training the L regional regression models using the inputs $\mathbf{X}_l$ and their corresponding target $\mathbf{t}_l$. This step is general and any type of regression technique can be considered for the task. The estimate $\hat{u}(t)$ for a previously unseen input signal $\mathbf{x}(t)$ is simply obtained from the model associated to the so called *winning region*, $l^*(t)$, that is the region such that

$$l^*(t) = \underset{l=1,\ldots,L}{\operatorname{argmin}} \|\mathbf{x}(t) - \mathbf{p}_l\|. \tag{5}$$

For regional linear models, one can simply consider OLS regression and estimate the coefficient vector $\hat{\boldsymbol{\beta}}_l = (\mathbf{X}_l^T \mathbf{X}_l)^{-1} \mathbf{X}_l \mathbf{t}_l$, for each $l = 1, \ldots, L$ region. The target of a new input $\mathbf{x}(t)$ is estimated as $\hat{u}(t) = \hat{\boldsymbol{\beta}}_{l^*}^T \mathbf{x}(t)$, with $\hat{\boldsymbol{\beta}}_l^*$ the coefficient vector of the winning region.

In this paper, we denote by *Regional Linear Model* (RLM) a regional model built using linear models. By the same token, we will use the term *Regional Extreme Learning Machine Models* (RELM) to denote regional models comprised of ELM networks.

The ELM is a single-hidden layer feedforward network (SLFN), proposed by [8], in which the weights from the inputs to the hidden neurons are randomly chosen, while the weights from the hidden neurons to the output are analytically determined. Hidden layer needs not to be tuned, and its parameters are independent from training data. Consequently, ELM offers significant advantages such as fast learning speed, ease of implementation, and least human intervene when compared to more traditional SLFNs, such as the MLP and RBF networks. The use of ELM networks here is due to its fast learning training and capacity of fit more complex datafold than linear models. It is feasible because some clusters in RM can be made more complex than when we use purely local modeling.

3 Robust Regression and Regional Modeling

A common feature of the aforementioned RM models is that they usually use the OLS technique for parameter estimation. In spite of the importance and wide application of the OLS method, a notable drawback is the assumption that errors follow a normal distribution. Unfortunately, in many real-world situations this assumption does not hold, thus affecting the reliability of the results. According to [1], the LSE method is very far from optimal in many non-Gaussian situations with heavy tails caused by outlying measurements, for example.

OLS computes the square sum of residuals between desired solution and solution computed by the system and, all points have the same importance in this computation. This type of computation, however, is not appropriate in the presence of outliers. Therefore, it is important to have a solution that is offers protection against such observations, like robust fitting methods. Huber [9] introduced the concept of M-estimation, where M stands for "maximum likelihood type". He suggests to obtain a more robust regression method by minimizing another function of the errors than the sum of their squares. Based on Huber theory, a general M-estimator minimizes the following objective function:

$$\sum_{i=1}^{n} \rho(e_i) = \sum_{i=1}^{n} \rho(t_i - \mathbf{x}_i^T \hat{\boldsymbol{\beta}}) \tag{6}$$

where the function ρ gives the contribution of each residual to the objective function, t_i is the desired value, $\mathbf{x}_i$ is the input, and $\hat{\boldsymbol{\beta}}$ is the parameter to be found by regression. The OLS method is a particular M-estimator, where $\rho(e_i) = e_i^2$. However, M-estimators can replace the squared sum of residual by another function. According to [7], ρ should have the following properties: *i)* $\rho(e_i) \geq 0$, *ii)* $\rho(0) = 0$, *iii)* $\rho(e_i) = \rho(-e_i)$ and *iv)* $\rho(e_i) \geq \rho(e_{i'})$ for all $|e_i| > |e_{i'}|$.

Parameter estimation is defined by the estimating equation which is a weighted function of the objective function derivative. Let $\psi = \rho'$ to be the derivative of ρ. Differentiating ρ with respect to coefficients $\hat{\boldsymbol{\beta}}$, we have

$$\sum_{i=1}^{n} \psi(t_i - \mathbf{x}_i^T \hat{\boldsymbol{\beta}})\mathbf{x}_i^T = 0. \tag{7}$$

Then, defining the weight function $g(e) = \psi(e)/e$, and let $g_i = g(e_i)$, the estimating equations are given by

$$\sum_{i=1}^{n} g_i(t_i - \mathbf{x}_i^T \hat{\boldsymbol{\beta}})\mathbf{x}_i^T = 0. \tag{8}$$

Solving the estimating equations is a weighted least-squares problem, minimizing $\sum_{i=1}^{n} g_i^2 e_i^2$. The weights, however, depend upon the residuals, the residuals depend upon the estimated coefficients, and the estimated coefficients depend upon the weights. In this case, an iterative solution like the iteratively reweighted least-squares, IRLS [7], is therefore required.

In this paper, we use the Huber estimator as learning method of the regional models, where the weight function is given by:

$$g(e_i) = \begin{cases} k/|e_i|, & \text{if } |e_i| > k \\ 1, & \text{otherwise.} \end{cases}$$

with tuning constant $k = 1.345\sigma$, where σ corresponds to the standard deviation of the residuals. It produces around 95-percent efficiency when the errors are normal, and still offers protection against outliers.

Henceforth, the resulting models will be generically referred to as *Robust Regional Models* (RRM). The performances of the proposed RRM methods will be evaluated on two benchmarking real-world datasets.

4 Computer Simulations

In this section, we present the results of computer simulations carried out with two input-output datasets commonly used for benchmarking purposes in system identification. The performances of the proposed Robust RLM (or R2LM) and Robust RELM (or R2ELM) models are compared to those achieved by six other models: the regular RLM and RELM models, an ELM-based global model, the KSOM model [4], the LLM model [17] and a global linear model adapted by OLS regression.

All the models are evaluated via the statistics of the resulting normalized mean-squared estimation error $NMSE = \frac{\sum_{t=1}^{N} e^2(t)}{N \cdot \hat{\sigma}_u^2}$, where $\hat{\sigma}_u^2$ is the variance of the original time series $\{u(t)\}_{t=1}^{N}$ and N is the length of the sequence of residuals. Two benchmarking datasets[2] were used to evaluate all the models, namely: (i) the hydraulic actuator dataset (input u: valve position, output y: oil pressure), and (ii) the flexible robot arm dataset (input u: reaction torque of the structure, output y: acceleration of the flexible arm).

4.1 Results on the Hydraulic Actuator Dataset

The models were trained using the first 384 samples (training set) of the input/output time series, while the following 112 samples (validation set) were used for validation purposes. The models were evaluated with the remaining 512 samples (testing set). Input/output time series are rescaled to the $[-1, +1]$ range. Memory orders were set to $p = 5$ and $q = 4$, respectively. For each SOM-based model, the number of neurons was set to $M = 20$. The initial and final learning rates are set to $\alpha_0 = 0.5$ and $\alpha_T = 0.01$. The initial and final values of the gaussian neighborhood function radius were $\sigma_0 = M/2$ and $\sigma_T = 0.001$. The learning rate α' (LLM model) was set to 0.01. The optimal number of hidden neurons of the ELM-based global model was found by searching from 2 to 30 for the value that achieved the smallest NMSE on the validation set. The best result was found for 20 hidden neurons. The number of hidden neurons of the ELM-based regional models (RELM and R2ELM) was set to 10, i.e. half the number of hidden neurons used by the ELM-based global model.

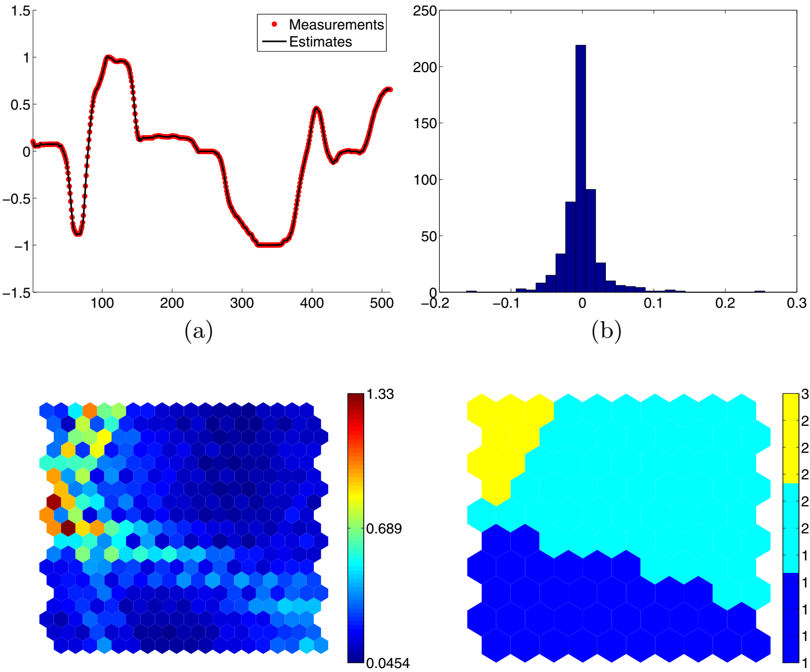
Table 1. Performance results for the hydraulic actuator data

Models	**NMSE**			
	mean	*min*	*max*	*variance*
RLM	1.14e-004	1.13e-004	1.38e-004	2.09e-011
RELM	1.14e-004	1.13e-004	1.27e-004	9.25e-010
R2ELM	1.18e-004	1.17e-004	1.29e-004	4.32e-012
R2LM	1.22e-004	1.17e-004	2.78e-004	5.25e-010
ELM	0.0012	0.0001	0.0026	1.04e-007
KSOM	0.0019	0.0002	0.0247	1.15e-005
LLM	0.0347	0.0181	0.0651	1.58e-004
Linear	0.0380	—	—	—

[2] Download from `http://homes.esat.kuleuven.be/~smc/daisy/daisydata.html`.

The regional models were comprised of SOM grids with 10×10 neurons, using the aforementioned parameters. The number of clusters found by the K-means algorithm in combination with the minimum DB index was $L = 3$. The obtained results are shown in Table 1, where are displayed the mean, minimum, maximum and variance of the NMSE values, collected over 50 training/testing runs, with the weights randomly initialized at each run. In this table, the models are sorted according to the mean NMSE values.

One can easily note that the performances of the regional approaches on this real-world dataset are better than those of all other models. The ELM and KSOM model also had acceptable performances on this dataset. We can see that without an appropriate selection of the number of models, the LLM presents an accuracy simliar to that of the global linear method.



(a) (b)

(c) (d)

Fig. 1. (a) Typical estimated sequences of the valve position provided by the R2ELM model. Solid line indicates the estimated sequence. (b) Histogram of residuals. (c) Associated U-Matrix. (d) Clusters of SOM prototypes for $L = 3$.

The proposed models (R2LM and R2ELM) achieved performances which are comparable to those of regular regional models (RLM and RELM), but still far better than the remaining models. In this regard, since the performances of the four regional models (R2LM, RLM, R2ELM and RELM) are statistically equivalent, we highly recommended the use of the proposed robust variants in real-world applications since they potentially handle better datasets with outliers and nongaussian noise.

Figure 1 shows typical results provided by the R2ELM model. Figure 1 (a) shows the sequence generated by the R2ELM model, where the actual and estimated sequences are almost indistinguishable because of the small estimation error. Figures 1 (c) and (d) show, respectively, the U-matrix associated with the trained SOM and the clusters of SOM prototypes found by the K-means algorithm using $K = L = 3$ as indicated by the DB index.

4.2 Results on the Robot Arm Dataset

In the following experiments, the number of neurons for the KSOM and LLM models was set to $N = 30$. For each SOM-based model, the initial and final learning rates were set to $\alpha_0 = 0.5$ and $\alpha_T = 0.01$. The initial and final values of radius of the neighborhood function are $\sigma_0 = N/2$ and $\sigma_T = 0.001$, and the learning rate α' (LLM model) was set to 0.1. The regional models were composed of SOM grids with 10 × 10 neurons, using the aforementioned parameters. The optimal number of clusters found was $L = 9$. Finally, the optimal number of hidden neurons found for the ELM global model on the validation set was 30, after a systematic search within the range from 2 to 50. The ELM-based regional models used then 15 hidden neurons.

The obtained results are shown in Table 2, where are displayed the mean, minimum, maximum and variance of the NMSE values, collected over 50 training/testing runs, with the weights randomly initialized at each run. In the table, the models are sorted according to the mean NMSE values.

Table 2. Performance results for the robotic arm data

Models	**NMSE**			
	mean	*min*	*max*	*variance*
RELM	0.0053	0.0051	0.0055	1.03e-008
R2ELM	0.0054	0.0052	0.0056	1.61e-008
RLM	0.0057	0.0053	0.0062	4.08e-008
R2LM	0.0060	0.0056	0.0066	5.66e-008
KSOM	0.0064	0.0045	0.0117	1.83e-006
ELM	0.0285	0.0171	0.0457	2.73e-005
LLM	0.3176	0.2685	0.3558	2.23e-004
Linear	0.3848	0.3848	0.3848	-

For this dataset, the best performances were achieved by the R2ELM and RELM approaches. The KSOM (a local linear model) and the two regional linear models (i.e. RLM and R2LM) models also achieved good accuracies, but the nonlinear regional models (i.e. RELM and R2ELM) acieved smaller variances than all the other models. This represents a case study where regional nonlinear models achieved better results than regional or local linear models. The LLM and global-linear models presented the worst overall performances.

5 Conclusions

We have introduced an extension to the recently proposed regional models [15], named robust regional models, for dynamical system identification. An evaluation of the proposed approach was carried out for the task of inverse system identification of two benchmarking dynamical systems. Their performances were compared to those achieved by regular regional models, by standard local linear models, and by linear/nonlinear global models.

The main general conclusion of the presented experiments is that robust regional models can be considered a promising approach for nonlinear dynamical system identification, especially in presence of outliers and nongaussian noise. They presented good performance results when compared to other traditional modeling methods.

Currently, we are working on a real application for the identification of the wastewater treatment plant of Helsinki (Finland). Variants of regional models including those with heterogeneous models and different metrics of evaluation have been developed.

References

1. Andrews, D.F.: A robust method for multiple linear regression. Technometrics 16(4), 523–531 (1974)
2. Azman, K., Kocijan, J.: Dynamical systems identification using gaussian process models with incorporated local models. Engineering Applications of Artificial Intelligence 24(1), 398–408 (2011)
3. Barreto, G.A., Araújo, A.F.R.: Identification and control of dynamical systems using the self-organizing map. IEEE Transactions on Neural Networks 15(5), 1244–1259 (2004)
4. Barreto, G.A., Souza, L.G.M.: Adaptive filtering with the self-organizing maps: A performance comparison. Neural Networks 19(6), 785–798 (2006)
5. Chen, J.-Q., Xi, Y.-G.: Nonlinear system modeling by competitive learning and adaptive fuzzy inference system. IEEE Transactions on Systems, Man, and Cybernetics-Part C 28(2), 231–238 (1998)
6. Cho, J., Principe, J., Erdogmus, D., Motter, M.: Quasi-sliding mode control strategy based on multiple linear models. Neurocomputing 70(4-6), 962–974 (2007)
7. Fox, J.: Applied Regression Analysis, Linear Models, and Related Methods. Sage Publications (1997)
8. Huang, G.B., Zhu, Q.Y., Ziew, C.K.: Extreme learning machine: Theory and applications. Neurocomputing 70(1-3), 489–501 (2006)
9. Huber, P.J.: Robust estimation of a location parameter. Annals of Mathematical Statistics 35(1), 73–101 (1964)
10. Jain, A.K., Dubes, R.C., Chen, C.: Bootstrap techniques for error estimation. IEEE Transactions on Pattern Analysis and Machine Ingelligence 9(5), 628–633 (1987)

11. Narendra, K.S., Parthasarathy, K.: Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Networks 1(1), 4–27 (1990)
12. Norgaard, M., Ravn, O., Poulsen, N.K., Hansen, L.K.: Neural Networks for Modelling and Control of Dynamic. Springer (2000)
13. Peng, H., Nakano, K., Shioya, H.: A comprehensive review for industrial applicability of artificial neural networks. IEEE Transactions on Control Systems Technology 15(1), 130–143 (2007)
14. Principe, J.C., Wang, L., Motter, M.A.: Local dynamic modeling with self-organizing maps and applications to nonlinear system identification and control. Proceedings of the IEEE 86(11), 2240–2258 (1998)
15. de Souza Junior, A.H., Barreto, G.A.: Regional Models for Nonlinear System Identification Using the Self-Organizing Map. In: Yin, H., Costa, J.A.F., Barreto, G. (eds.) IDEAL 2012. LNCS, vol. 7435, pp. 717–724. Springer, Heidelberg (2012)
16. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11(3), 586–600 (2000)
17. Walter, J., Ritter, H., Schulten, K.: Non-linear prediction with self-organizing map. In: Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN 1990), vol. 1, pp. 587–592 (1990)
18. Wang, X., Syrmos, V.L.: Nonlinear system identification and fault detection using hierarchical clustering analysis and local linear models. In: 15th Mediterranean Conference on Control and Automation (2007)

Learning Embedded Data Structure with Self-Organizing Maps

Edson C. Kitani[1], Emilio Del-Moral-Hernandez[1], and Leandro A. Silva[2]

[1] University of Sao Paulo, School of Engineering, Av. Prof. Luciano Gualberto,
travessa 3 nº 380 ZIP 05508-010 - SP – Brazil
{ekitani,emilio}@lsi.usp.br
[2] School of Computing and Informatics, Mackenzie Presbyterian University,
Rua da Consolação, 930 - ZIP 01302-907 – SP – Brazil
prof.leandro.augusto@mackenzie.com.br

Abstract. Self-Organizing Map (SOM) is undoubtedly one of the most famous and successful artificial neural network approaches. Since the SOM is related with the Vector Quantization learning process, minimizing error quantization and maximizing topology preservation can be concurrent tasks. Besides, even with some metrics, sometimes the analysis of the map results depends on the user and poses an additional difficulty when the user deals with high dimensional data. This work discusses a proposal of relocating the voted map units after the training phase in order to minimize the quantization error and evaluate the impact in the topology preservation. The idea is to enhance the visualization of embedded data structure from input samples using the SOM.

Keywords: Self-Organizing Map, Manifold Learning, Dimensionality Reduction, Quantization Error, Topology Preservation.

1 Introduction

Self-Organizing Maps (SOM) developed by Kohonen have been studied and applied in different areas, from Robotics to Linguistics applications [1]. Each area creates a specific theory of application, interpretation of results and limitations of the SOM map. In the area named Manifold Learning, SOM is known as a method of predefined lattice and intuitively works as a nonlinear discrete PCA [20]. It is noteworthy explain that predefined lattice are methods that impose in advance a regular structure such as rectangular or hexagonal grid made of regularly or not spaced point [2]. The main interest in Manifold Learning area is to discover intrinsic dimensionality unfolding the data structure in order to represent it in a lower dimension than the original, however, not necessary SOM can accomplish it. PCA is the oldest method applied to uncover intrinsic low embedded dimension structure in high dimensional data. Despite the fact that PCA is a linear approach, it has been used for a long time as preprocessing to dimensionality reduction as a full spectral technique based on eigenvector decomposition of sample covariance matrix of the input data $\mathbf{x}$ [3]. Techniques such

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 225–234.
DOI: 10.1007/978-3-642-35230-0_23

as Sammon's NonLinear Mapping [4], ISOMAP [5], Locally Linear Embedding (LLE), [6] as well as the Self-Organized Manifold Mapping (SOMM), the latter proposed in [7], [8], and [9], belong to this category of nonlinear dimensionality reduction methods.

All the approaches above, except SOMM, work projecting the high dimensional input data to low dimensional space creating a representation that can be understood by humans. In essence, the techniques aim to assist humans to visualize multidimensional structures projecting or mapping them into low dimensional space and when possible, to uncover low dimensional structures embedded into input data. An interesting question arises: how to evaluate the quality of the projection and how to be sure that it provides correct results, if we are not able to visualize the original structure and compare the results?

For SOM maps, several metrics and methodologies were proposed, such as Quantization Error [1], Topographic Error [10], Topographic Product [11], Topographic Function [12], Trustworthiness and Neighborhood Preservation [13] and Distortion Measure [1], [14], and the last one is used in this work. Even with such an amount of metrics to measure or evaluate the quality of SOM mapping, there are some situations in which results do not match with real situation. An experiment with a classical toy example will be discussed herein highlighting the tradeoff between how to reduce quantization error while preserving the topology of the map. In order to understand the results calculated from these metrics, this work used a Swiss Roll toy example to train the SOM maps. The visual and numerical results were analyzed in order to create insight, to understand and to improve the quality of the SOM map after training.

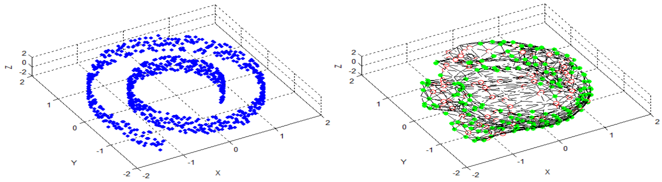
This paper is organized as follows: in Section 2, a brief review of the Distortion Measure method is presented. In Section 3, computational experiments with the new proposal on how to relocate the neurons are discussed. Finally, Section 4 discusses the results and concludes the paper.

2 Distortion Measure and Neuron Relocation

As mentioned before, the SOM maps tries to discover the structure where the input data lies. It works contracting or stretching the grid over the input manifold to the end of the training phase to have a copy of the input structure. The predefined lattice of SOM has advantages and drawbacks. The predefine lattice creates a neighborhood relationship between each neuron and this connection is the key for topology maintenance. In order to keep this link, the map can fold over itself or create connection in space where data are not defined.

A famous example of this situation explored extensively in the literature is the Swiss Roll data structure. Despite the simplicity of its visualization, the Swiss Roll has some challenges to be dealt with the SOM algorithm. First, in the Swiss Roll the data are scattered in a tight structure and second, it is a non-convex structure. As SOM preserves the neighborhood relation between neurons on the grid, it creates shortcuts inside the represented structure.

Figure[1] 1 illustrates the situation described above, in which a square 20×20 SOM map with hexagonal neighborhood was trained using the SOMToolbox released by [15]. In (a) a Swiss Roll distribution format with 900 points with zero mean and unitary variance randomly distributed along the structure was used to train the SOM map. In (b), the structure learned by the SOM after the training phase. The green dots represent neurons that became a BMU (Best Matching Unit) during training phase and red circles are neurons without votes. The lines in black represent the grid connections.



(a) (b)

Fig. 1. Swiss Roll distribution structure with 900 random points in (a). In (b) the neurons of the SOM map after the training phase with Swiss Roll data distribution. The neurons that became BMU are marked in green1 and the neurons in red1 did not win the competition during the training phase.

Two basic quality metrics well known by the SOM community and implemented in the SOMToolbox [15] are Quantization Error [1] and Topographic Error [10]. After training a SOM with the Swiss Roll distribution, the outcomes were the SOM map illustrated in Figure 1 (b) and numeric results for QE = 0.2275 and TE = 0.0289. Only the numbers do not provide a reasonable interpretation of the maps result. However, a visual inspection of the map does not provide a correct idea of the structure that SOM is trying to represent. The final map is a consequence of the training rule defined by Kohonen as $\mathbf{m}_j(t+1) = \mathbf{m}_j(t) + \alpha(t)hci_j[\mathbf{x}_i - \mathbf{m}_j(t)]$ and the predefined lattice. It means that, every time (t+1), neuron j is updated based on $[\mathbf{x}_i - \mathbf{m}_j(t)]$ weighted by a learning factor α and a neighborhood function hci between neuron j and the best matching unit ci for the input sample $\mathbf{x}_i$ given by $ci = argmin\{\|\mathbf{x}_i - \mathbf{m}_j\|^2\}$.

The map with hexagonal topological neighborhood, in this example a 20×20 square format, and the neighborhood function hc_i controls the "movement" of the neurons along the training phase. In other words, while the input sample $\mathbf{x}_i$ is pulling the winner neuron ci, the neighborhood function is influencing the rest of neurons j according to the Kohonen's training rule. Considering the format of map and learning rule, it is reasonable to have neurons allocated in the middle of the Swiss Roll structure, even in an empty space. However, what happens when a high dimensional data set is trained without any previous knowledge of its real structure?

We here address an alternative way to deal with this problem. The proposal is to run the traditional SOM algorithm and, at the end of training, to relocate the neurons j with individual quantization error qe_j greater than global quantization error QE. The

[1] Please, access the digital version of this paper to visualize all colored images.

new weight vector $\mathbf{m}_j$ will assume the mean value of the $\mathbf{x}_i$ inputs allocated to neuron *j*. Then, a new global QE is calculated and another evaluation regarding local quantization error is made. The process stops when the difference between the new QE and the previous QE is lower than a threshold *qeth*. The algorithm of this process presented in Table 1 can clarify the proposal. The motivation for this neuron relocating strategy is based on Kohonen's books [1] "*… the best map is expected to yield the smallest average quantization error…*". However, the relocating is conducted without losing the neighborhood concept.

The discussion here is how to have a tradeoff of good representation based on minimizing quantization error and preserving the topology. As the SOM works based on vector quantization, the learning rule needs to deal with two concurrent forces. One of the parameters that control those opposite forces is the neighborhood kernel hc_i. This is one of the reasons why all final maps usually have a contracted form as compared to the original structure. Additionally, the final map has a border effect in which each neuron represents more samples than the neurons in regions with dense input data [1].

Taking all effects discussed above, we understand that relocating some neurons based on the strategy presented here it will improve the quality of the representation of the SOM map. Besides, the relocation could avoid the representation of empty spaces as illustrated in the Swiss Roll example and reduce misleading interpretation of the map's result. If this effect happens in low dimension, it will naturally happen in high dimensional space and with additional difficulty, one cannot visualize these effects.

Table 1. Algorithm to relocate winner neurons

```
a)Calculate the SOM map composed of $k$ neurons using Kohonen´s algorithm.
b)Calculate the global $QE = \frac{1}{k}\sum_{j=1}^{k}\frac{1}{N}\sum_{i=1}^{N}\|x_i - m_j\|$; $N$ samples $x_i$ assigned to each winner neuron $j$ during training phase;
c)Calculate the threshold $qeth = 1\%\ of\ global\ QE$;
d)While$(QE^t - QE^{t-1}) > qeth$
  d.1) Calculate for each winner neuron $j \mid C_j \geq 2$, $qe_j = \frac{1}{c}\sum_{i \in \{C_j\}}\|x_i^j - m_j\|$,
       where c=$size\ C_j$;
  d.2) Calculate $\bar{x}_j = \frac{1}{c}\sum_{i \in \{C_j\}}\{x_i^j\}$;
  d.3) If $qe_j > QE$ then $mj = \bar{x}_j$;
  d.4) Calculate a new global $QE^t$;
  e)end
```

The algorithm presented in Table 1 relocates only neurons *j* with local quantization qe_j error higher than global error QE and only for those neurons that represent more than one input sample. Our proposal will privilege neurons at the border of the map and reduce the distortion of the map. The step (d) is similar to the "Batch Map Algorithm" proposed by Kohonen [1]. However, instead of updating all neurons our approach only relocates some neurons to the centroid of respective input samples, in order to preserve as much as possible the final ordered map defined by the SOM training process. Figure 2 presents the results of the algorithm applied over the SOM map after training it with the Swiss Roll data set. Figures 2 (a) and (b) represent a 3D view of the final structure learned, original and relocated position, respectively. In figures

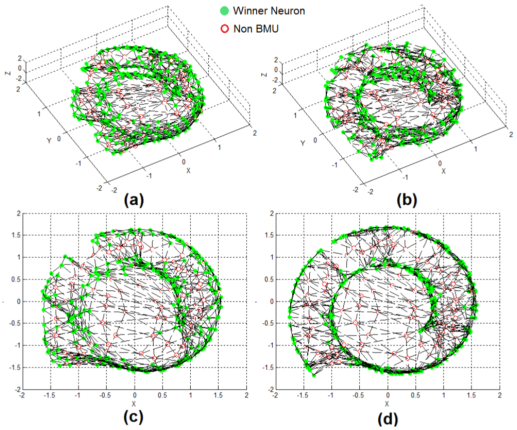


Fig. 2. In figure (a) 3D view and in (c) top view of the structure represented by a 20×20 SOM map. In (b) a 3D view and in (d) top view of the structure represented by a relocated SOM map.

(c) and (d) a top view is shown and the distributions of the winner neurons are visibly more reasonable in figure (d).

In order to avoid only visual inspection and not restricted to the simple QE metrics, this work conducted a set of experiment varying the size and format of the map for the same input sample structure. We used a metric named Distortion Measure in order to evaluate the impact of our relocating process for SOM using more accurate measurement quality [17]. The idea of using SOM Distortion Measure error (DM in short) is to decompose the error into three factors and evaluate which one has more impact on the final result. The DM error can be defined as:

$$E_d = \sum_{i=1}^{N}\sum_{j=1}^{k} hc_{i,j}\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^2. \tag{1}$$

where $hc_{i,j}$ is the value of the neighborhood function between unit map j and BMU $c_i = argmin_j\left\{\left\|\mathbf{x}_i - \mathbf{m}_j\right\|^2\right\}$. If neighborhood $hc_{i,j} = 1$, E_d became the sum of all quantization errors between all input samples and neurons. The equation (1) can be expressed as a summation of three main components as was proposed in (14) and reproduced in equation (2)

$$E_d = Eqx + Env + Enb \tag{2}$$

The first term *Eqx* represents the quantization error based on the Euclidean distance from a set of input sample $\mathbf{x}_i^j$ and the Voronoi centroid $\mathbf{n}_j = {}^1\!/_{\boldsymbol{N_j}} \sum_{x_i \in V_j} \mathbf{x}$ defined by neuron j and $V_j = \{x_i |\ \|x_i - m_j\| < \|x_i - m_k\| \forall k \neq j\}$. In other words, each winner neuron j defines approximately the Voronoi tessellation of the input space and *Eqx* calculates the distance from each input sample to the Voronoi centroid $\mathbf{n}_j$ and not to neuron $\mathbf{m}_j$. The idea is to measure the average of local variance of the input samples that belongs to Voronoi set V_j defined by each neuron j. Then, *Eqx* can be expressed as:

$$Eqx = \sum_{j=1}^{k} N_j H_j \left(\Sigma \mathrm{x} \in \mathrm{v} j || \mathbf{x} i - \mathbf{n} j ||^2 / Nj\right). \tag{3}$$

where $H_j = \Sigma^k_{i=1}(hc_{i,}j)$ is the accumulated neighborhood factor of each neuron i to neuron j and H_j is considered the weighting factor to compensate the border effect. Another advantage here is the possibility to evaluate the influence of each unit j and its H_j to the amount of the final distortion error. The second term *Env* calculates the variance between each neuron j and measure how close it is to each other.

$$Env = \sum_{j=1}^{k} N_j H_j Var_h\{\mathbf{m}|j\}, \tag{4}$$

$$Var_h\{\mathbf{m}|j\} = \sum_g \frac{hc_{jg} \|\mathbf{m}_g - \bar{\mathbf{m}}_j\|^2}{H_j}, \tag{5}$$

$$\bar{\boldsymbol{m}}_j = \sum_g \frac{h_{jg}\mathbf{m}_g}{H_j} \quad g=1,2,3,\ldots,k. \tag{6}$$

The low value is obtained when the neurons are close to each other, so that the number of neurons and format of the grid have an influence on it. Finally, the third term *Enb* measures a combination of quantization and projection quality. As pointed out by Kohonen [1], there are two main forces acting on SOM's neurons. First, the neuron tends to represent the density function of the input space and second, the neighborhood relation tends to preserve the continuity of the grid. Both forces are concurrent and this results in a smooth surface that tries to imitate the original space. Based on the assumption that each neuron tessellates the input space, the position of each Voronoi centroid $\mathbf{n}_j$ of this mosaic will depend on how spread the neurons are along the input space. If $\mathbf{n}_j = \mathbf{m}_j$ we have a regular lattice form. Then, the term *Enb* combines the effect of the number of neurons and their concentration along the input space. It can be observed that each factor is the sum of error measured per neuron. Thus, it is possible to identify which neuron or group of neurons has an influence on the total distortion value [14].

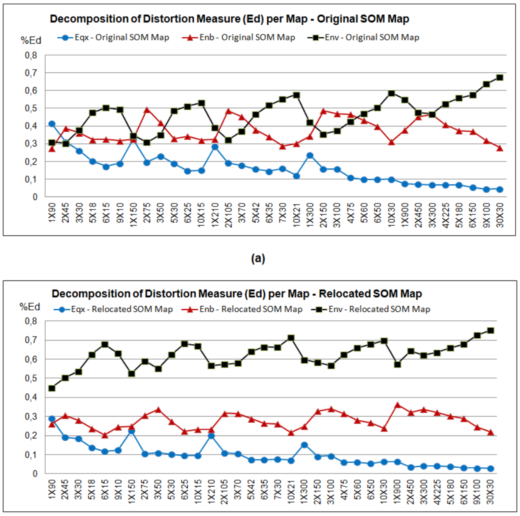
$$Enb = \sum_{j=1}^{k} N_j H_j \|\mathbf{n}_j - \bar{\mathbf{m}}_j\|^2. \tag{7}$$

3 Experiments and Results

The proportional amount of each error factor described above is measured here as a percentage of its contribution to the total *Ed*. The parameters of the SOM training was chosen according to the following: Numbers of neurons vary from 64 to 900 and several map combination and lateral dimensions to reach that number, hexagonal neighborhood, training length rough = 1000 and fine = 10000, linear initialization, one and two dimensional formats and all the rest of parameters were keep *default* as defined in SOMToolbox package. One additional map format was trained with 100×30 neurons. After the training phase of each map, neurons were relocated using the algorithm described in Table 1.

Figure 3 presents the result of 34 maps trained with the Swiss Roll data and the rate of each distortion error was plotted in the graphic (a) for original SOM map and in (b) the relocated SOM map. The total *Ed* Distortion Measure value was not presented in absolute numbers, since it varies with the size of the map and it is not useful to compare different maps [16]. Thus, the percentage of distortion contribution per factor was compared. Considering that the number of neurons of each group of maps does not change, but only the format (length and width), the error *Eqx* should keep constant. However, depending on the format of the map, error grows.

According to these results, tinny maps, linear or close to linear, increase the quantization error because the initial map does not cover all input space and during the training process the strip of neurons try to move along the space but is limited by two neighborhood relationship (ahead and back). On the other hand, even with tinny maps, after relocating some neurons it is possible to reach low and constant values along several maps configuration.



(a)

(b)

Fig. 3. Curves of the distortion components (*Eqx*, *Enb* and *Env*). In (a) curves related with the original SOM maps and in (b) for relocated SOM maps.

The curve of the *Env* factor shows that the format of the map (rectangle or square) has a direct impact on results. As *Env* measures the trustworthiness of the map, considering the topology of the neurons and how smooth they are, square maps will fold a lot in the Swiss Roll structure. Then, the final result is not a smooth map and the factor will capture this impact. Thus, in order to have low impact of *Env* error, the grid must have a regular and compact format. However, in relocated maps the variance $Var_h\{\mathbf{m}|j\}$ will be higher than in the original map, affected by the relocation of the neurons. The *Enb* factor shows the combination of the previous two factors, quantization and smoothness of map surface. Observe in Figure 3 that the number of neurons has a direct impact on all the factors, but in all of these tests the number of neurons was kept constant. Then, the results indicate that the map format has a contribution to the quality results. In this case, large maps allow a good tessellation of the input space and adequately smooth the surface of the map. A reasonable interpretation of the *Enb* factor is considering it as a stress between the quantization error and topology preservation [17].

The results of the experiments with Swiss Roll indicate that the low contribution of *Eqx* and *Enb* create better representation of the input structure. Inevitably, as the sum of the three factors must be one, the factor *Env* will naturally rise. Additional experiments were conducted with 12 squared maps (from 8×8 to 30×30) and the each DM factor per map was plotted. The error contributions by each factor for square maps and the map with the size 100×30 are illustrated in Figure 4.

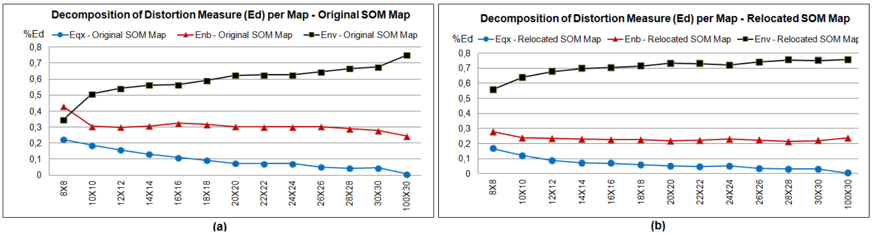


Fig. 4. The curves (a) – original SOM Map – and (b) – relocated SOM Map – illustrate the contribution of the error factor along 12 different square SOM maps and the last one for a map of 100×30 in size.

Figures 4 at (a) and (b) illustrate the error contribution by each factor for square maps, except the last with size 100×30. The map with 3000 neurons can reach a good representation with minimal distortion. In spite of the minimal distortion in quantization and topology, in the final training phase it 2187 neurons were left without representation (circles in red) occupying space in memory and computation time during the training phase. The curves at Figure (4) (b) shows that *Enb* factor has low contribution and low variance along each group of maps, indicating that the relocated maps have a good balance between quantization error and topology preservation.

The question that arises is how to reach similar results on low error without spending so much time and memory? One way is to reduce the number of neurons and just relocate the neurons, as proposed in this work. Figure 5 shows the visual results of 3 selected size maps, (a) 8×8, (b) 3×100 and (c) 30×30. Original SOM maps and Swiss Roll data distribution superimposed on it are shown at the top and relocated SOM maps at bottom. The visual quality of the representation is higher at relocated maps than in original ones and it coincides with the *Eqx* and *Enb* as the quality measures indicate a good representation of input data structure, even in maps with low number of neurons. As the relocating processes are performed only for those neurons with absolute quantization error (QE) higher than a threshold, this strategy relocates the neuron j to the Voronoi centroid $\mathbf{n}_j = {}^{1}/_{\boldsymbol{N_j}} \sum_{x_i \in V_j} \mathbf{x}$ defined during the learning process. Then, this process maximizes the probability density at the Voronoi region defined by the neurons [18]. Improving the trustworthiness of the SOM map, it is possible to understand the neurons of the SOM map as the "the backbone" of the data structure and using them as reference support to "walk over the data manifold". The methodology SOMM [9] uses this strategy and cuts the manifold and creates a low dimensional representation of how possibly the data lied on it. When the user works with high dimensional data, such as images, the SOM map alone will not give a good visualization of the input structure. Additional techniques will be necessary, such as U-Matrix [19] or Sammon Projections [4].

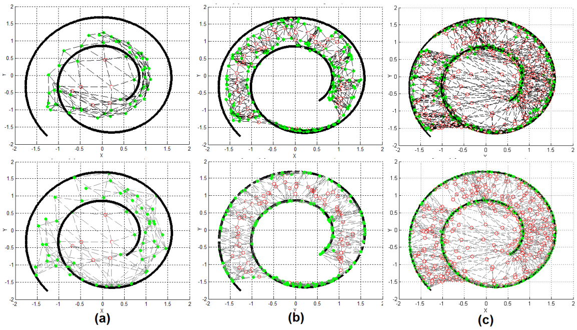


Fig. 5. Top view of the structure learned by 3 SOM maps with different size and format, (a) 8×8, (b) 3×100 and (c) 30×30, respectively. The images at top are related with original SOM maps and at bottom, the relocated SOM maps. The curves in black are the original data set overlapped on final SOM map. The green circles indicate the winner units. The red circles are units without voting.

4 Conclusion and Discussion

As presented in this work, sometimes the results of the SOM map can be misleading concerning of the original data structure. As SOM works with predefined lattice and several free parameters, the training process can be conducted in different ways until reaching the final result. SOM surely learns the input structure, but if a post-process analysis depends on the quality of the map's result to extract information, the approach presented herein can provide an improvement in the final map without losing the main principle regarding SOM, topology preservation. The aim of this proposal is to use the SOM to reduce the dimensionality of the data and after relocate the neurons in order to approximate the SOM map to the original data structure. Then, any post-process technique can work with reduced amount of data but with high quality of map regarding to minimum error quantization and topology preservation. Comparison of different size and format maps using the relative values of *Eqx*, *Enb* and *Env* factor, instead of absolute value of *Ed* Distortion Measure, was presented herein and can be considered as a new way to compare different size and format of SOM maps.

Acknowledgments. The authors are very grateful to the reviewers for their valuable comments, suggestions and corrections.

References

1. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, New York (2001)
2. Lee, J.A., Verleysen, M.: Nonlinear dimensionality reduction. Springer (2010)
3. Maaten, L., Postma, E., Herik, J.: Dimensionality reduction: A comparative review. Tilburg Centre for Creative Computing, pp. 1–33. Tilburg University, Tilburg (2009)
4. Sammon Jr., J.W.: A nonlinear mapping for data structure analysis. IEEE Transaction on Computer, 401–409 (1969)
5. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science Magazine 290, 2319–2323 (2000)
6. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by local linear embedding. Science 290, 2323–2326 (2000)
7. Kitani, E.C., Del-Moral-Hernandez, E., Giraldi, A.G., Thomaz, C.E.: Exploring and understanding the high dimensional and sparse image face space: A self organized manifold mapping. New approaches to characterization and recognition of faces. Intech Open Access Publisher (2011)
8. Kitani, E.C., Del-Moral-Hernandez, E., Thomaz, C.E., Silva, L.A.: Visual Interpretation of Self Organizing Maps. In: Proceedings of the XI Brazilian Symposium on Neural Networks SBRN 2010, pp. 37–42. IEEE CS Press (2010)
9. Kitani, E.C., Del-Moral-Hernandez, E., Silva, L.A.: SOMM – Self-Organized Manifold Mapping. In: Villa, A.E.P., Duch, W., Érdi, P., Masulli, F., Palm, G. (eds.) ICANN 2012, Part II. LNCS, vol. 7553, pp. 355–362. Springer, Heidelberg (2012)
10. Kiviluoto, K.: Topology preservation in Self Organizing Maps, pp. 294–299 (1996)
11. Bauer, H.U., Pawelzik, K.R.: Quantifying the neighborhood preservation of Self-Organizing Feature Maps. IEEE Transactions on Neural Networks 3, 570–579 (1992)
12. Villmann, T., Der, R., Martinetz, T.: A new quantitative measure of topology preservation in Kohonen´s feature maps. In: IEEE World Congress on Computational Intelligence, pp. 645–648 (1994)
13. Venna, J., Kaski, S.: Neighborhood Preservation in Nonlinear Projection Methods: An Experimental Study. In: Dorffner, G., Bischof, H., Hornik, K. (eds.) ICANN 2001. LNCS, vol. 2130, pp. 485–491. Springer, Heidelberg (2001)
14. Vesanto, J., Sulkawa, M., Hollmén, J.: On the decomposition of the Self-Organizing Map distortion measure. In: Proceedings of the Workshop on Self-Organizing Maps (WSOM 2003), pp. 11–16 (2003)
15. Vesanto, J., et al.: SOM Toolbox for Matlab 5, Helsinki University of Technology, Helsinki, pp. 1–60. Report A57 (2000)
16. Pölzbauer, G.: Survey and comparison of quality measures for self organizing maps. In: Paralic, J., Pölzbauer, G., Rauber, A., (ed.), pp. 67–82 (2004)
17. Vesanto, J.: Data exploration process based on the self-organizing map, Computer Science and Engineering, Helsink University. Espoo Finland, Finnish Academies of Technology, Thesis (2002) ISBN - 951-666-596-9
18. Lampinen, J., Oja, E.: Clustering properties of hierarchical Self-Organizing Maps. Journal of Mathematical Imaging and Vision 3, 261–272 (1992)
19. Ultsch, A., Siemon, H.P.: Kohonen's Self Organizing Feature Maps for exploratory data analysis. In: Proceedings of International Neural Network Conference, pp. 305–308 (1990)
20. Haykin, S.: Neural networks: a comprehensive foundation, 2nd edn. Prentice Hall (1999)

Enhancing NLP Tasks by the Use of a Recent Neural Incremental Clustering Approach Based on Cluster Data Feature Maximization

Jean-Charles Lamirel, Ingrid Falk, and Claire Gardent

LORIA, Campus Scientifique,
BP 239, Vandœuvre-lès-Nancy, France
{lamirel,falk,gardent}@loria.fr

Abstract. The IGNGF (Incremental Growing Neural Gas with Feature maximisation) method is a recent neural clustering method in which the use of a standard distance measure for determining a winner is replaced in IGNGF by cluster feature maximization. One main advantage of this method as compared to concurrent methods is that the maximized features used during learning can also be exploited in a final step for accurately labeling the resulting clusters. In this paper, we apply this method to the unsupervised classification of French verbs. We evaluate the obtained clusters (i.e., verb classes) in three different ways. The first one relies on an usual gold standard, the second one on unsupervised cluster quality indexes and the last one on a qualitative analysis. Our experiment illustrates that, conversely to former approaches for automatically acquiring verb classes, IGNGF method permits to produce relevant verb classes and to accurately associate the said classes with an explicit characterisation of the syntactic and semantic properties shared by the classes elements.

Keywords: clustering, NLP, verb classification, feature maximization, incremental learning.

1 Introduction

The IGNGF (Incremental Growing Neural Gas with Feature maximisation) method is a recent neural clustering method in which the use of a standard distance measure for determining a winner is replaced in IGNGF by cluster feature maximization. The method has been shown to outperform other clustering methods for the task of clustering highly multidimensional textual data including multiple topics[1]. Interestingly, another main advantage of this clustering method is that the features used for producing the clusters are also used for labeling those latter. That is, each cluster in the output clustering is labelled with a ranked list of features that best characterises that cluster. An accurate exploitation of such advantage, as compared to other clustering methods, can be found in the domain of automatic classification of verbs. Hence, classifications which group together verbs and a set of shared syntactic and semantic feature have proved useful both in linguistics and in Natural Language Processing tasks. In NLP, the predictive power and the syntax/semantic interface provided by these classifications have been

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 235–244.
DOI: 10.1007/978-3-642-35230-0_24

shown to benefit such tasks as computational lexicography [2], machine translation [3], word sense disambiguation [4] and subcategorisation acquisition [5].

Several methods have thus been proposed to automatically acquire verb classifications, like recently [6, 7]. However, these approaches mostly concentrate on acquiring verb classes that is, sets of verbs which are semantically and/or syntactically coherent. The specific syntactic and semantic features characterising each verb class are usually left implicit: they determine the clustering of similar verbs into verb classes but they do not explicitly label these classes.

In this paper, we present a novel approach to the automatic acquisition of French verb classes based on the IGNGF method which addresses this shortcoming and produces classifications which not only group together verbs that share a number of features but also explicitly associate each verb class with a set of subcategorisation frames and thematic grids characteristic of that class. To acquire a verb classification, we extract the syntactic and semantic features required for learning verb classes from existing lexical resources for French verbs. These include in particular, subcategorisation frames, thematic grids and (English) VerbNet classification [8] class names.

We evaluate the acquired classification both on the clusters (verb sets) it produces and on its cluster labeling i.e., the syntactic and semantic features associated by the IGNGF clustering with the clusters. We perform an evaluation of the verb clusters both by a comparison against an established test set [6] and by the use of our own unsupervised cluster quality indexes. We then carry out a manual analysis of the clusters examining both the semantic coherence of each cluster regarding to its associated features.

The paper is structured as follows. Section 2 introduces the IGNGF clustering algorithm. Section 3 describes the evaluation metrics, the features used for clustering and reports on the results of the clustering focusing on the associations between verbs, subcategorisation frames and thematic grids it provides. Finally, conclusion is drawn.

2 Clustering Algorithm

The IGNGF clustering method is an incremental neural clustering method belonging to the family of the free topology neural clustering methods. The algorithm underlying this clustering method is described in details in [1]. We here briefly summarise the features of that method which are relevant to the present work.

Like other neural free topology methods such as Neural Gas (NG) [9], Growing Neural Gas (GNG) [10], or Incremental Growing Neural Gas (IGNG) [11], the IGNGF method makes use of Hebbian learning for dynamically structuring the learning space. Hebbian learning is inspired by a theory from neurosciences which explains how neurons connect to build neural networks. Whereas for NG the number of output clusters is fixed, GNG adapts the number of clusters during the learning phase, guided by the characteristics of the data to be classified. Clusters and connections between them can be created or removed depending on evolving characteristics of learning (as for example the “age” or “maturity” of connections and the cumulated error rate of each cluster prototype). A drawback of this approach is that clusters are created or removed after a fixed number of iterations yielding clusters which might not appropriately represent complex or sparse multidimensional data. With the IGNG clustering method this issue is addressed by allowing more flexibility when creating new clusters: a cluster is added whenever the distance of a new data point to an existing cluster is above a predefined global threshold, the average distance of all the data points to the centre of the data set. The clustering process thus becomes incremental: each incoming data point (verb in our setting) is considered as a potential cluster. At each iteration over all the data points, a data point is connected with the "closest" clusters and at the same time interacts with the existing clustering by strengthening the connections between these "closest" clusters and weakening those to other, less related clusters. Because of these dynamically changing interactions between clusters, these methods are "winner take most" methods in contrast to K-means (for example), which represents a "winner-take-all" method. The notion of "closeness" is based on a distance function computed from the features associated to the data points.

IGNGF uses the Hebbian learning process as IGNG, but the use of a standard distance measure as adopted in IGNG for determining the "closest" cluster is replaced in IGNGF by feature maximisation. Feature maximisation is a cluster quality metric which favours clusters with maximum feature F-measure. *Feature F-measure* (FF) is the harmonic mean of *feature recall* (FR) and *feature precision* (FP) which in turn are defined as:

$$FR_c(f) = \frac{\sum\limits_{v \in c} W_v^f}{\sum\limits_{c' \in C} \sum\limits_{v \in c'} W_v^f}, \qquad FP_c(f) = \frac{\sum\limits_{v \in c} W_v^f}{\sum\limits_{f' \in F_c, v \in c} W_v^{f'}}$$

where W_x^f represents the weight of the feature f for element x (1 or 0 in the case of our application) and F_c designates the set of features associated with the verbs occuring in cluster c. A feature is then said to be maximal for a given cluster iff its feature F-measure is higher for that cluster than for any other cluster. Finally the Feature F-measure FF_c of a cluster $c \in C$ is the average of the Feature F-measures of the maximal features for c:

$$FF_c = \frac{\sum\limits_{f \in F_c} FF_c(f)}{|F_c|} \tag{1}$$

With feature maximisation, the clustering process is roughly the following. During learning, an incoming data point v is temporary added to every existing cluster, its feature profile is updated (i.e. each cluster is associated with its maximal features) and its average Feature F-measure is computed. Then the winning cluster is the cluster which maximises the distance κ given in Equation (2).

$$\kappa(c) = \mathit{\Delta}(FF_c) * |F_c \cap F_v| - \frac{EucDist(\boldsymbol{c}, v)}{weight} \tag{2}$$

where $\mathit{\Delta}(FF_c)$ represents the gain in Feature F-measure for the new cluster and $F_c \cap F_v$ are the features shared by cluster c and the data point v. This way, those clusters are preferred which share more features with the new data point and clusters which don't have any common feature with the data point are ignored. The gain in Feature F-measure multiplied by the number of shared features is adjusted by the euclidean distance of the new data point v to the cluster centroid vector $\boldsymbol{c}$. Thus, the smaller the euclidean distance to the cluster, less the κ value decreases. The influence of the euclidean distance can be parametrised with a $weight$ factor ($\sqrt{2}$ for this application). Clusters with negative κ score are ignored. The data point is then added to the cluster c with maximal $\kappa(c)$ and the connections to the winner and its neighbours are updated.

Cluster Labeling. Cluster labeling, i.e., the process of associating the clusters with features considered relevant with respect to the application, has proven promising both for visualising clustering results and for validating or optimising a clustering method [12]. As mentioned above, IGNGF clustering associates each cluster with the set of features representative of that cluster, which are the features with highest Feature F-measure on that cluster. We make use of this cluster labeling method in all our experiments and systematically compute cluster labeling on the output clusterings. This has two advantages for verb clustering. On the one hand, it facilitates clustering interpretation in that cluster labeling clearly indicates the association between clusters (verbs) and their prevalent features. On the other hand, it supports the creation of a Verbnet style classification in that cluster labeling directly provides classes grouping together verbs, thematic grids and subcategorisation frames.

3 Data and Evaluation Results

3.1 Gold Standard

To evaluate the association between verbs, frames and grids provided by the IGNGF clustering method, we used a reference corpus called V-gold proposed in [6]. V-gold consists of 16 fine grained Levin classes with 12 verbs each (translated to French) whose predominant sense in English belong to that class. Because we aim to use the classification for semantic role labelling and therefore wish to associate each verb with a thematic grid, we use a slightly modified version of this gold standard which associates each Levin class with the corresponding Verbnet class name and thematic grid; merges some of the thematic roles; and groups together classes sharing the same thematic grids. The resulting gold standard groups 116 verbs into 12 Verbnet classes each associated with a unique thematic grid.

3.2 Evaluation Metrics

For evaluating the association between verbs, frames and grids provided by the IGNGF clustering, we use several evaluation metrics which bear on different properties of the clustering. The first group of metrics are supervised metrics based relying the V-gold corpus and the second group are unsupervised metrics relying solely on the clustering results.

Modified Purity and Accuracy. Following [6], we use modified purity (mPUR); weighted class accuracy (ACC) and F-measure to evaluate the clusterings produced. These are computed as follows. Each induced cluster is assigned the gold class (its *prevalent class*, prev(C)) to which most of its member verbs belong. A verb is then said to be correct if the gold associates it with the prevalent class of the cluster it is in. Given this, purity is the ratio between the number of correct gold verbs in the clustering and the total number of gold verbs in the clustering[1]:

$$mPUR = \frac{\sum_{C \in \text{Clustering}, |\text{prev}(C)|>1} |\text{prev}(C) \cap C|}{\text{Verbs}_{\text{gold} \cap \text{Clustering}}}$$

where $\text{Verbs}_{\text{gold} \cap \text{Clustering}}$ is the total number of gold verbs in the clustering.

Accuracy represents the proportion of gold verbs in those clusters which are associated with a gold class, compared to all the gold verbs in the clustering. To compute accuracy we associate to each gold class C_{gold} a dominant cluster, ie. the cluster $\text{dom}(C_{\text{gold}})$ which has most verbs in common with the gold class. Then accuracy is given by the following formula:

$$ACC = \frac{\sum_{C \in \text{gold}} |\text{dom}(C) \cap C|}{\text{Verbs}_{\text{gold} \cap \text{Clustering}}}$$

Finally, F-measure is the harmonic mean of mPUR and ACC.

Coverage. To assess the extent to which a clustering matches the gold classification, we additionally compute the *coverage* of each clustering that is, the proportion of gold classes that are prevalent classes in the clustering.

Cumulative Micro Precision (CMP). As pointed out in [12], unsupervised evaluation metrics based on cluster labelling and feature maximisation can prove very useful for identifying the best clustering strategy. Following [1], we use CMP to identify the best clustering. Computed on the clustering results, this metrics evaluates the quality of a clustering with respect to the cluster features rather than with respect to a gold standard. It was shown in [13] to be effective both in detecting degenerated clustering results including a small number of large heterogeneous, “garbage” clusters, and a big number of small size “chunk” clusters and in figuring out the coherency of obtained clusters, whatever their size.

First, the *local Recall* (R_c^f) and the *local Precision* (P_c^f) of a feature f in a cluster c are defined as follows:

$$R_c^f = \frac{|v_c^f|}{|V^f|} \quad P_c^f = \frac{|v_c^f|}{|V_c|}$$

where v_c^f is the set of verbs having feature f in c, V_c the set of verbs in c and V^f, the set of verbs with feature f.

Cumulative Micro-Precision (CMP) is then defined as follows:

$$CMP = \frac{\sum_{i=|C_{inf}|,|C_{sup}|} \frac{1}{|C_{i+}|^2} \sum_{c \in C_{i+}, f \in F_c} P_c^f}{\sum_{i=|C_{inf}|,|C_{sup}|} \frac{1}{C_{i+}}}$$

where C_{i+} represents the subset of clusters of C for which the number of associated verbs is greater than i, and: $C_{inf} = argmin_{c_i \in C}|c_i|, C_{sup} = argmax_{c_i \in C}|c_i|$

[1] Clusters for which the prevalent class has only one element are ignored.

3.3 Experimental Setup

For our clustering experiments we use the 2183 French verbs occurring in the translations of the 12 classes in the V-gold standard (cf. Section 3.1). Since we ignore verbs with only one feature, the number of verbs and ⟨verb, feature⟩ pairs considered may vary slightly across experiments. Since our aim is to acquire a classification which covers the core verbs of French, we choose to extract the verb features used for clustering, not from a large corpus parsed automatically, but from manually validated resources.

The lexical resources used for feature extraction are (i) a syntactic lexicon for French verbs and (ii) the English Verbnet. The syntactic lexicon merges three manually validated lexicons namely, Dicovalence [14], TreeLex [15] and the LADL tables [16]. It contains 5918 verbs, 20433 lexical entries (i.e., verb/frame pairs) and 345 subcategorisation frames. More details on the exploited ressources are given in [17].

From the syntactic lexicon, we extract for each verb its subcategorisation frames (**scf**) together with syntactic features (**syn**). These additional features indicate for ex. whether a verb accepts symmetric arguments; has four or more arguments; combines with a predicative phrase; takes a sentential complement or an optional object. These features are meant to help identify specific Verbnet classes and thematic roles. We also extract four semantic features (**sem**) from the lexicon which indicate whether a verb takes a locative or an asset argument and whether it requires a concrete object (non human role) or a plural role. From Verbnet, we extract thematic grid information (**grid**) as follows. We first translate the verbs in the English Verbnet classes to French using English-French dictionaries. We then associate each French verb with a Verbnet class whenever it is a translation of an English verb in that class. Finally, we train a SVM classifier for determining a probability estimate for each ⟨French verb, English Verbnet class⟩ association.

We apply an IDF-Norm weighting scheme on the obtained features to decrease the influence of the most frequent features (IDF component) and to compensate for discrepancies in feature number (normalisation).

We use K-means as a baseline. For each clustering method (K-means and IGNGF), we let the number of clusters vary between 1 and 30 to obtain a partition that reaches an optimum F-measure and a number of clusters that is in the same order of magnitude as the initial number of V-gold classes (i.e. 12 classes). K-means method is initialized with data samples. For IGNGF method parameters, we use the standard learning ratios proposed by [10] ($\epsilon_a = 0.05$: winner, $\epsilon_b = 0.006$: neighborhood); the maximun age of connexion (age_{conn}) and maximum age of embryo neurons (age_{conn}) are set to usual values exploited by [11] ($age_{conn} = 5$, $age_{embr} = 15$); distance influence parameter related to equation(2) is experimentally set to $\sqrt{2}$.

3.4 Quantitative Results Analysis

Table 1 includes the evaluation results summary for all feature sets. In terms of F-measure, the results range from 0.61 to 0.70. These results outperform [6] whose best F-measures vary between 0.55 for verbs occurring at least 150 times in the training data and 0.65 for verbs occurring at least 4000 times in this training data. The results are not directly comparable however for two reasons. First, the gold data is slightly different due to the grouping of Verbnet classes through their thematic grids. Second, [6] use syntactic and semantic features for training that are automatically acquired from a large corpus of automatically parsed sentence. In contrast, we extract our features from existing lexical resources.

In terms of features, the best results are obtained using the **grid-scf-sem** feature set shown in Table 1 with an F-measure of 0.70. However, as soon as the gold reference verbs represent only 5% of the whole dataset (i.e. 116 reference verbs among approx. 2183 exploited verbs), this measure can be considered as not sufficiently statistically reliable. Indeed, for this data set, the unsupervised evaluation metrics (cf. Section 3.2) highlight strong cluster cohesion with a number of clusters close to the number of gold classes (13 clusters for 12 gold classes) and a high Cumulative Micro-Precision (CMP = 0.3) indicating homogeneous clusters in terms of maximising features. The coverage of 0.75 indicates that approximately 9 out of the 12 gold classes could be matched to a prevalent label. That is, 9 clusters were labelled with a prevalent label corresponding to 9 distinct gold classes.

Table 1. Results. Cumulative micro precision (CMP) is given for the clustering at the mPUR optimum and in parantheses for 13 clusters clustering.

Features set	Nb fea	Nb vrb	mPUR	ACC	F	Nb cla	Cov	CMP at opt
					(gold)			(at 13 cla.)
scf	220	2085	0.93	0.48	0.64	17	0.58	0.28 (0.27)
grid, scf	231	2085	0.94	0.54	0.68	14	0.67	0.12 (0.12)
grid, scf, sem	237	2183	0.86	0.59	**0.70**	13	**0.75**	0.30 (**0.30**)
grid, scf, synt	236	2150	0.87	0.50	0.63	14	0.75	0.13 (0.14)
grid, scf, synt, sem	242	2201	0.99	0.52	0.69	16	0.83	0.50 (0.22)
scf, sem	226	2183	0.83	0.55	0.66	23	0.67	0.40 (0.26)
scf, synt	225	2150	0.91	0.45	0.61	15	0.50	0.17 (0.22)
scf, synt, sem	231	2101	0.89	0.47	0.61	16	0.67	0.57 (0.11)

In contrast, the classification obtained using the **scf-synt-sem** feature set has a higher CMP for the clustering with higher mPUR (0.57); but a lower F-measure (0.61), a larger number of clusters (16). That is, this clustering has many clusters with strong feature cohesion but a class structure that markedly differs from the gold. Since there might be differences in structure between the English Verbnet and the thematic classification for French we are building, this is not necessarily incorrect however. Further investigation on a larger data set would be required to assess which clustering is in fact better given the data used and the classification searched for.

In general, data sets whose description includes semantic features (**sem** or **grid**) tend to produce better results than those that do not (**scf** or **synt**). This is in line with results from [6] which shows that semantic features help verb classification. It differs from it however in that the semantic features used by [6] are selectional preferences while ours are thematic grids and a restricted set of manually encoded selectional preferences.

The best results are obtained with the IGNGF method on most of the data sets. Hence, IGNGF method systematically produce models with much higher CMP values than

Table 2. Sample output for a cluster produced with the **grid-scf-sem** feature set and the IGNGF clustering method

```
C1- 7(7) [315(315)]
----------
Prevalent Label --- = Cause-Experiencer

0.273245 G-Cause-Experiencer
0.173498 C-SUJ:Ssub,OBJ:NP
0.138411 C-SUJ:NP,DEOBJ:PP
0.091732 C-SUJ:NP,DEOBJ:PP,DUMMY:REFL
...
**********
**********
0.013839 T-Asset
0.013200 C-SUJ:NP,DEOBJ:Ssub,POBJ:PP
0.009319 C-SUJ:Ssub,OBJ:NP,POBJ:PP
...
[flatter 0.907200 3(1)] [charmer 0.889490 3(0)] [ex-
ulter 0.889490 3(0)] [**frissonner 0.889490 3(0)]
[mortifier 0.889490 3(0)] [poustoufler 0.889490 3(0)]
[ptir 0.889490 3(0)] [ravir 0.889490 3(0)] [**trem-
bler 0.889490 3(0)] [**trembloter 0.889490 3(0)]
[dcourager 0.872350 2(2)]...
```

K-means (3x higher for **grid-scf-sem** feature set) figuring out the much higher cohesion of its resulting clusters.

3.5 Qualitative Analysis

We carried out a manual analysis of the clusters examining both the semantic coherence of each cluster (do the verbs in that cluster share a semantic component?) and the association between the thematic grids, the verbs and the syntactic frames provided by clustering.

Table 2 shows an illustrating cluster and its features as derived by the IGNGF algorithm on our experimental dataset. Features are displayed in decreasing order of Feature F-measure given by Equation (1) and features whose Feature F-measure is under the average Feature F-measure of the overall clustering are clearly separated from others. In the sample cluster shown in Table 2 these are listed above the two star lines. In addition, for each verb in a cluster, a confidence score is computed as follows. Let the main features of a class be the features whose Feature F-measure is above the average Feature F-measure of the features labelling that cluster. Then the confidence score of a verb v in a cluster c is the ratio between the sum of the Feature F-measures of v's features over the sum of the Feature F-measures of c's features. Verbs of a cluster whose confidence score is 0 are considered as orphans[2].

[2] The overall number of orphans can be considered as an additional clustering quality index. The higher that number is, the worse is the result.

To assess semantic homogeneity, we examined each cluster seeking to identify one or more Verbnet labels characterising the verbs contained in that cluster. From the 13 clusters produced by clustering, 11 clusters could be labelled. As it can be observed, some clusters group together several subclasses and conversely, some Verbnet classes are spread over several clusters. This is not necessarily incorrect though.

To start with, recall that we are aiming for a classification which groups together verbs with the same thematic grid on the basis of English Verbnet grids. Given this, in all cases of subclass grouping, it can be observed that semantic features necessary to provide a finer grained analysis of their differences are lacking. Conversely, in all cases of class spreading, clustering interestingly highlights classes which are semantically homogeneous but syntactically distinct. In these last cases, it figures out a syntactic distinction which is present in French but not in English.

Conclusion

We firstly achieve in this paper a short presentation of the IGNGF (Incremental Growing Neural Gas with Feature maximisation) method : a recent neural clustering method in which the use of a standard distance measure for determining a winner is replaced in IGNGF by cluster feature maximization. One main advantage of this method, as compared to concurrent methods, is that the maximized features used during learning can also be exploited in a final step for accurately labeling the resulting clusters with a "cluster profile" i.e., a set of features representative of those clusters.

We then present a novel approach to verb classification which makes use of the specific clustering and labeling capabilities of the IGNGF method. Our experiment conducted on French verbs showed that this method outperforms alternative approaches on a (slightly modified) existing benchmark. In this context, it also illustrates that the use of obtained labels and associated unsupervised measures based on cluster maximized features signficantly helps to confirm the coherency of the results.

A complementary task would be to more deeply estimate the quality of the syntactic frames and thematic grids associated by IGNGF with the verb clusters. For that purpose, we plan to compare the acquired classification with a reference corpus in which the syntactic arguments have been manually annotated with semantic roles. In the case of successfull results, our final goal would be to exploit the IGNF method to bootstrap a Verbnet style classification for French.

References

[1] Lamirel, J.C., Mall, R., Cuxac, P., Safi, G.: Variations to incremental growing neural gas algorithm based on label maximization. In: The 2011 International Joint Conference on Neural Networks (IJCNN), pp. 956–965 (2011)

[2] Kipper, K., Dang, H.T., Palmer, M.: Class-based construction of a verb lexicon. In: AAAI/IAAI, pp. 691–696 (2000)

[3] Dorr, B.J.: Large-scale dictionary construction for foreign language tutoring and interlingual machine translation. Machine Translation 12(4), 271–325 (1997)

[4] Prescher, D., Riezler, S., Rooth, M.: Using a probabilistic class-based lexicon for lexical ambiguity resolution. In: 18th International Conference on Computational Linguistics, Saarbrucken, Germany, pp. 649–655 (2000)

[5] Korhonen, A.: Semantically motivated subcategorization acquisition. In: ACL Workshop on Unsupervised Lexical Acquisition, Philadelphia (2002)

[6] Sun, L., Korhonen, A., Poibeau, T., Messiant, C.: Investigating the cross-linguistic potential of VerbNet-style classification. In: Proceedings of the 23rd International Conference on Computational Linguistics, COLING 2010, pp. 1056–1064. Association for Computational Linguistics, Stroudsburg (2010)

[7] Schulte im Walde, S.: Experiments on the automatic induction of german semantic verb classes. Computational Linguistics 32(2), 159–194 (2006)

[8] Kipper Schuler, K.: VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon. PhD thesis, University of Pennsylvania (2006)

[9] Martinetz, T., Schulten, K.: A ”Neural-Gas” Network Learns Topologies. Artificial Neural Networks I, 397–402 (1991)

[10] Fritzke, B.: A growing neural gas network learns topologies. In: Advances in Neural Information Processing Systems 7, pp. 625–632 (1995)

[11] Prudent, Y., Ennaji, A.: An incremental growing neural gas learns topologies. In: Proceedings of the 2005 IEEE International Joint Conference on Neural Networks, IJCNN 2005, vol. 2, pp. 1211–1216 (2005)

[12] Attik, M., Al Shehabi, S., Lamirel, J.C.: Clustering Quality Measures for Data Samples with Multiple Labels. In: Databases and Applications, pp. 58–65 (2006)

[13] Ghribi, M., Cuxac, P., Lamirel, J.C., Lelu, A.: Mesures de qualité de clustering de documents: prise en compte de la distribution des mots clés. In: Béchet, N. (ed.) Évaluation des Méthodes d'Extraction de Connaissances dans les Données, EvalECD 2010, Hammamet, Tunisie, Fatiha Saïs, pp. 15–28 (January 2010)

[14] van den Eynde, K., Mertens, P.: La valence : l'approche pronominale et son application au lexique verbal. Journal of French Language Studies 13, 63–104 (2003)

[15] Kupść, A., Abeillé, A.: Growing TreeLex. In: Gelbukh, A. (ed.) CICLing 2008. LNCS, vol. 4919, pp. 28–39. Springer, Heidelberg (2008)

[16] Gross, M.: Méthodes en syntaxe. Hermann, Paris (1975)

[17] Falk, I., Gardent, C., Lamirel, J.C.: Classifying French Verbs Using French and English Lexical Resources. In: ACL, pp. 207–214 (2012)

Combining Neural Clustering with Intelligent Labeling and Unsupervised Bayesian Reasoning in a Multiview Context for Efficient Diachronic Analysis

Jean-Charles Lamirel

Loria, Inria-Talaris Project,
615 r. du Jardin Botanique, 54600 Villers-lès-Nancy (France)
lamirel@loria.fr

Abstract. To cope with the current defects of existing incremental clustering methods, an alternative approach for accurately analyzing textual information evolving over time consists in performing diachronic analysis. This type of analysis is based on the application of a clustering method on data associated with two, or more, successive periods of time, and on the study of the evolution of the clusters contents and of their mappings between the different periods. This paper propose a new unsupervised approach for dealing with time evolving information with is based on the combination of neural clustering and unsupervised Bayesian reasoning. The experimental context is related to the study of the evolution of research fields in scientific literature.

Keywords: diachronic analysis, clustering, neural gas, bayesian reasoning.

1 Introduction

The literature taking into account the chronological aspect in information flows is mainly focused on "DataStream" whose main idea is the "on the fly" management of incoming (i.e. not stored) data. In this context, the data that have been considered up to now are primarily physical measurements or Web usage data. Applications on textual data (bibliographical databases, online news, …) are still stammering. Research on "DataStream" has been initiated, amongst other things, in 1996 by the DARPA through the TDT project [1]. But the algorithms resulting from this work are intended to treat very large volumes of data (i.e. DataStream) and are thus not optimal for accurately detecting topics changes in specialized domains, as for example precisely following-up the evolution of' research fields in scientific literature.

To cope with the current defects of existing incremental clustering methods, an alternative approach for sharply analyzing textual information evolving over time consists in performing diachronic analysis. This type of analysis is based on the application of a clustering method on data associated with two, or more, successive periods of time, and on the study of the evolution of the clusters contents and of their mappings between the different periods. For analyzing the evolution of the vocabulary describing the clusters of different periods, Schiebel and al. [15] propose to construct

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 245–254.
DOI: 10.1007/978-3-642-35230-0_25

a matrix of keywords comparison which is based on the percentage of keywords of one period which pre-exist in the clusters of another period. Thanks to this matrix, it is then possible for a domain expert to highlight different cluster behaviors: stability, but also merging or splitting. Even if it avoids the use of incremental clustering methods, an important limitation of this approach is that the process of comparison between clustering models must be achieved in a supervised way.

An alternative unsupervised solution has been proposed by [16]. It makes use of core documents to bridge clustering results issued from different time periods. The core documents are defined as the documents that combine high bibliographic coupling and high index terms similarities with other documents. In such a way, clusters of two time periods are considered as similar if they share a sufficient amount of references to the same core documents. Clusters are themselves built up using a co-clustering methodology mixing reference and contents information. This approach presents the advantage to be relatively independent of vocabulary changes between periods, but it necessitates exploiting referencing data.

Lamirel and al. [6] firstly introduced the dynamic and unsupervised cooperation between clustering models in the context of information retrieval. This new approach represents a major improvement of the basic clustering approach. From a practical point of view, the *MultiView Data Analysis paradigm* (MVDA), introduces the use of viewpoints associated with unsupervised Bayesian reasoning in the clustering process (fig. 1). Its main advantage is to be a generic paradigm that can be applied to any clustering method and that allows to enhance the quality and the granularity of data analysis while limiting the noise that is inherent to a global approach.

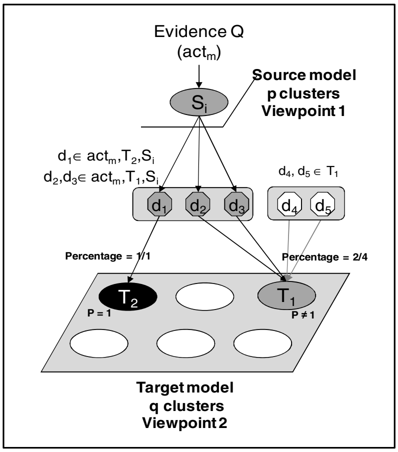


Fig. 1. The MVDA inter-models communication principle

The MVDA paradigm represents a challenging paradigm in the context of the analysis of time varying information. Hence, it allows defining efficient and precise strategies for unsupervised diachronic analyses based on the mapping into separate viewpoints of the clustering models related to the different time periods. In section 2, we highlight how to exploit the principles of MVDA to automatically perform such analyses. Section 3 describes our first experiment and its results. Section 4 draws our conclusion and perspectives.

2 A New Approach for Analyzing Time-Varying Information

Analyzing the difference between time periods concerns different kinds of topics changes or similarities that could occur between the periods (appearing topics, disappearing topics, splitting topics, merging topics, stable topics). For achieving comparison between two time periods, a *label-based diachronic approach* relying both on data properties (i.e. features) and on the MVDA paradigm can be thus defined. Thanks to this approach, a further step of cluster labeling is achieved after the construction of the clustering model for each time period. The purpose of the labeling step is to figure out which peculiar properties or endogenous labels can be associated to each cluster of a given time period. The identification of the topics relationships between two time periods is then achieved through the use of Bayesian reasoning relying on the extracted labels that are shared by the compared periods (fig. 2).

The use of reliable cluster evaluation and labeling strategies becomes thus a central point in this methodology. The labeling strategy we propose hereafter is a general-purpose strategy that has been already experienced for visualizing or synthesizing clustering results [8], for optimizing the learning process of a clustering method [3] and for highlighting the content of the individual clusters. It is based on a probabilistic approach relying on unsupervised recall and precision measures performed on cluster associated data.

For a feature *f* of a cluster *c*, *Feature Recall* (*FR_c)* and *Feature Precision* (*FP_c)* are expressed as:

$$\mathrm{FR_c(f)} = \frac{\sum_{\mathrm{d \in c}} \mathrm{W_c^f}}{\sum_{\mathrm{c' \in C}} \sum_{\mathrm{d \in c'}} \mathrm{W_d^f}} \ , \ \mathrm{FP_c(f)} = \frac{\sum_{\mathrm{d \in c}} \mathrm{W_c^f}}{\sum_{\mathrm{f' \in d, d \in c}} \mathrm{W_c^{f'}}} \tag{1}$$

where W_x^f represents the weight of the feature *f* for element x *(Feature Recall* is equivalent to the conditional probability *P(c|f)* and *Feature Precision* is equivalent to the conditional probability *P(f|c)).*

Consequently, the set of labeling features, or labels, L_c that can be considered as prevalent for a cluster *c* can be expressed as the set of endogenous cluster data features (i.e. unsupervised labels), or even exogenous cluster data features (i.e. external labels or supervised validation labels), which verifies:

$$\mathrm{L_c} = \ \{\mathrm{f \in d, d \in c \mid FF_c = Max(FF_{c'})}\} \tag{2}$$

where the *Feature F-measure (FF_c)* of a feature *f* of a cluster *c* can be defined as the harmonic means between *Feature Recall* (*FR_c)* and *Feature Precision* (*FP_c).*

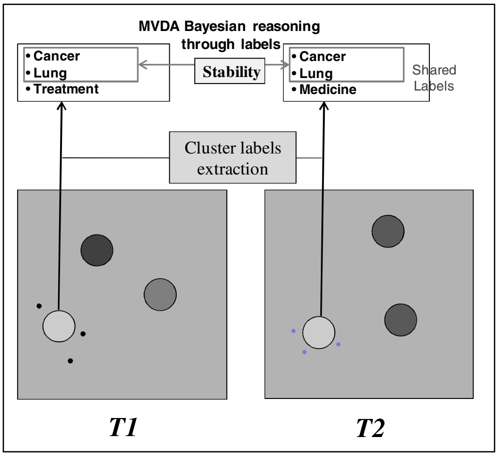


Fig. 2. The label-based approach

3 Experimentation and Results

In the context of the PROMTECH IST project, Schiebel et al. [15] have chosen to rely on the INIST PASCAL database. For their diachronic experiments, they selected the set of themes of the optoelectronic devices because this field is one of the most promising of the last decade. 3890 records related to these topics were thus extracted from the PASCAL database. Similarly to the former authors, our approach consisted in cutting out the resulting PROMTECH corpus in two periods, (1996-1999: period 1) and (2000-2003: period 2), to carry out for each one an automatic classification by using the content provided by the bibliographic records. For each year period, a specific dataset is generated. For that purpose, a set of pre-processing steps is applied to the keywords field of the corresponding records in order to obtain a weighted vector representation of the information it contains. Keywords of overall frequency less than 3 are firstly removed from the record descriptions. 1797 records indexed by 1256 keywords are consequently kept in period 1, and 2074 records indexed by 1352 keywords in period 2. In a further step, the resulting vectors associated to each record are weighted using an IDF weighting scheme [14] in both periods in order to decrease the effect of more frequent indexes.

The clustering of the datasets associated to the two periods is achieved by the use of different clustering methods. For our experiment, we select K-means as the reference method in the category of non-neural methods, as well as various neural methods, ranging from static ones, like SOM [5], NG [11] or GNG [4], to incremental ones, like IGNG [12] or IGNG-F [9]. For each method, we do many different experiments letting varying the number of clusters in the case of static methods and the vigilance parameters in the case the incremental ones. The best (i.e. optimal) clustering model of each period regarding the optimal compromise between F-average values of unsupervised *Macro-Recall* and *Macro-Precision* indexes, F-average values of unsupervised *Micro-Recall* and *Micro-Precision* indexes and F-average values of unsupervised *Cumulated Micro-* indexes is finally kept. Details on this specific cluster quality evaluation metrics, which has been proven the most efficient for textual data, are presented in [7]. The obtained values highlight that GNG neural method provides the best results on our experimental dataset for both periods. Table 1 specifically presents the quality results obtained in the first period with all the methods. It highlights that GNG reaches high quality values with the lowest difference between the *Macro-* and *Micro-* values (most homogeneous results) and the highest *Cumulated Micro-Precision* (CMP) value, indicating the best big-sized clusters. Table I also highlights the inadequacy of MSE for evaluating quality in our context.

Table 1. Summary of clustering results (time period 1)

CLUSTERING METHOD	NBR CLUSTERS	MACRO-F	MICRO-F	CMP	MSE
SOM	38	0,37	0,35	0,30	0,80
K-means	39	0,41	0,37	0,36	0,47
NG	40	0,43	0,39	0,38	0,70
GNG	40	0,44	0,41	0,48	0,62
IGNG	42	0,47	0,41	0,24	0,93
IGNG-F	39	0,49	0,42	0,32	0,98

In the end, the labels of the clusters of the best models are identified in an unsupervised way by the method of cluster feature maximization described by (Eq. 2).

To compute the probability of matching between clusters belonging to two time periods, we slightly modify the standard computation of the Bayesian inference provided by the original MVDA model [2]. The new computation is expressed as:

$$\mathrm{P(t|s)} = \frac{\sum_{\mathrm{f \in L_s \cap L_t}} \mathrm{FF_t(f)}}{\sum_{\mathrm{f \in L_t}} \mathrm{FF_t(f)}} \tag{3}$$

where *s* represents a cluster of the source period, *t* a cluster of the target period, L_x is the set of labels associated to the cluster *x*, *u*sing the cluster feature maximization approach defined by (Eq. 2), and $L_x \cap L_y$ represents the common labels, which can be called the **label matching kernel** between the cluster *x* and the cluster *y*.

The average matching probability $\mathrm{P_A(S)}$ of a source period cluster can be defined as the average probability of activity generated on all the clusters of the target period clusters by its associated labels:

$$\mathrm{P_A(S)} = \frac{1}{\mathrm{|Env(s)|}} \sum_{\mathrm{t \in Env(s)}} \mathrm{P(t|s)} \tag{4}$$

where *Env(s)* represents the set of target period clusters activated by the labels of the source period cluster *s*.

The global average activity A_s generated by a source period model *S* on a target period model *T* can be defined as:

$$A_S = \frac{1}{|S|} \sum_{s \in S} P_A(s) \tag{5}$$

Its standard deviation can be defined as σ_S.

The similarity between a cluster s of the source period and a cluster t of the target period is established if the 2 following similarity rules are verified:

$$P(t|s) > P_A(s) \quad \text{and} \quad P(t|s) > A_s + \sigma_s \tag{6}$$

$$P(s|t) > P_A(t) \quad \text{and} \quad P(s|t) > A_t + \sigma_t \tag{7}$$

Cluster splitting is verified if there is more than one cluster of the target period which verifies the similarity rules (6) and (7) with a cluster of the source period. Conversely, **cluster merging** is verified if there is more than one cluster of the source period which verifies the similarity rules (6) and (7) with a cluster of the target period.

Clusters of the source period that do not have similar cluster on the target period are considered as **vanishing clusters**. Conversely, clusters of the target period that do not have similar cluster on the source period are considered as **appearing clusters**.

Table 2. Summary of the time comparison results

TIME PERIOD	NBR GROUPS	NBR MATCH	NBR DISAPPEAR	NBR APPEAR	NBR SPLIT	NBR MERGE
1996-1999	43	33	10	-	7	-
2000-2003	50	38	-	12	-	3

Table 2 summarizes the results of our experiment of time periods comparison, in terms of identification of correspondences and differences. For a given period, the number of clusters implied in the comparison corresponds to its optimal number of clusters. It should be noted that the number of cluster splitting of the first period into the second period is more important than the converse number of merging into this latter period, which indicates a diversification of the research in the field of optoelectronics during the second period.

Finally, clusters similarity and divergence reports are automatically build up for presentation to the analysts. Each report includes one cluster of each period, whenever it is a similarity report, or one cluster of a single period, whenever it is a divergence report (i.e. an appearing or disappearing topic). In the case of a similarity report, the similarities between the clusters of the compared periods are identified by shared groups of labels (i.e. **matching kernels**), extracted from the clusters maximized features (Eq. 2), which we have also named **core-labels**. These **core-labels** illustrate in a specific way the nature of the temporal correspondences. The labels of the clusters of each period which does not belong to the matching kernel of a similarity report are also considered separately. They are used to figure out small temporal changes occurring in the context of an overall topic similarity between two periods. Said labels are displayed in decreasing order of their *Feature F-measure* difference with the alternative periods. If a specific label of a given period does not exist in the alternative period, or if its *Feature F-measure* is under the *Average Feature F-measure* $(\overline{FF})$ of the overall clustering, it is marked as absent of the latter period.

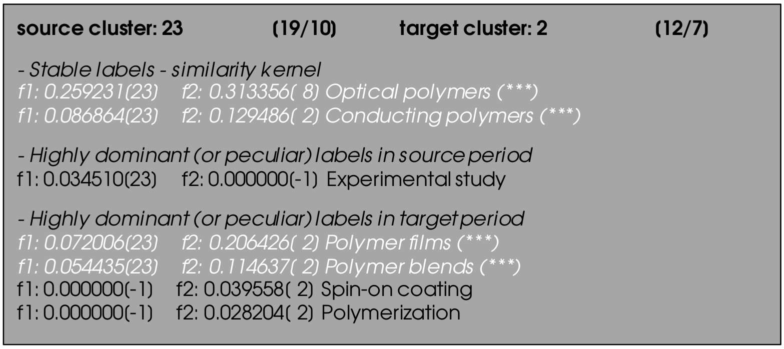
In a final step, reports are slightly adapted using an automatic process in order to highlight the most important information that they provide. For similarity reports, an automatic core label migration process is used to better figure out to which period each **core label** is mostly related. The migration of one **core label** to a given period is applied if the *Feature F-measure* of this label is twice more important in this period than in the other one. Moreover, the important differences of *Feature F-measure* between periods are highlighted by color gradation in the reports (see fig. 3).

For the sake of validation, all the adapted similarity and divergence reports have been made available to a pool of French INIST librarians specialized in the optoelectronics domain. Looking to these reports, the librarians clearly point out that the latter, whilst maintaining both a sufficiently general description level and an accurate contextual background, make it possible to very precisely reveal the tremendously rich developments of the research topics in the optoelectronic domain during the 1996-2003 period, altogether, from the theoretical studies to the practical applications (from optical polymers to polymer films (fig. 3), from surface emitting lasers or semi-conductor lasers to vertical cavity lasers or VCSEL, …), from the exploitation of new chemical components to the production of new devices (from gallium arsenide to quantum well devices, …), or new semi-conductors types (from **silicon compounds** to **amorphous semi-conductors**, from **gallium compound** to **wide band gap semi-conductors**, raise of exploitation of **germanium**, …), or the slight emerging of **new semiconductors structures** or **organization** which might become **autonomous** or **self-assembling structures** .

Another interesting point concerning the behavior of the proposed method is that the vocabulary changes which are related to slight or contextual thematic evolutions might well be merged in the same similarity report, without thus associating those changes to different contexts, or even missing to detect them. As an example, one of the resulting report helps to confirm the progressive evolution of the optoelectronics domain from punctual developments to high scale industrial processes (evolution of the concept of **optical fabrication** to the one of **optical design**).

Thanks to the experts, automatic reports of divergence between periods, materializing disappearances or emergences of subjects (topics), play the role of highlighting more important changes in the domain than the ones that could be highlighted by the similarity reports. The complete disappearance of research on **optical fibers** during the second period is thus clearly highlighted. Conversely, the full appearance of new research works on **phosphorescence**, jointly with the very significant development of those on **fluorescence**, is also correctly highlighted in such a way. Last but not least, the emergence of research works on **high-resolution optical sensors** and on their **integration** on **chips**, directly related to the important development of digital camera market in the second period (fig. 4), as well as the emergence of promising research on new generation of high efficiency optical nano-transistors (**quantum dots**) are also accurately figured out by the divergence reports.

An objective validation of the results of the proposed approach can also be achieved by looking up to the evolution of the count of the papers related to the main emerging or disappearing topics highlighted by the approach between the two periods. For that purpose we use the top-ranked keywords (i.e. the maximized ranked features or labels) associated with said topics and search for the related papers in the exploited dataset. Table 3 synthesizes the resulting count of such papers in each period. Both techniques clearly demonstrate the efficiency of the method to detect main changes. They also highlight the efficiency of the related Feature F-measure to quantify the amount of change between the periods.



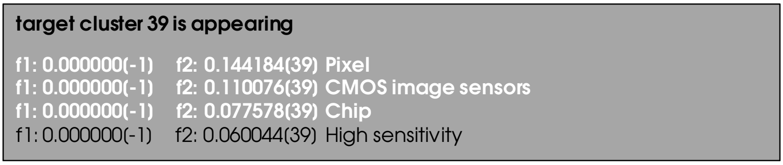
```
source cluster: 23          (19/10)       target cluster: 2            (12/7)

- Stable labels - similarity kernel
f1: 0.259231(23)   f2: 0.313356( 8) Optical polymers (***)
f1: 0.086864(23)   f2: 0.129486( 2) Conducting polymers (***)

- Highly dominant (or peculiar) labels in source period
f1: 0.034510(23)   f2: 0.000000(-1) Experimental study

- Highly dominant (or peculiar) labels in target period
f1: 0.072006(23)   f2: 0.206426( 2) Polymer films (***)
f1: 0.054435(23)   f2: 0.114637( 2) Polymer blends (***)
f1: 0.000000(-1)   f2: 0.039558( 2) Spin-on coating
f1: 0.000000(-1)   f2: 0.028204( 2) Polymerization
```

Fig. 3. Similarity report related to the strong development of polymer blends and films



```
target cluster 39 is appearing

f1: 0.000000(-1)   f2: 0.144184(39) Pixel
f1: 0.000000(-1)   f2: 0.110076(39) CMOS image sensors
f1: 0.000000(-1)   f2: 0.077578(39) Chip
f1: 0.000000(-1)   f2: 0.060044(39) High sensitivity
```

Fig. 4. Divergence report related to the strong emergence of the development and integration of high sensitivity image sensors

The complete results provided by the method cannot be presented here. They have thus been made available at a specific address [13]; the results are also presented with more details in [10]. However, one might already remark that such a topic change mining process using single keywords information was until now impossible to reach with the existing methods, which, in addition, remained at most semi-supervised. It thus makes this new approach particularly promising.

The results produced by our automated approach of comparison of time periods were finally compared with those of the analysis carried out by experts of the domain on the partitions produced over separated periods of time in the former experiment of Schiebel et al. [15]. Said analysis has mainly highlighted the following facts:

1. General set of topics of the studied corpus corresponded to the optoelectronic devices containing **mineral** or **organic semi-conductors**,
2. The research and applications of optoelectronics evolved from the field of the "**photo-detectors**" (probes, measuring instruments, …), in period 1, to the field of the "**electroluminescent diodes**", in period 2.

The above-mentioned conclusions present the disadvantage to provide only surface information on the potential topics evolutions. As it is formerly shown, the examination of the reports of similarities as well as those of divergences provided by our new diachronic method of analysis shows that it is possible to obtain both synthetic and precise conclusions, together with clear indications of tendencies (growth or decrease) in a unsupervised way, while preserving the possibility of observing general orientations, such as those expressed by the PROMTECH project experts.

Table 3. Evolution of the paper count related to the emerging and disappearing topics between the two time periods

CLUSTER REF.	TOPIC MAIN KEYWORDS	FEATURE F-MEASURE DIFF.. BTW PERIODS	PAPER COUNT IN PERIOD 1 (1996-1999)	PAPER COUNT IN PERIOD 2 (2000-2003)
16	Optical fiber	0.14	28	13
9	Fluorescence	0.12	18	36
39	CMOS image sensors	0.11	0	18
39	Pixel	0.14	0	26
48	Semicon. quantum dots	0.23	16	74

4 Conclusion

We illustrate in this paper the feasibility of an unsupervised incremental approach based on a time-step analysis of bibliographical data. This analysis has been carried out thanks to the exploitation of a specific model of data analysis managing multiple views on the data, namely the MVDA model. It was also based on the exploitation of a neural clustering method in combination with original and stable measures for evaluating the quality and the coherence of the clustering results, and even for precisely and automatically synthesizing (i.e. labeling) clusters content. To our knowledge, our approach represents the first approach that has being proposed for fully automatizing the process of analysis of time evolving textual information using solely the textual content. Our experimentation proved that this approach is reliable and that it can produce precise and significant results on a complex dataset constituted of bibliographic records, like a European reference dataset related to the research domain of optoelectronic devices.

To help to figure out the robustness of our method to high vocabulary change, we finally plan to precisely compare it with the recent diachronic approaches based on co-clustering of lexical and bibliographical information [16].

References

1. Allan, J., Carbonell, J., Doddington, G., Yamron, J., Yang, Y.: Topic detection and tracking pilot study, final report. In: Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, Lansdowne, Virginia (1998)

2. Al Shehabi, S., Lamirel, J.-C.: Inference Bayesian Network for Multi-topographic neural network communication: a case study in documentary data. In: Proceedings of ICTTA, Damas, Syria (April 2004)
3. Attik, M., Lamirel, J.-C., Al Shehabi, S.: Clustering Analysis for Data with Multiple Labels. In: Proceedings of the IASTED International Conference on Databases and Applications (DBA), Innsbruck, Austria (February 2006)
4. Frizke, B.: A growing neural gas network learns topologies. In: Tesauro, G., Touretzky, D.S., leen, T.K. (eds.) Advances in Neural Information Processing Systems, vol. 7, pp. 625–632. MIT Press, Cambridge MA (1995)
5. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics 43, 56–59 (1982)
6. Lamirel, J.-C., Créhange, M.: Application of a symbolico-connectionist approach for the design of a highly interactive documentary database interrogation system with on-line learning capabilities. In: Proceedings ACM-CIKM 1994, Gaitherburg, Maryland, USA (November 1994)
7. Lamirel, J.-C., Al-Shehabi, S., François, C., Hoffmann, M.: New classification quality estimators for analysis of documentary information: application to patent analysis and web mapping. Scientometrics 60(3), 445–462 (2004)
8. Lamirel, J.-C., Ta, A.P., Attik, M.: Novel Labeling Strategies for Hierarchical Representation of Multidimensional Data Analysis Results. In: IASTED International Conference on Artificial Intelligence and Applications (AIA), Innsbruck, Austria (February 2008)
9. Lamirel, J.C., Mall, R., Cuxac, P., Safi, G.: Variations to incremental growing neural gas algorithm based on label maximization. In: Proceedings of IJCNN 2011, San José, CA, USA (August 2011)
10. Lamirel, J.-C.: A new approach for automatizing the analysis of research topics dynamics: application to optoelectronics research. Scientometrics 93(1), 151–466 (2012)
11. Martinetz, T., Schulten, K.: A "neural gas" network learns topologies. In: Kohonen, T., Makisara, K., Simula, O., Kangas, J. (eds.) Articial Neural Networks, pp. 397–402. Elsevier Amsterdam (1991)
12. Prudent, Y., Ennaji, A.: An Incremental Growing Neural Gas learns Topology. In: 13th European Symposium on Artificial Neural Networks, ESANN 2005, Bruges, Belgium, April 27-29 (2005); published in Proceedings of 2005 IEEE International Joint Conference on Neural Networks, IJCNN 2005, vol. 2(31), pp. 1211–1216, July 31-August 4 (2005)
13. Results (2012), https://sites.google.com/site/diacresults2012
14. Robertson, S.E., Sparck Jones, K.: Relevance Weighting of Search Terms. Journal of the American Society for Information Science 27, 129–146 (1976)
15. Schiebel, E., Hörlesberger, M., Roche, I., François, C., Besagni, D.: An advanced diffusion model to identify emergent research issues: the case of optoelectronic devices. Scientometrics 83(3), 765–781 (2010)
16. Thijs, B., Glänzel, W.: A new hybrid approach for bibliometrics aided retrieval. In: Sixth International Conference on Webometrics, Informetrics & Scientometrics, and 11th COLLNET Meeting, Mysore, India (October 2010)

Lexical Recount between Factor Analysis and Kohonen Map: Mathematical Vocabulary of Arithmetic in the Vernacular Language of the Late Middle Ages

Nicolas Bourgeois[1], Marie Cottrell[1], Benjamin Déruelle[2], Stéphane Lamassé[2], and Patrick Letrémy[1]

[1] SAMM - Université Paris 1 Panthéon-Sorbonne
90, rue de Tolbiac, 75013 Paris, France
nbourgeo@phare.normalesup.org,
{marie.cottrell,patrick.letremy}@univ-paris1.fr
[2] PIREH-LAMOP - Université Paris 1 Panthéon-Sorbonne
1, rue Victor Cousin, Paris, France
{benjamin.deruelle,stephane.lamasse}@univ-paris1.fr

Abstract. In this paper we present a combination of factorial projections and of SOM algorithm applied to a text mining problem. The corpus consists of 8 medieval texts which were used to teach arithmetic techniques to merchants. Classical Factorial Component Analysis (FCA) gives nice representations of the selected words in association with the texts, but the quality of the representation is poor in the center of the graphs and it is not easy to look for the successive projections to conclude. So using the nice properties of Kohonen maps, we can highlight the words which seems to play a special role in the vocabulary since their are associated with very different words from a map to another. Finally we show that combination of both representations is a powerful help to text analysis.

Keywords: Text Analysis, Factorial Component Analysis, Kohonen Map.

1 Introduction

1.1 Context

One approach to the understanding of the evolution of science is the study of the evolution of the language used in a given field. That is why we would like to pay attention to the vernacular texts dealing with practical arithmetic and written for the instruction of merchants: such texts are known since the XIIIth century, and from that century onwards and especially after the diffusion of the Latin Leonard of Pisa's *Liber Abaci*, the vernacular language appears more and more as the medium of practical mathematics.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 255–264.
DOI: 10.1007/978-3-642-35230-0_26

Treaties on arithmetical education were therefore mostly thought and written in local languages. In this process, the XV$^{\text{th}}$ century appears as a time of exceptional importance because we can see then how the inheritance of two hundred years of practice transfers into words[1]. For the authors of these texts, the purpose was not only to teach merchants but also to develop knowledge in vernacular language, and their books were circulated far beyond the shopkeepers' world, as far as the humanists' circles for example.

1.2 An Objective of Historical Research: The Study of Specialized Languages

The work previously done (Lamassé [2012]) consisted in the elaboration of a dictionary of the lexical forms found in all the treaties in order to identify the different features of the mathematical vernacular language of the time. This done, we have worked on the contexts of some especially important words in order to understand the lexicon in all its complexity, and on the specificities of each text to study the proximities and the differences between them. In other words, we should like to determine the common language that forms the specialized language beyond the specificities of each text.

2 The Data, the Objectives, the Protocol

In order to delimit a coherent corpus among the whole European production of practical calculation education books, we have chosen to pay attention to those treaties which are sometime qualified as commercial (*marchand* in French) which have been written in French between 1415 and 1520. This last date is the date of the publication of *L'arismetique novellement compose* of Estienne de La Roche which is closely related to the works of Nicolas Chuquet and which provides in the same time an opening towards the Italian authors, such as Fra Luca Pacioli. In this way, our corpus is in conformity with the rules of the discourse analysis: homogeneity, contrastiveness and diachronism[2]. It contains eight treaties on the same topic, written in the same language and by different XV$^{\text{th}}$ century authors. The following table 1 describes some elements of the lexicometric characteristics of the corpus and shows its main quantitative imbalance.

2.1 Humanities and Social Sciences Traditional Protocol

Traditionally on this kind of textual data, HSS researchers use to work on the statistical specificities and on the contextual concordances, since they allow an easy discovery of the major lexical splits within the texts of the corpus while remaining close to the meanings of the different forms. Then, the factorial and

[1] These treaties were written not only in French but also in Italian, Spanish, English and in German.

[2] For further explanations about texts, methodology and purpose of the analysis see Lamassé [2012].

Table 1. Corpus of texts and main lexicometric features (Hapax are words appearing once in a text)

Manuscripts and **Title**	**Date**	**Author**	**Number of occurrences**	**Words**	**Hapax**
Bibl. nat. Fr. 1339	ca. 1460	anonyme	32077	2335	1229
Bibl. nat. Fr. 2050	ca. 1460	anonyme	39204	1391	544
Cesena Bibl. Malest. S - XXVI - 6, *Traicté de la praticque*	1471?	Mathieu Préhoude?	70023	1540	635
Bibl. nat. Fr. 1346, Commercial appendix of *Triparty en la science des nombres*	1484	Nicolas Chuquet	60814	2256	948
Méd. Nantes 456	ca. 1480-90	anonyme	50649	2252	998
Bibl. nat. Arsenal 2904,*Kadran aux marchans*	1485	Jean Certain	33238	1680	714
Bib. St. Genv. 3143	1471	Jean Adam	16986	1686	895
Bibl. nat. Fr . Nv. Acq. 10259	ca. 1500	anonyme	25407	1597	730

clustering methods, combined with co-occurrences analysis - see Martinez and Salem [2003] help us to cluster the texts without breaking the links with semantic analysis. However, such a method of data processing requires a preliminary treatment of the corpus, the lemmatization. It consists in gathering the different inflected forms of a given word as a single item. It offers the possibility to work at many different levels of meaning, depending upon the granularity adopted: forms, lemma, syntax. We can justify this methodological choice here by its effect on the dispersion of the various forms which can be linked to the same lemma, a high degree of dispersion making the comparison between texts more difficult. It must also be remembered that in the case of medieval texts, this dispersion is increased by the lack of orthographic norms. In our case, this process has an important quantitative consequence on the number of forms in the corpus, which declines from 13516 forms to 9463, a reduction of some 30%.

The factorial analysis allowed us to establish a typology of the complete parts of the corpus, based upon all the forms. However, it can be useful to improve this global analysis with a probabilistic calculation for each component of the corpus, by using the table of the under-frequencies (Lebart and Salem [1994]). It makes it possible to compare the parts of the corpus with each other, taking into account the occurrences of the words and their statistical specificities.

This process has been made with a particular attention to meaning of the word in order to suppress ambiguities : a good example is the French word *pouvoir* which can be a verb translated by "can" or "may", and which is also a substantive meaning "power".

Finally, to realize a clustering of the manuscripts, we have only kept the 219 words with the highest frequencies. The set of words thus selected for text classification relate both to mathematical aspects, such as operations, numbers and theirs manipulations, as well as to didactic aspects. Their higher frequencies reflect the fact that they are the language of the mathematics as they appear to be practiced in these particular texts. Thus, in what follows the data are displayed in a contingency table T with $N = 219$ rows (the words) and $p = 8$ columns (the texts) and the entry $t_{i,j}$ is the number of occurrences of word i in text j.

2.2 Factorial Correspondence Analysis (FCA)

Factorial Correspondence Analysis is a factorial method which provides the simultaneous representation of both the individuals and their characteristics, that is to say the columns and the rows of a table, in our case the texts (columns) and the words (rows). Figure 1 and 2 show the projection of the data on the first two factorial axes.

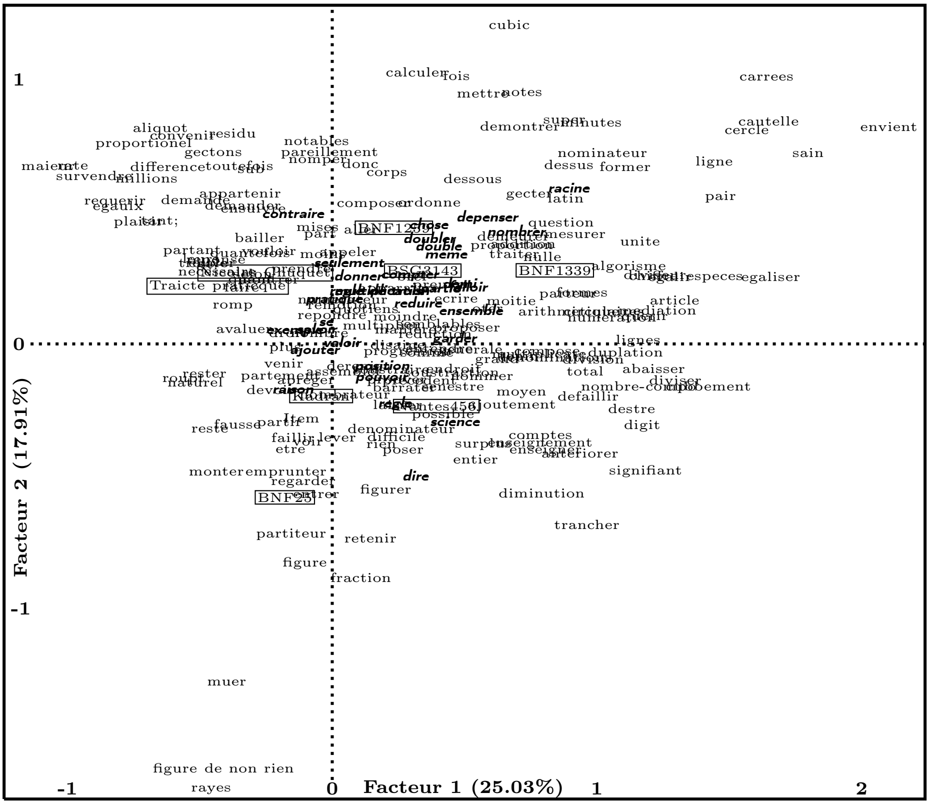


Fig. 1. Projection on first two factors of the FCA. The eight texts appear in frames, the slanted words stand for fickle words (this notion will be defined in section 4).

The first two factors (43.94% of the total variance) show the diversity of the cultural heritages which have informed the language of these treaties. The first factor (25.03%) discriminates between the vocabulary according to its relation to the university legacy on the left, and to the tradition of mathematical problems on the right.

On the left, we can observe a group whose strong homogeneity comes from its orientation towards mathematical problems (*trouver* that is to say "to find", *demander* which we can translate as "to ask") and their iteration (*item, idem*).

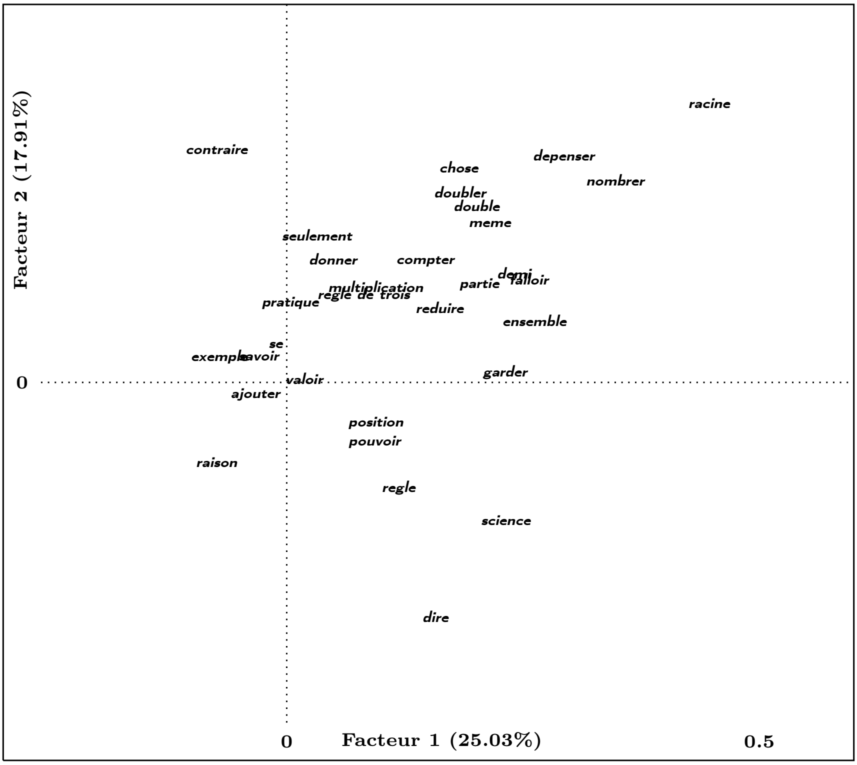


Fig. 2. Projection on first two factors of the FCA (zoom on the central part). Only the fickle words are represented.

That vocabulary can be found most often in both the appendix of *Triparty en la science des nombres* and *Le Traicte de la praticque*. Furthermore, there are more verbal forms on this side of the axis than on the other. And we can find verbs like *requerir* which means "to require", *convenir* "to agree", *faire* "to do", *vouloir* "to want". Some of them are prescriptive, as *devoir* "to have to" or *vouloir* "to want" for example, while others introduce examples, as *montrer* "to show". All these texts contain a lot of mathematical problems and in a way that texts are more practical.

On the right, the texts of BnF. fr. 1339 and Med. Nantes 456 are clearly more representative of the university culture, containing latin words sequences. They describe basic operations and numbers through words developed around the XII$^{\mathrm{th}}$ and the XII$^{\mathrm{th}}$ century in the universities, such as *digit*, *article* and *nombre composé*[3].

The second axis (17.91% of the variance) is mostly characterized by the text of BNF. fr. 2050 and also by *Kadran aux marchans*. It displays words of

[3] *Digit* is used for 0 to 9, *article* for every multiple of ten, and *nombre composé* is a mixed between *article* and *digit*.

Italo-Provencal origin, like *nombrateur* which refers to the division's numerator. Designation of the fraction and operation of division take a significant part of the information while the most contributory words (for ex. *figurer* "to draw") allow us to examine another dimension of these works: the graphical representation as a continuation of writing.

Correspondence Analysis displays the particularities of each text, but leaves untouched some more complex elements of the data. For instance, we cannot conclude from this opposition that the authors of the appendix of *Triparty en la science des nombres* and the *Traicte de la praticque* are ignorant of the university texts that inspire the other books. Correspondence analysis does not make fully possible the analysis of the attractions. Moreover, we cannot assert that the words which appear in the center of the graph represent a "common vocabulary": as a matter of fact, we ought to analyze all the successive factors in order to build the list of the words constituting the "normal" vocabulary.

3 SOM Algorithm for Contingency Table

One way to overcome the limitations of the Factorial Correspondence Analysis (FCA) consists of using a variant of the SOM algorithm which deals with the same kind of data, that is a contingency table (see Oja and Kaski [1999] for other applications of SOM to text mining). See Cottrell et al. [1998] for a definition of this variant of SOM, we called KORRESP.

The KORRESP algorithm consists in a *normalization* of the rows and of the columns in order to sum to 1, the *definition* of an extended data table by associating to each row the most probable column and to each column the most probable row, a *simultaneous classification* of the rows and of the columns onto a Kohonen map, by using the rows of the extended data table as input for the SOM algorithm.

After convergence of the training step, the modalities of the rows and of the columns are simultaneously classified. In our example, one can see proximities between words, between texts, between words and texts. It is the same goal as in Factorial Correspondence Analysis. The advantage is that it is not necessary to examine several projection planes : the whole information can be read on the Kohonen Map. The drawback is that the algorithm is a stochastic one, and that apparent contradictions between several runs can be troublesome.

In fact, we can use this drawback to improve the interpretation and the analysis of relations between the studied words. Our hypothesis is that the repetitive use of this method can help us to identify words that are strongly attracted/repulsed and fickle pairs.

In its classical presentation Kohonen [1995], Cottrell et al. [1998], the SOM algorithm is an iterative algorithm, which takes as input a dataset $\mathbf{x}_i, i \in \{1, \dots, N\}$ and computes prototypes $\mathbf{m}_u, u \in \{1, \dots, U\}$ which define the map.

We know that self-organization is reached at the end of the algorithm, which implies that close data in the input space have to belong to the same class or to neighboring classes, that is to say that they are projected on the same prototypes or on neighboring prototypes on the map. In what follows we call neighbors data that belong either to the same unit or to two adjacent units. But the reciprocal is not exact : for a given run of the algorithm, two given data can be neighbor on the map, while they are not in the input space. That drawback comes from the fact that there is no perfect fit between a two-dimensional map and the data space (except when the intrinsic dimension is exactly 2). Moreover, since the SOM algorithm is a stochastic algorithm, the resulting maps can be different from one run to another.

We address the issue of computing a reliability level for the neighboring (or no-neighboring) relations in a SOM map. More precisely, if we consider several runs of the SOM algorithm, for a given size of the map and for a given data set, we observe that most of pairs are almost always neighbor or always not neighbor. But there are also pairs whose associations look random. These pairs are called *fickle* pairs. This question was addressed by Bodt et al. [2002] in a bootstrap frame.

According to their paper, we can define: $NEIGH^l_{i,j} = 0$ if x_i and x_j are not neighbor in the l-th run of the algorithm, and $NEIGH^l_{i,j} = 1$ if x_i and x_j are neighbor in the l-th run of the algorithm, where (x_i, x_j) is a given pair of data, l is the number of the observed run of the SOM algorithm.

Then they define the stability index $\mathcal{M}_{i,j}$ as the average of $NEIGH_{i,j}$ over all the runs ($l = 1, \dots, L$), i. e. $\mathcal{M}_{i,j} = (1/L) \sum_{l=1}^{L} NEIGH^l_{i,j}$. The next step is to compare it to the value it would have if the data x_i and x_j were neighbor by chance in a completely random way.

So we can use a classical statistical test to check the significance of the stability index $\mathcal{M}_{i,j}$. Let U be the number of units on the map. If edge effects are not taken into account, the number of units involved in a neighborhood region (as defined here) is 9 in a two-dimensional map. So for a fixed pair of data x_i and x_j, the probability of being neighbor in a random way is equal to $9/U$ (it is the probability for x_j to be a neighbor of x_i by chance once the class x_i belongs to is determined).

Let $Y_{i,j} = \sum_{l=1}^{L} NEIGH^l_{i,j}$ be the number of times when the data x_i and x_j are neighbor for L different, independent runs. It is easy to see that $Y_{i,j}$ is approximately distributed as a Binomial distribution with parameters L and $9/U$. Using the classical approximation of Binomial Distribution by a Gaussian one (L is large and $9/U$ not too small), we can build the critical region of the test of null hypothesis H_0 "x_i and x_j are neighbor by chance" against hypothesis H_1 : " the fact that x_i and x_j are neighbor or not is significant".

We conclude that the critical region for a test level of 5% based on $Y_{i,j}$, is

$$\mathbb{R} - [L\frac{9}{U} - 1.96\sqrt{L\frac{9}{U}(1 - \frac{9}{U})}, L\frac{9}{U} + 1.96\sqrt{L\frac{9}{U}(1 - \frac{9}{U})}]$$

$$\text{Fix } A = \frac{9}{U} \text{ and } B = 1.96\sqrt{\frac{9}{UL}(1 - \frac{9}{U})}.$$

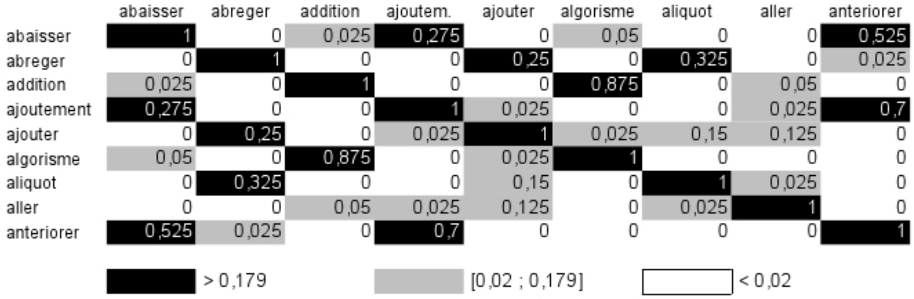
Practically, in this study, for each pair of words, we can compute (over 40 experiments) the index $\mathcal{M}_{i,j} = Y_{i,j}/L$, and conclude. Henceforth:

- if their index is greater than $A + B$, they are almost always together in a significant way, the words attract each other.
- if their index is comprised between $A - B$ and $A + B$, their proximity depends on the text they belong, they are a fickle pair.
- if their index is less than $A - B$, they are almost never neighbor, the words repulse each other.

4 Analysis of Fickle Pairs and Nodes

4.1 Identification of Fickle Pairs

We run KORRESP L times and store the result in a matrix $\mathcal{M}$ of size $(N + p) \times (N + p)$. The value stored in a given cell i, j is the proportion of maps where i and j are neighbors.



	abaisser	abreger	addition	ajoutem.	ajouter	algorisme	aliquot	aller	anteriorer
abaisser	1	0	0,025	0,275	0	0,05	0	0	0,525
abreger	0	1	0	0	0,25	0	0,325	0	0,025
addition	0,025	0	1	0	0	0,875	0	0,05	0
ajoutement	0,275	0	0	1	0,025	0	0	0,025	0,7
ajouter	0	0,25	0	0,025	1	0,025	0,15	0,125	0
algorisme	0,05	0	0,875	0	0,025	1	0	0	0
aliquot	0	0,325	0	0	0,15	0	1	0,025	0
aller	0	0	0,05	0,025	0,125	0	0,025	1	0
anteriorer	0,525	0,025	0	0,7	0	0	0	0	1

■ > 0,179 ▨ [0,02 ; 0,179] □ < 0,02

Fig. 3. Excerpt from matrix $\mathcal{M}$ with $L = 40$ and $r = 1$

Figure 3 displays an example of the nine first rows and columns of such a matrix. We have highlighted with colors three different situations. According to the theoretical study mentioned above:

- Black cells stand for pairs that are neighbors with high probability (proximity happens with frequency greater than $A + B$).
- White cells stand for pairs that are not neighbors with high probability (proximity happens with frequency less than $A - B$).
- Grey cells are not conclusive.

4.2 From Fickle Pairs to Fickle Words

We call fickle a word which belongs to a huge number of fickle pairs:

$$|\{i, |\mathcal{M}_{i,j} - A| \leq B\}| \geq T$$

Unfortunately, it is not quite an easy task to find an appropriate threshold T. Here we have decided to fix it according to data interpretation. The 30 ficklest words, whose number of safe neighbors/non-neighbors is between 89 and 119, are displayed in table 2.

Table 2. 30 ficklest words among 219 studied

contraire "opposite" (89)	*regle de trois* "rule of three" (104)	*depenser* "to expend" (112)
doubler "to double" (89)	*savoir* "to know" (105)	*racine* "root" (113)
falloir "to need" (93)	*partie* "to divide" (105)	*chose* "thing" (113)
meme "same, identical" (93)	*position* "position" (107)	*compter* "to count" (113)
pratique "practical" (94)	*exemple* "for example" (107)	*dire* "to say" (113)
seulement "only" (94)	*demi* "half" (108)	*nombrer* "count" (115)
double "double" (97)	*garder* "to keep"(109)	*raison* "calculation, problem" (116)
multiplication (99)	*science* "science" (109)	*donner* "to give" (117)
reduire "to reduce" (103)	*pouvoir* "can" (111)	*ensemble* "together" (117)
regle "rule" (103)	*se* "if" (111)	*valoir* "to be worth" (119)

FCA with Fickle Pairs. The combination of both techniques FCA and SOM whose result is displayed in figure 1 is interesting because it preserves properties from the FCA while giving additional information about the center of the projection - which is usually difficult to interpret. Indeed, the identification on the FCA of the fickle forms allows us to control the general interpretation of the factorial graph, where some words find their place because of the algorithm and not because of their attraction with other forms and with the texts.

Remember that, on the first two factorial axes (see section 2.2), we have observed an opposition between the university legacy, on the right, and a more practical pole with rule, problems and fractions, on the left. It was tempting to support this observation with words such as "practical" or "rule of three". On the other hand, the fickle forms enhancement shows that these words are shared between a lot of different texts and not only linked to the treaty of Nicolas Chuquet and the *Traicté en la praticque*. As a matter of fact, they do belong to all the texts. And we can observe that the first factor opposes two technical languages that are on either side of a set of common words – and these words are obviously not to say necessarily in the center of the FCA (see for instance *racine* "root").

To conclude, we can see that two levels of interpretation are superimposed: the fickle pairs which reveal the shared lexicon and the factorial map which inserts it in a local interaction system. And because the list is not sensitive to the FCA, we can play on this combination for each successive factorial axis. It is the articulation between these two levels which makes this representation interesting. In the end, the meaning of this new kind of factorial map is quite intuitive and offers easy tools to the argumentation.

5 Perspectives and Conclusion

First, we intend to use the proposed method for other corpus to confirm its capacity to extract specialized vocabulary. In particular we want to make a new experimentation on a corpus of medieval and renaissance prologues of epics. This corpus has been constituted in order to discern the political and ideological appropriations of the chivalric culture in the context of the XV$^{\text{th}}$ and XVI$^{\text{th}}$ centuries Renaissance. Secondly, the method has to be appropriated by linguists in order to improve it according to their own paradigms.

Another challenge will be to work on a statistical characterization of a threshold for the definition of fickle pairs. Indeed, while we managed to define (through confidence intervals) a theoretical frame for reliability of a pair, we still need to infer a similar method for each data.

Finally, we think that we have open a new perspective for clustering through Kohonen maps. Indeed, the study of robust attraction/repulsion between data as well as fickle pairs can be translated into a graph. Then, we can apply methods from graph representation and graph mining in order to get visualization containing more information than a single SOM.

References

Cottrell, M., Fort, J.-C., Pagès, G.: Theoretical aspects of the SOM algorithm. Neurocomputing 21, 119–138 (1998)

de Bodt, E., Cottrell, M., Verleysen, M.: Statistical tools to assess the reliability of self-organizing maps. Neural Networks 15(8-9), 967–978 (2002)

Kohonen, T.: Self-Organizing Maps. Springer Series in Information Science, vol. 30, Berlin (1995)

Lamassé, S.: Les traités d'arithmétique médiévale et la constitution d'une langue de spécialité. In: Ducos, J. (ed.) Sciences et Langues au Moyen Âge, Actes de l'Atelier Franco-Allemand, Paris, Janvier 27-30, pp. 66–104. Universitätesverlag, Heidelberg (2012)

Lebart, L., Salem, A.: Statistique textuelle. Dunod, Paris (1994)

Martinez, W., Salem, A.: Contribution à une méthodologie de l'analyse des cooccurrences lexicales multiples dans les corpus textuels. Thèse doctorat (2003)

Oja, E., Kaski, S.: Kohonen Maps. Elsevier (1999)

Computational Study of Stylistics: Visualizing the Writing Style with Self-Organizing Maps

Antonio Neme[1,2], Sergio Hernández[3], Teresa Dey[4],
Abril Muñoz[5], and J.R.G. Pulido[6]

[1] Complex Systems Group, Universidad Autónoma de la Ciudad de México
San Lorenzo 290, México, D.F. México
[2] Institute for Molecular Medicine, Finland
neme@nolineal.org.mx
[3] Postgraduation Program in Complex Systems,
Universidad Autónoma de la Ciudad de México
[4] Faculty of Literary Creation, Universidad Autónoma de la Ciudad de México
[5] CINVESTAV IDS, México D.F.
[6] Facultad de Telemática, Universidad de Colima, México

Abstract. The style authors follow to express their ideas has been a subject of great debate. Several perspectives have been followed to try to analyze the style. In this contribution we present a computational methodology to study the writing style in a collection of hundreds of texts. For each text several attributes, which include different time series, are extracted and a battery of tools from the signal processing and the machine learning communities are applied to identify a set of features that may define a candidate style space. We applied self-organizing maps to visualize how several authors are distributed in the high-dimensional space associated to the style, and to visually prospect the similarities between styles from different authors.

Keywords: computational stylistics, authorship attribution, visualization, self-organizing maps, mutual information.

1 Introduction

The automatic identification of the style authors follow in their texts has proven to be elusive. The study of stylistics has attracted the attention of different practitioners from diverse areas. An experimented reader may be able to easily recognize the general style of his/her favorite author, but declaring the procedure they followed to recognize the style is a much harder task [1]. The style authors follow, namely the use of certain words, the avoidance of certain others, the preferential use of some grammatical structures, or any other measurable pattern is what defines the stylistics [2].

A closely related concept is that of *authorship attribution* (AA), which refers to the task of identifying the author of a text from a group of possible candidate authors [1]. This task is strongly based on the cited concept of stylistics. Stylistics

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 265–274.
DOI: 10.1007/978-3-642-35230-0_27

may be seen as the identification of attributes that define a high-dimensional space in which authors can be distinguished from each other.

The relevance of computational stylistics pervades several areas. The first one is that of literature theory, in which experts struggle to dig into the patterns that the analyzed writers tend to follow in their texts. Also, the evolution of that style is of academic interest. A second impacted field is on forensic linguistics in which it has to be determined the authorship of a text either for historical reasons or criminal charges [1,3]. Also, as a consequence of web dissemination, with several hundreds of thousands of public documents, there are several situations in which it is relevant to identify the author of particular texts or situations in which it is necessary to confirm the existence of apocrypha documents.

The impact of stylistics also reaches psychiatry. It has been stated that some of the early symptoms in certain mental disorders can already be detected in writing [4]. For example, a detailed analysis over the novels of Iris Murdoch shows that there are qualitative and quantitative differences in her novels prior to the disease and in the early stages of it. Being aware of the general patterns of evolution in stylistics may help psychiatrists and other mental health professionals to early detect symptoms of mental disorders.

Different algorithms have been proposed to identify the author of a given text [1]. However, most of those algorithms lack of explanatory properties. For example, some kernel methods present good performance, but those models are unable to show what attributes are really relevant as it is only focused in finding a high-dimensional space in which points representing texts are linearly separable.

In this contribution, we present some results associated to a project focused on the study of computational stylistics. In this project, machine learning and data mining tools are applied to a corpus of hundreds of texts covering dozens of authors. The first objective of the project is to identify those attributes that can summarize the stylistics of authors at the time that are relevant to the authorship attribution task. Also, we are interested in the analysis of the evolution of stylistics for individual authors. As the stylistics space is high-dimensional, a visualization tool is of the greatest relevance. We have applied self-organizing maps in the data exploration phase and we have been able to identify some attributes that are relevant in the definition of the minimum list of attributes.

Several attributes have been proposed as relevant in order to discern the stylistics of an author. Also, many features have been proposed to be relevant for the AA task. In this contribution, we focus our attention on attributes about the way authors make use of words. Here, we refer to words as the vocabulary but also to punctuation signs. One of the open questions is the identification of the minimum set of attributes that can lead to the identification of authors. Several attributes have been proposed, for example, the use of certain words and the lack of use of other [1]. In general, the concept of *bag of words* is frequently mentioned and, although relevant results have emerged, there are even more questions to be answered [2,5]. Writers use language following different ways to express their ideas. This variation in language allows authorship attribution to be possible [6].

The rest of this contribution is presented as follows. In section 2 we describe the attributes that define stylistics as well as the relevant aspects of SOM. In section 3 some results are described and we present a proposal to select subspaces from the stylistics space able to distinguish between texts from different authors. Finally, in section 4 some conclusions are discussed and we pinpoint to ongoing and future work.

2 Attributes, Stylistics and Data Analysis Tools

The style authors follow in their texts is described by several attributes. In this contribution we aim to identify a high-dimensional space of attributes, also called the stylistics space, in which authors can be distinguished from each other. Each text is transformed into a set of time series and from them, several tools from the signal processing and data mining fields are applied. Each text is then mapped to a point in that high-dimensional space of attributes.

The most common approach in the field of computational linguistics and natural language processing is to deal with texts under the perspective of *bag of words*. There, the relevant quantities are the relative frequencies of each word, sentence, or any other relevant structure [7]. There are several works in which texts are analyzed and classified with self-organizing maps based on very high-dimensional vectors containing the relative frequency of appearance of words [8].

In this contribution we are not only interested in the relative frequency of words, but also in the cadence authors follow when using certain words or symbols (we will refer to words also as symbols). That is, we are interested in the time series defined as the distance between consecutive instances of several relevant symbols. By that distance, we refer to the number of words that separates consecutive appearances of a given word or symbol. We are interested in obtaining those time series for the following symbols:

- the comma
- sentence length (number of words in each sentence)
- number of sentences per paragraph
- the most common word excluding the comma and the word *the/el*
- the most common word excluding articles and prepositions
- the word *the/el*

Besides the time series for certain symbols, we defined another time series, that we call simply T. It is defined as follows. Each text is transformed into a sequence of integers: each word or symbol is associated to an integer in order of appearance. The first word to appear in the text will be assigned a 0, the second non-repeated word will be associated to a 1, and so on. For example, the sentence S_1 = *My baptismal name is Egaeus; that of my family I will not mention.* is transformed to the sequence $T = \{0, 1, 2, 3, 4, 5, 6, 7, 0, 8, 9...\}$. The word *My* is assigned to code 0 as it is the first word. The second appearance of *my* is also assigned code 0. In this contribution, there is no difference between upper and lower cases. This time series is positive definite, and presents some properties that prevent the use of time series analysis tools over it, for example it is not stationary. However, from time series T a new time series B can be constructed: It is a sequence of 0 and 1, where 1 indicates the appearance of a previously unseen word and a 0 reflects the appearance of a repeated word within the analyzed text.

Other attributes are also considered, for example, the entropy of the text, the ratio between vocabulary and text length, maximum sentence length, probability distribution of the most common words, among others. The complete list of the attributes is shown in table 1. From this list a high-dimensional stylistics space S is constructed, and each text is then mapped to that space.

Table 1. Attributes defininig the stylistics space S

Attribute	Description	No. var
V	Vocabulary size	1
T	Text length in words	1
V/T	Ratio V/T	1
H	Entropy	1
MPL	Maximum paragraph length (sentences per paragraph)	1
APL	Average paragraph length	1
mPL	Minimum paragraph length	1
PDPL	Probability distribution of paragraph length (up to 30 sent. per paragr.)	30
MSL	Maximum sentence length (words per sentence)	1
ASL	Average sentence length	1
mSL	Minimum sentence length	1
PDSL	Probability distribution of sentences length (up t 200 words per sentence)	200
pMFSL	Probability of the most frequent sentence length	1
PkMCW	Probability distribution of the 30 most common words	30
pMCW	Probability of the Most Common Word (except , and the)	1
adMCW	Avg distance between consecutive appearances of MCW	1
mdMCW	Minimum distance between consecutive appearances of MCW	1
MdMCW	Maximum distance between consecutive appearances of MCW	1
pThe	Probability of the word *the/el*	1
adThe	Avg distance between consecutive appearances of *the/el*	1
mdThe	Minimum distance between consecutive appearances of *the/el*	1
MdThe	Maximum distance between consecutive appearances of *the/el*	1
pMCWx	Probability of the MCW (except articles, prepositions and ,)	1
adMCWx	Avg. dist between appearances of MCW (except articles, prepositions and ”,”)	1
mdMCWx	Min. dist. between appearances of MCW (except articles, prepositions and ”,”)	1
MdMCWx	Max. dist. between appearances of MCW (except articles, prepositions and ”,”)	1
PkMCWx	Probability distribution of the 30 MCWs (except articles, prepositions and ”,”)	30
pComma	Probability of the comma	1
adComma	Average distance between consecutive appearances of the comma	1
mdComma	Minimum distance between consecutive appearances of the comma	1
MdComma	Maximum distance between consecutive appearances of the comma	1
MIFS	MIF for time series S (40 displacements)	40
MIFPL	MIF for time series paragraph length (40 displacements)	40
MIFSL	MIF for time series sentence length(40 displacements)	40
MIFMCW	MIF for time series distance between MCW(40 displacements)	40
MIFMCWx	MIF for time series distance between MCWx(40 displacements)	40
MIFComma	MIF for time series distance between comma(40 displacements)	40
MIFThe	MIF for time series distance between *the/el*(40 displacements)	40
MIFBin	MIF for time series B (40 displacements)(40 displacements)	40
PWSS	Power spectrum of time series S (5 highest frequencies)	40
PWSPL	Power spectrum of time series paragraph length (5 highest frequencies)	5
PWSSL	Power spectrum of time series sentence length (5 highest frequencies)	5
PWSMCW	Power spectrum of time series distance between MCW (5 highest frequencies)	5
PWSMCWx	Power spectrum of time series distance between MCWx (5 highest frequencies)	5
PWSCWy	Power spectrum of time series distance between comma (5 highest frequencies)	5
PWSThe	Power spectrum of time series distance between *the* (5 highest frequencies)	5
PWSB	Power spectrum of time series B (5 highest frequencies)	5

Time series extracted from texts are the basis of the concept of stylistics we follow. However, time series *per se* only give limited details, and more processing on them is necessary. Texts may present different lengths so a normalizing methodology is needed to compare time series that may come from texts of different size. Time series are not analyzed directly. Several tools from the time series and signal processing communities can be applied in order to extract subtle and relevant patterns [9]. Among the attributes that can be extracted from time series the most common ones are the power spectrum, the Lyapunov, and the mutual information function [10]. In this contribution, we extracted the mutual information function (MIF) and power spectrum (PWS) from the time series coming from texts.

MIF is a measure of non-linear correlation between random variables or systems [11]. It gives an answer to the following question: Once we know the state a system is in, how much information does knowledge give about the state a second system is in? MIF is based in Shannon's information theory [12]. When MIF is applied to a time series, the second system (or random variable) is constructed by shifting the time series up t k positions. The length of that shift is represented the x axis when plotted.

In summary, the high-dimensional stylistics space S is a high-dimensional space formed by several MIF (40 displacements each), several power spectrum (the five highest frequencies), and several statistics. All these attributes are described in table 1. As the length of texts is distributed along one range of magnitude, finite size correction was applied for normalizing data. Once the S space is defined, we can visualize the distribution texts follow in that space by applying a non-linear mapping to a low-dimensional space. The self-organizing map (SOM) is an accurate and powerful tool to accomplish that mapping. Also, we are interested in identifying a small subset of attributes in S able to distinguish between texts from different authors and thus propose that subset as a *very small candidate stylistics space.*

SOM is frequently applied as a visualization tool. SOM is able to preserve in a low-dimensional space the approximate distribution shown by vectors in the high-dimensional input space [13]. It outperforms common visualization tools such as principal component analysis as SOM takes into account high-order statistics, instead of at most second-order statistics[14].

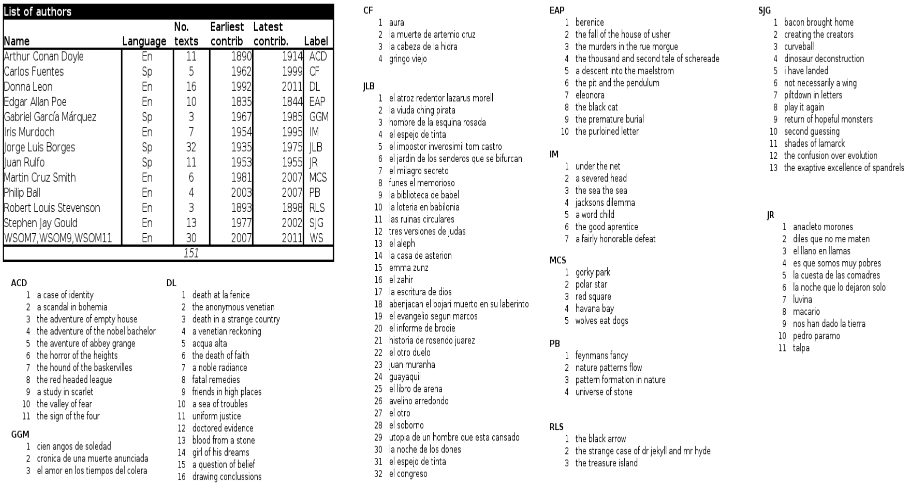
We are interested in studying stylistics from a pure statistical and signal processing perspective. That is, we think of texts as signals and we systematically study how far we can reach by leading aside grammatical and lexical issues. We are not interested in the already well established concepts of bag of words and other related aspects. In the next section, we present some maps for several texts from a dozen of writers.

3 Results

Each text is transformed to a point in the high-dimensional stylistics S space. The coordinates of each text are given by the attributes described in the previous section. We now want to know what attributes of this space are relevant to identify the author of a text.

The analyzed authors and their texts are shown in fig. 1. Texts were analyzed in accordance to the stylistics attributes described in the previous section. Only these authors are show in this contribution in order to simplify the visualization and the analysis (the full list of texts and analysis is available from authors).

In order to discover the distribution texts present in the stylistics space S, a visualization tool is needed. As there are $\sim$ 1500 dimensions in that space (see table 1), a projection over all possible two-dimensional spaces is out of the question. Also, not all of the variables are necessarily relevant to define the stylistics. We have then two issues to solve: the visualization task and the



List of authors					
		No.	Earliest	Latest	
Name	Language	texts	contrib	contrib.	Label
Arthur Conan Doyle	En	11	1890	1914	ACD
Carlos Fuentes	Sp	5	1962	1999	CF
Donna Leon	En	16	1992	2011	DL
Edgar Allan Poe	En	10	1835	1844	EAP
Gabriel García Márquez	Sp	3	1967	1985	GGM
Iris Murdoch	En	7	1954	1995	IM
Jorge Luis Borges	Sp	32	1935	1975	JLB
Juan Rulfo	Sp	11	1953	1955	JR
Martin Cruz Smith	En	6	1981	2007	MCS
Philip Ball	En	4	2003	2007	PB
Robert Louis Stevenson	En	3	1893	1898	RLS
Stephen Jay Gould	En	13	1977	2002	SJG
WSOM7,WSOM9,WSOM11	En	30	2007	2011	WS
		151			

ACD
1 a case of identity
2 a scandal in bohemia
3 the adventure of empty house
4 the adventure of the nobel bachelor
5 the aventure of abbey grange
6 the horror of the heights
7 the hound of the baskervilles
8 the red headed league
9 a study in scarlet
10 the valley of fear
11 the sign of the four

GGM
1 cien angos de soledad
2 cronica de una muerte anunciada
3 el amor en los tiempos del colera

DL
1 death at la fenice
2 the anonymous venetian
3 death in a strange country
4 a venetian reckoning
5 acqua alta
6 the death of faith
7 a noble radiance
8 fatal remedies
9 friends in high places
10 a sea of troubles
11 uniform justice
12 doctored evidence
13 blood from a stone
14 girl of his dreams
15 a question of belief
16 drawing conclussions

CF
1 aura
2 la muerte de artemio cruz
3 la cabeza de la hidra
4 gringo viejo

JLB
1 el atroz redentor lazarus morell
2 la viuda ching pirata
3 hombre de la esquina rosada
4 el espejo de tinta
5 el impostor inverosimil tom castro
6 el jardin de los senderos que se bifurcan
7 el milagro secreto
8 funes el memorioso
9 la biblioteca de babel
10 la loteria en babilonia
11 las ruinas circulares
12 tres versiones de judas
13 el aleph
14 la casa de asterion
15 emma zunz
16 el zahir
17 la escritura de dios
18 abenjacan el bojari muerto en su laberinto
19 el evangelio segun marcos
20 el informe de brodie
21 historia de rosendo juarez
22 el otro duelo
23 juan muranha
24 guayaquil
25 el libro de arena
26 avelino arredondo
27 el otro
28 el soborno
29 utopia de un hombre que esta cansado
30 la noche de los dones
31 el espejo de tinta
32 el congreso

EAP
1 berenice
2 the fall of the house of usher
3 the murders in the rue morgue
4 the thousand and second tale of schereade
5 a descent into the maelstrom
6 the pit and the pendulum
7 eleonora
8 the black cat
9 the premature burial
10 the purloined letter

IM
1 under the net
2 a severed head
3 the sea the sea
4 jacksons dilemma
5 a word child
6 the good aprentice
7 a fairly honorable defeat

MCS
1 gorky park
2 polar star
3 red square
4 havana bay
5 wolves eat dogs

PB
1 feynmans fancy
2 nature patterns flow
3 pattern formation in nature
4 universe of stone

RLS
1 the black arrow
2 the strange case of dr jekyll and mr hyde
3 the treasure island

SJG
1 bacon brought home
2 creating the creators
3 curveball
4 dinosaur deconstruction
5 i have landed
6 not necessarily a wing
7 piltdown in letters
8 play it again
9 return of hopeful monsters
10 second guessing
11 shades of lamarck
12 the confusion over evolution
13 the exaptive excellence of spandrels

JR
1 anacleto morones
2 diles que no me maten
3 el llano en llamas
4 es que somos muy pobres
5 la cuesta de las comadres
6 la noche que lo dejaron solo
7 luvina
8 macario
9 nos han dado la tierra
10 pedro paramo
11 talpa

Fig. 1. Authors and their works studied in this contribution

identification of a subset of variables (dimensions) that indeed are enough as to identify the stylistics.

In fig. 2-a it is shown the SOM formed for all variables in space S. It is observed that, although some texts from the same author tend to be mapped in clusters, this is not a general fact for all authors. In fig. 2-b, it is shown a SOM for the analyzed texts, but now embedded in the attribute space defined only by the eight MIF shown in table 1.

The stylistics space S includes several features, including relative frequencies, MIF and power spectrum. We are interested now in the following question: Is there a subset of $A \in S$ such that authors may be recognized based on their texts position on that space A? In order to give an answer to that question, we applied a recently introduced method for variables selection [18] based in information theory. We are interested in finding at most K variables ($K < dim(S)$) from S such that such that the mutual information (MI) from A to the class Z (author's name) is maximal. Let $\Phi(A, Z)$ be the mutual information between systems A and Z. We seek to find A such that $\Phi(A, Z)$ is maximum.

This task differs from what information-based algorithms as C4.5 follows. We are not interested in classifying objects based on MI. We are trying to find a subspace such that the coordinates in that space give as much information about the label or class as possible. Then, a machine learning algorithm can be fed with vectors in that space A, instead of being fed with vectors from space S whose dimension is higher. The task we have declared is somehow similar to that followed in algorithms such as testors [16], in which a matrix of differences is systematically explored to identify those features that correctly classify patterns.

We intend to find an attribute space such that the MI between points, representing texts in that space and authors, is as maximum as possible. To do this, the MI of a compound system is needed. That is, if there is only one

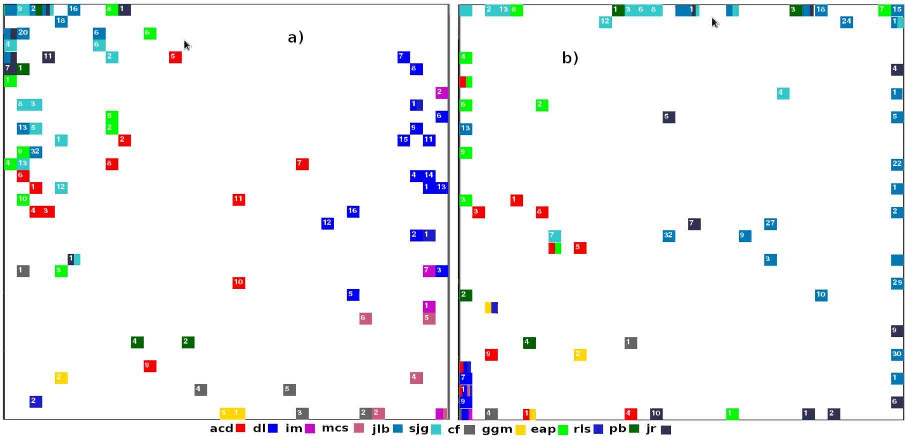


Fig. 2. a) SOM formed on a 30x30 lattice for all texts from stylistics space S. When texts of different authors are mapped to the same unit, the square is divided in equal slices for each text and colored accordingly to the author code. b) Map formed for the input space of all MIF (see table 1). The number in the cell indicates the text index shown in fig. 1.

attribute then the MI is calculated straightforward. In the case of two continuous attributes $X^{'}$ and $Y^{'}$ and ns is the number of states in which each attribute is to be discretized (X and Y), a compound system Z is constructed as follows. $Z'_i = X_i \times ns + Y_i$ and $Z = discretize(Z^{'}, ns)$. For more than two attributes, the procedure is applied recursively.

The náive scheme to construct the space A from S will be to select the K most informative variables. Such strategy is followed, for example, by C4.5 [15] but that greedy strategy leads to local optima. The space generated by K attributes from space S is called A. The number of possible spaces A is the number of combinations of K positions available to D different attributes $C(D, K)$. The exhaustive search for the case here presented is prohibitively time- consuming for $K > 3$. Thus, a search scheme is needed in order to select the relevant features [17]. We applied an heuristic search method, a genetic algorithm, in order to find at most K attributes from T that generate a space such that $\Phi(A; Class)$ is maximum.

A genetic algorithm was implemented in Python with an elitist scheme and probabilities of mutation of 0.05 and crossover of 0.9. Population size was settled to 200 and the algorithm was allowed to run for 1000 epochs. Note that the algorithm identifies a space A of dimension $D \leq K$. That space is not easily observed once $D > 3$. In order to visualize the distribution of the analyzed texts in that space, we decanted our options towards the SOM.

Fig. 3 shows the SOM achieved by different K values. The image on the left corresponds to a SOM for a space A of 27 dimensions, which include MIFThe, MIFMCWy, among others. The image on the right is a SOM for a space A of

5 dimensions that corresponds to MIFSL and MIFMCWy It can be observed that, indeed, there are detectable general distribution patterns that may allow to discriminate the author. Texts do not necessarily form clusters: once again, we are interested in an attribute space such that mutual information between the distribution and the author of a text is maximized. Clusters are only one way in which that mutual information can be maximized, but there are many others. Our methodology finds a family of those distributions.

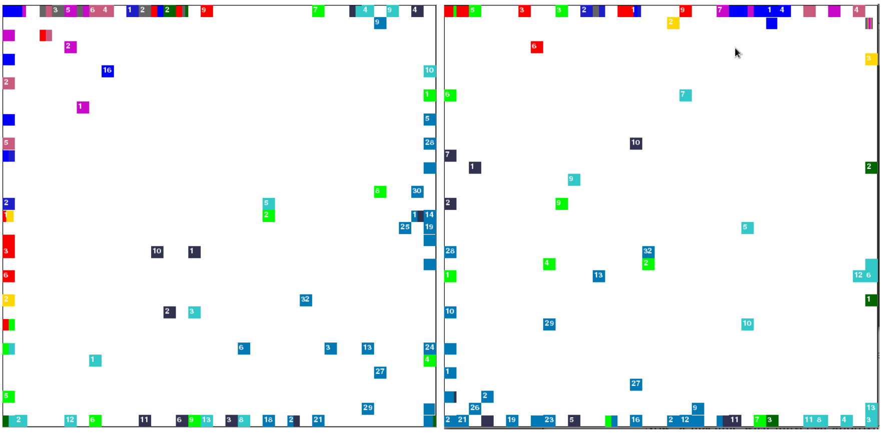


Fig. 3. SOM formed on a 30x30 lattice for all texts from stylistics space $A \in S$. a) A of dimension 27 b) A of dimension 5. Both spaces A were obtained by the genetic algorithm mentioned in the text.

The B time series mentioned in the previous section is interesting because it summarizes the rate at which writers include new words in the text. If now we define space A as specifically the MIF for B, the distribution of that space is approximated in the SOM in fig. 4-a. In general, texts from the same author are similar in that space, that is, they are located in similar areas (see 4-b), but there are some exceptions: Iris Murdoch presents a clear evolution in the style if defined as MIF of B (fig. 4-c). This is consistently with the fact that her last novel (*Jacksons Dilemma*) was written at the time she was suffering from a brain disorder, so a change in her style was expected.

In a different experiment, the label associated to each text was not the author but the language in which it was written. We applied the described genetic algorithm to find the attributes that maximize the MI about the language (class) and we show two SOM for two different conditions in fig. 5. In both cases there are variables that once again include attributes related to MIF, but now, regarding the use of the most common word which is not an article/preposition. Also, a variable selected by the algorithm was the maximum distance between the most common word (including article/preposition).

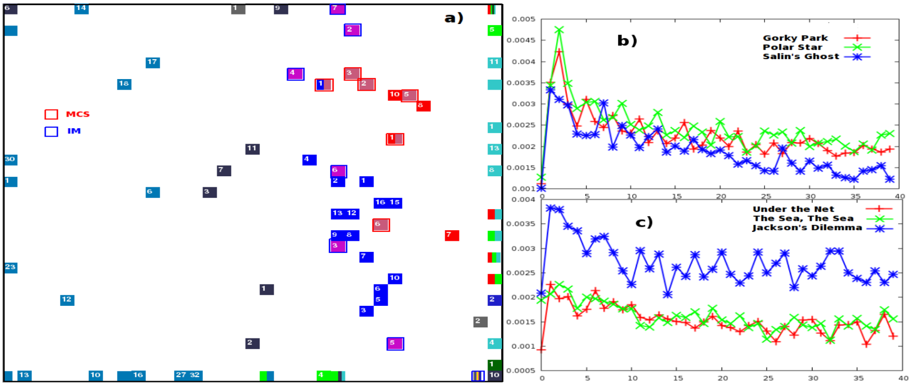


Fig. 4. SOM for input space A defined as MIF for B (a). In b) it is show the MIF for one of the authors (MCS). c) Shows MIF for three texts of author IM.

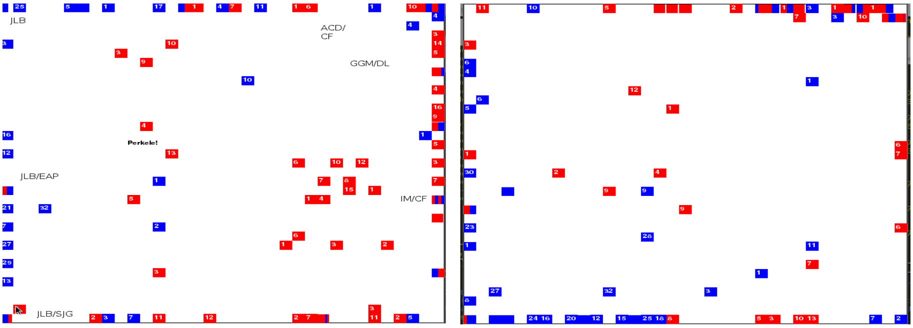


Fig. 5. SOM for two input spaces $A \in S$. MI between A and the language of the text was maximized. Red: English, blue: Spanish. Left: $\dim(A) = 8$, Right: $\dim(A) = 11$.

4 Conclusions

In the tasks of authorship attribution and computational stylistics, it is of major interest to identify a set of attributes that can offer as much information as possible about the author of the text. Here, we have applied a self-organizing map to visualize the distribution followed by several texts from different authors. A genetic algorithm that constructs a space of at most K attributes such that it maximized the information about the class or author of the text was implemented and the distribution of texts in that space was visualized with the SOM

The analysis of texts as time series is also powerful to distinguish between authors. The stylistics is at least partially, well described by the particular pattern authors follows when using certain words. From those patterns it is possible also to analyze the evolution of the style. The methodology here described can be applied to any kinds of texts and it consistently shows that the properties of the extracted time series are relevant to distinguish the stylistics and thus are valuable in the authorship attribution task.

Acknowledgement. A. Neme thanks SNI CONACYT and ICyTDF.

References

1. Juola, P.: Authorship attribution. NOW Press (2008)
2. Stamatatos, E.: A survey of modern authorship attribution methods. J. of the American Soc. for Information Science and Technology 60(3), 538–556 (2010)
3. Canter, D.: An evaluation of Cusum stylistics analysis of confessions. Expert Evidence 1(2), 93–99 (1992)
4. Garrard, P., Maloney, L.M., Hodges, J.R., Patterson, K.: The effects of very early Alzheimer's disease on the characteristics of writing by a renowned author. Brain 128, 250–260 (2005)
5. Mayer, R., Rauber, A.: On Wires and Cables: Content Analysis of WikiLeaks Using Self-Organising Maps, pp. 238–246 (2011)
6. Neme, A., Cervera, A., Lugo, T.: Authorship attribution as a case of anomaly detection: A neural network model. Int. J. of Hybrid Intell. Syst. 8, 225–235 (2011)
7. Manning, C., Schutze, H.: Foundations of statistical natural language processig. MIT Press (2003)
8. Lagus, K., Kaski, S., Kohonen, T.: Mining massive document collections by the WEBSOM method. Information Sciences 163(1-3), 135–156 (2004)
9. Abarbanel, H.: Analysis of observed chaotic data. Springer (1996)
10. Kantz, H., Schreiber, T.: Nonlinear time series analysis, 2nd edn. Cambridge Press
11. Cellucci, C.J., Albano, A.M., College, B., Rapp, P.E.: Statistical Validation of Mutual Information Calculations: Comparison of Alternative Numerical Algorithms. Physical Review E 71(6) (2005), doi:10.1103/PhysRevE.71.066208
12. Shannon, C.E.: A Mathematical Theory of Communication. Bell System Technical Journal 27, 379–423, 623–656 (1948)
13. Kohonen, T.: Self-organizing maps, 2nd edn. Springer (2000)
14. Hujun, Y.: The Self-Organizing Maps: Background, Theories, Extensions and Applications. Studies in Computational Intelligence (SCI) 115, 715–762 (2008)
15. Quinlan, R.: Programs for Machine Learning. Morgan Kaufmann Publishers (1993)
16. Cortes, M.L., Ruiz-Shulcloper, J., Alba-Cabrera, E.: An overview of the evolution of the concept of testor. Pattern Recognition 34, 753–762 (2001)
17. Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. J. of Machine Learning Res. 3, 1157–1182 (2003)
18. Hernández, S., Neme, A.: Identification of the minimal set of attributes that maximizes the authorship information (to appear in LNCS, 2012)

Verification of Metabolic Syndrome Checkup Data with a Self-Organizing Map (SOM): Towards a Simple Judging Tool

Heizo Tokutaka[1], Masaaki Ohkita[1], Nobuhiko Kasezawa[2], and Makoto Ohki[3]

[1] SOM Japan Inc., Tottori, Japan
[2] Ayurvastra Japan
[3] Tottori University, Tottori, Japan
tokutaka@somj.com

Abstract. Minor cases of the metabolic syndrome (MS) for which the lipid, blood pressure, and blood glucose levels are at the border between “normal” and “abnormal” require careful monitoring. We devised a method that addresses this issue by first introducing a “non-ill” condition between the former two. Based on observations of the MS indicator distribution of all examinees, the checkup data was then labeled as "normal", "non-ill" and "abnormal" and applied to a Self-Organizing Map (SOM) whose aim was to visualize the MS indicator distribution in relation to the 3 patient conditions mentioned. Our method was then validated by comparing the MS judgment results with those obtained using the conventional method. The ability to visualize with our method the positional relations between the MS indicators and the 3 conditions further adds to its usability as a health guidance tool.

Keywords: Metabolic Syndrome, Health Evaluation, Self-Organizing Map (SOM), Medical Checkup Data.

1 Introduction

In Japan, an evaluation method for the Metabolic Syndrome (MS) was created that is part of the early lifestyle-related disease prevention program [1]. The diagnostic criteria of MS were developed by the Japanese Society of Internal Medicine and 7 other related societies, and are now widely enforced in Japan. At present, according to the diagnostic criteria, the "extraordinary" condition refers only to the case of slightly exceeded reference values for cholesterol levels, as well as blood pressure and blood sugar levels. The condition “abnormal” depends on the number of exceeded reference values. However, when observing medical checkup data in practice, as time series, there are many cases for which the values sometimes come close to the border of the reference values making it difficult to judge them. Therefore, the MS indicators should be carefully monitored for slight deviations. As a way to address this problem, we introduced the concept of "non-ill area" [2], which are defined as the gray zone between the upper limit of the normal values and the high abnormal ones (i.e., the critical

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 275–284.
DOI: 10.1007/978-3-642-35230-0_28

region). The distribution of each MS indicator, recorded during the patient's general medical checkup, was observed and the data categorized into three areas: "normal", "non-ill" and "abnormal". Then, the check up data categorized in this way (thus, according to our method) was applied to a Self-Organizing Map (SOM) [3, 4, 5, 6, 7] for calculating the MS score and for observing trends in the MS condition of the examinee.

The abdominal circumference must first exceed its reference value in the MS method that is at present recommended in Japan. Then, the case is judged as "MS" or "between" (MS and non-MS) if more than 2 or 1, respectively, out of the following 5 items exceed their reference values: blood sugar level (GLU), systolic blood pressure, the diastolic blood pressure, acylglycerol (TG), and HDL cholesterol. However, with this method, every item is added to the list of MS indicators that is required for classifying the case as abnormal even when their reference values are only slightly exceeded. Also, for the non-obesity type, the MS condition of the patient can not be evaluated from the mentioned items, except for abdominal circumference, even when the reference values for the indicators are exceeded. We established a "non-ill area" outside the range of the reference values. A SOM method was developed to display the MS indicators (MS map) and the similarity with the conventional method was verified.

With our method [2], the difference between a slight deviation from the reference value and a large one could be evaluated by linear interpolation. Also, from the MS maps, one can assess the effect of taking medication and the effect of a change in one's habits, e.g., by first having a meal and then to do some exercise, and so on, which we expect will further encourage the examinee. In the current paper, the MS judgment based on the proposed method is compared with the conventional one. Given our method's distinctive features we believe it can be a useful health guidance tool for the Ministry of Health, Labor and Welfare.

2 Data Preparation

2.1 Reference Values for MS Analysis

The details of the reference values for the Metabolic Syndrome (MS) used in the current study are detailed in Section 4.

2.2 Data Pre-processing

When creating the SOM map, the following normalization procedure [2] was applied. Define the minimum reference value as L, the maximum reference value as H, the data value as X, and the normalized value as Y:

$$\text{When } (X < L); \qquad Y=L/X \tag{1}$$

$$(L <= X <= H); \qquad Y=1 \tag{2}$$

$$\text{When } (X > H) \qquad Y=X/H \tag{3}$$

With this procedure, all normalized values exceed 1. Also, some parameters exceed high values. Since it is our purpose to evaluate a general lifestyle disease, then, we have to decide on a ceiling-value. For example, a high ceiling value is defined as HCV. Then, the normalized values of (1), (2), and (3) are subtracted by 1. Also for (1) and (3), the calculation proceeds as (Y-1)/(HCV-1). Then, all the normalized values belong to the [0, 1] interval.

2.3 Proposed Method

To evaluate the stage of MS, we proceed as follows: -1) - the degree of obesity is averaged by BMI and the abdominal circumference after normalization. -2) - the degree of carbohydrate metabolism is averaged by FBS and HbA1c. -3) - the degree of High blood pressure is averaged by H-BP and L-BP, and -4) - the degree of lipid metabolism abnormality is averaged by TG and HDL. All 4 components are equally considered in our method.

2.4 Calculation of MS (score)

The MS score from which we can obtain the MS degree is calculated using the following procedure [2]. First, the health mark point is determined by

$$Health\ mark\ po\mathrm{int}_i = \frac{\sqrt{\sum_{n=1}^{n}(WV_n - NV)^2} - \sqrt{\sum_{n=1}^{n}(\chi_{ni} - NV)^2}}{\sqrt{\sum_{n=1}^{n}(WV_n - NV)^2}} \times 100 \quad (4)$$

Where $\mathrm{WV_n}$ is the worst value of the respective parameter, NV the normal value, χ_{ni} the data of the i-th examinee and 'N' the number of parameters (here 4) in the pertinent MS case. When the examinee's data is in the normal range, χ_{ni} becomes NV. Then, the Health mark point (HMP) becomes 100 points from eq. (4). However, the MS score can be calculated by MS(score)=100-(HMP). Therefore, the healthy examinee's MS (score) is 0 and the worst examinee's 100.

3 Objective and Method

Our research started with Metabolic Syndrome (MS) medical checkup data of 19,151 men and 10,483 women from April 2007 to March 2009 in Shizuoka, Japan. In order to balance the data set, the female data were used twice in the analysis. Informed consents stating the purpose of the study and the protection of the patient's privacy were obtained.

There were 8 MS indicators, classified into four categories: carbohydrate metabolism, degree of obesity, extraordinary fat symptom, and high blood pressure. First, the data was pre-processed by grouping them into the categories mentioned. Next, for training the SOM, the tool [2] was used. We choose a Torus type of SOM [7].

The details of the tool to judge the MS components of criteria 1-4 are listed in Section 3 and the MS scores displayed in Fig. 1 (mapping) using the SOM developed earlier [2]. There, the medical checkup data and the diagnosis of 8458 men and 4497 women from April 2009 to March 2010 were first analyzed by the existing method and then by the proposed SOM-based method. We continue with our method and evaluate it here by comparing its performance to the existing method. First, the obtained results for both cases were compared visually using the planar SOM map [2, 7]. The cases for which the 2 methods disagreed were selected. The validity of the judgment reached by the 2 methods was re-evaluated. First, the correctness of each test was confirmed. Then, the elapsed change in the values for the examinee was investigated using SOM maps.

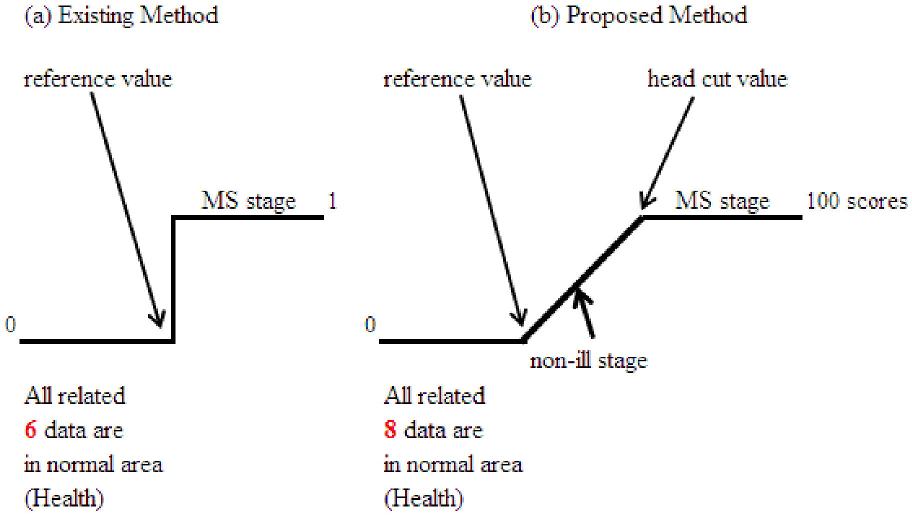


Fig. 1. The difference of the metabo definition between (a) the existing and (b) the proposed method. In (a), the BMI and HbA1c data are removed.

As shown in Fig. 1(a)-, when the examinee’s data exceeds the reference value- , he/she is immediately classified as MS stage by the existing method. However, by the proposed method, when his/her data exceeds the reference value, he/she is first classified as non-ill stage using eqs. (1-4).

4 Characteristics of the Proposed Method

The MS diagnostic criteria used in the proposed method are as follows:

1. A Body Mass Index (BMI) of more than 25; an abdominal circumference above 85 cm for men and above 90 cm for women;
2. A fasting blood sugar level (FBS) above 110mg/dl; a HbA1c above 5.5%;
3. A systolic blood pressure (H-BP) above 130 mmHg; a diastolic blood pressure (L-BP) above 85mm Hg
4. A acylglycerol (TG) of more than 150mg/dl; a HDL cholesterol (HDL-C) below 40mg/dl.

The above 8 criteria were considered as pairs and their means were calculated pairwise as well. The pairs are regarded as the 4 components of: 1) the degree of obesity, 2) the carbohydrate metabolism, 3) high blood pressure, and 4) fat abnormality. The method for determining and normalizing the metabolic label (MS score) were calculated using eqs. (1-4) [2]. In the proposed method, when all items fall within their reference value ranges, the MS score becomes 0 and the position on the map is marked in blue. Apart from that, we consider the following 4 regions depending on the MS score.

Region I: The non-MS region (0 < score < 20) , "DM-normal",
Region II : The MS boundary region (20 <= score < 40), "between" in Table 1,
Region III: The region as MS corresponding (40 <= score < 60), "DM-MS",
Region IV: The MS region (60 <= score <= 100), "DM-MS",
Levels I – IV are displayed by 4 consecutive shades of gray in the MS score maps.
The descriptions "DM-normal", "between", and "MS" are used in Table 1, where DM is the abbreviation for Doctor Metabo which refers to our tool. "DM-normal" refers to the non-MS zone in our proposed method and similarly for "DM-MS".

In the existing method a stage is labeled metabolic if an item exceeds its reference value; else it is judged to be normal. In the proposed method, we have the normal range and a boundary region above it. When the data takes a value in this region, then it will be considered as a level II case. We label this region as non-ill. Even if an item exceeds its reference value by a small amount, the examinee's condition is not judged as abnormal immediately, but his/her data is further evaluated step-by-step.

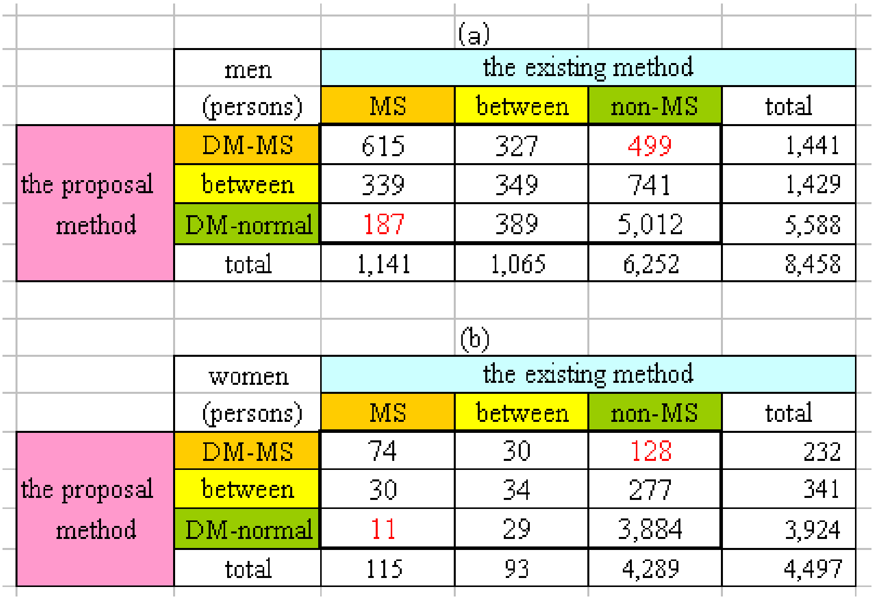
5 Results and Discussion

Using data from 8458 men and 4497 women, the existing method was compared with the proposed one as shown in Table 1: In total, 1141 men and 115 women were labeled as MS abnormal according to the existing method. Also, 187 men (16%) and 11 women (10%) were labeled as normal according to the proposed SOM method and the inconsistencies occur at a high rate. The possibility of having an overestimate with the existing method was also considered as it can not be ignored. On the other hand, 6252 men and 4289 women in Table 1 were judged to belong to the non-MS condition according to the existing method. Also there, according to the proposed method, 499 men (8%) and 128 women (3%) were judged to belong to the MS group. These were further examined as they could be false negatives, overlooking cases of "hidden obesity". The result is shown in Table 1. Such a result could be due to the difference between the existing and the proposed methods, as shown previously in Fig. 1. In other words, the MS cases that were overlooked by the existing method, and that were wrongly considered as MS cases, were clarified by the proposed method. Moreover, as to the non-MS patients, the group in the region between non-MS and MS, and the MS group, as shown in Table 1, were judged by the proposed method. The results for the male and female patients are shown in Fig. 2, for all MS groups, and in Fig. 3 for an example male and female patient of the DM-normal group, all by using the existing method.

The 1141 men and 115 women of Table 1(a) and (b), respectively, which became labeled by the existing method, are shown in Fig. 2 together with the metabolic degree score map and the component map of the proposed method. Also, the 2 overestimated cases, among 187 men and 11 women in Table 1(a) and (b), are shown and explained by using the metabolic degree judgment bar graph, the component map, and the metabolic degree score map of the proposed SOM method (Fig. 3). By using historical data of the corresponding patients, an overestimate by the existing method was revealed (Fig.3).

Next, let us consider the case where the need for changing the standard normal value range into a more convenient one becomes apparent. By using the tool that comes with the proposed method, it is possible to implement the change easily only by changing in the configuration file setting.csv the value that prescribes the normal range. An example is shown in Fig. 4.

Table 1. Comparison between the results of the existing (government recommended) method and the proposed one for (a) men and (b) women



(a)

	men	the existing method			
	(persons)	MS	between	non-MS	total
the proposal method	DM-MS	615	327	499	1,441
	between	339	349	741	1,429
	DM-normal	187	389	5,012	5,588
	total	1,141	1,065	6,252	8,458

(b)

	women	the existing method			
	(persons)	MS	between	non-MS	total
the proposal method	DM-MS	74	30	128	232
	between	30	34	277	341
	DM-normal	11	29	3,884	3,924
	total	115	93	4,289	4,497

All panels in Fig. 3 show men and women that are in the “MS normal area". The arrow in the female case indicates that the abdominal circumference, the H-BP (the systolic blood pressure), and the L-BP (diastolic blood pressure) exceed their standard values only by a small amount. For the male case, the values of the waist, H-BP, and TG, are slightly not normal. However, according to the existing MS judgment method, their cases are labeled according to the "MS criteria".

Finally, another advantage of the proposed method is that in the configuration file one can specify the reference value used for the evaluation. In Fig. 4, the changes in the SOM maps following the change in the reference values stored in the csv file are shown. When the data of the patient is examined according to the reference values as in (a), GLU:100 and HbA1c:5.2, the carbohydrate metabolism and the blood pressure are in the range requiring further observation. When the reference values are loosely defined as in (b) GLU:110 and HbA1c:5.5 then, for this patient, the carbohydrate metabolism is in the first 2 years in "the observation required" condition. Then, only for the high blood pressure, the reference value is left as it is. Therefore, the examinee's condition remains "observation required" because of the high blood pressure. The component maps are shown in (c) GLU:100, HbA1c:5.2 and (d) GLU:110, HbA1c:5.5. In (d), the carbohydrate metabolism becomes small in 2008 and in 2009. And in 2010, only due to the high blood pressure the case is labeled as "observation required".

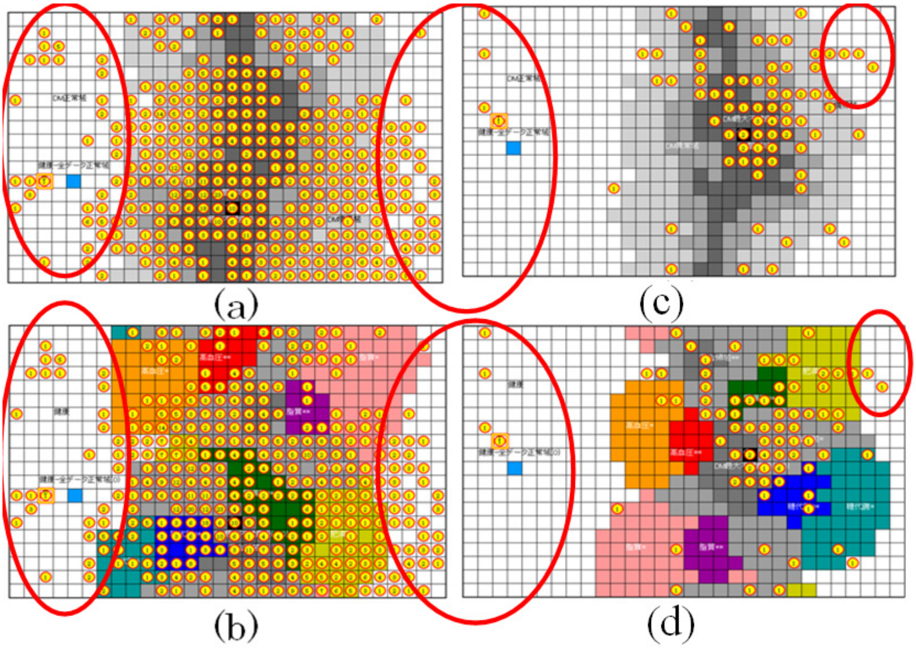


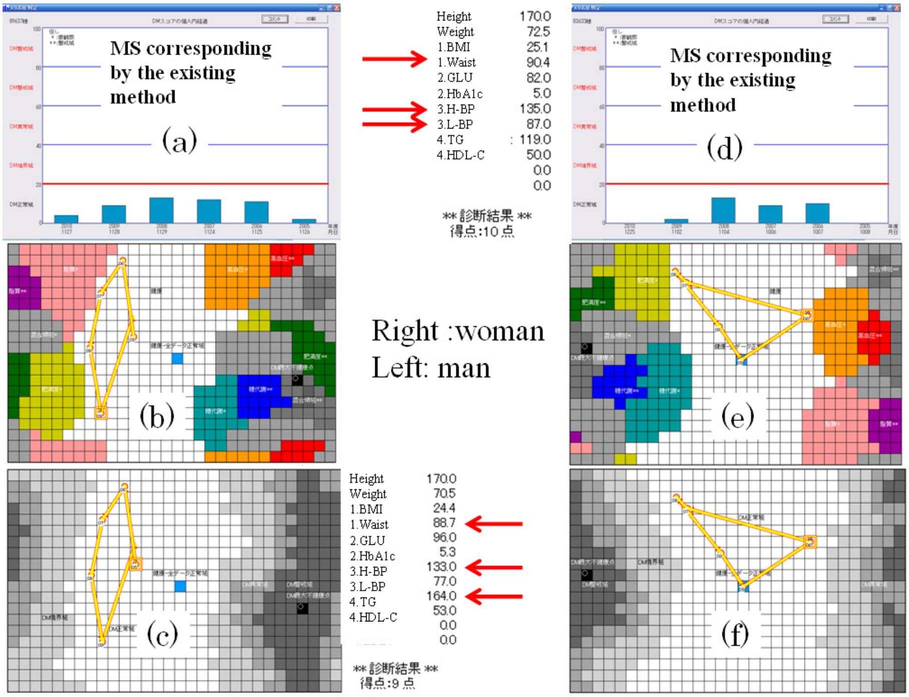
Fig. 2. The score map (a) and the component map (b) for 1141 men, the score map (c) and the component map (d) for 115 women comprising the MS cases labeled by the existing method (Table 1). The 187 male cases (Table 1(a)) and the 11 female cases (Table 1(b)) that were overestimated, i.e., that in fact had to be in the DM-normal area, are marked by red circles. The numbers in the small yellow circles refer to the frequencies of occurrence for such cases.

6 Summary

By using the health checkup database, the existing method and the proposed one were compared. As shown on the left side of Fig. 1, the existing method decides that a case is

MS abnormal, irrespective of the fact that the data point is in the reference region or not. To address this issue, the proposed method considers a marginal region (non-ill) as shown in Fig. 1(b). When the data is taking values in the critical region, it was assigned to this marginal region. This was done to ensure that an accidental large value would not affect the other patient's data values when a head-cut (bound) value for each item was introduced. Thus, when the data slightly went out of the reference range, the data was incorporated into the non-ill area and it was evaluated by linearly interpolating it with the value that largely exceeded the non-ill range. In this way, one avoids an overestimation of the existing method when the patient's MS condition only once and only to a small degree exceeds the reference range.

Hence, we believe that in order to improve the reliability of the MS labeling by the existing method, a close inspection of the overestimated cases (where the possibility of a false positive is high) and of the overlooked ones (where the possibility of a false negative is high) is necessary and should be done by using another method such as the proposed SOM-based method. The above results are summarized as follows:


MS corresponding by the existing method
(a)
MS corresponding by the existing method
(d)
Right :woman
Left: man
(b)
(e)
(c)
(f)

Height	170.0
Weight	72.5
1.BMI	25.1
1.Waist	90.4
2.GLU	82.0
2.HbA1c	5.0
3.H-BP	135.0
3.L-BP	87.0
4.TG	119.0
4.HDL-C	50.0
	0.0
	0.0

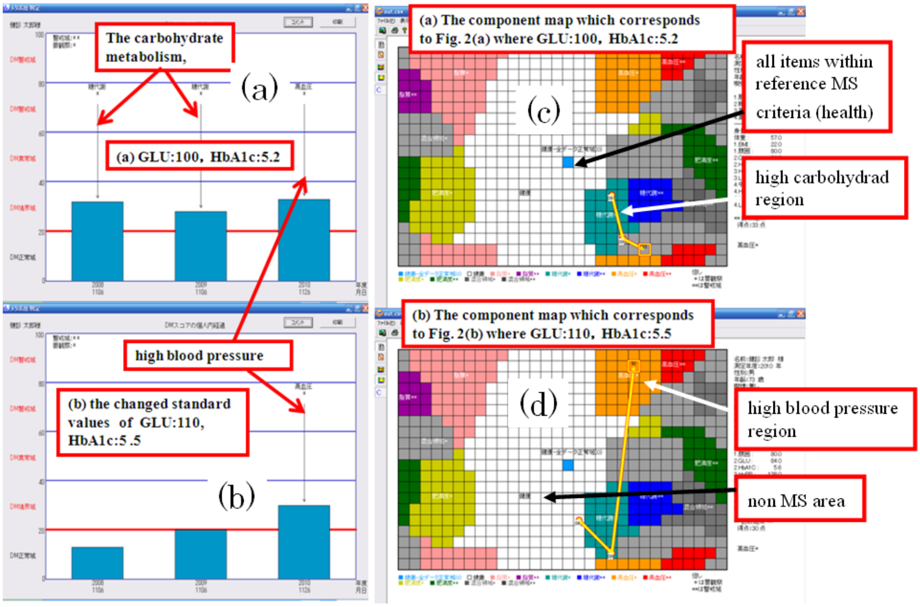
** 診断結果 **
得点:10 点

Height	170.0
Weight	70.5
1.BMI	24.4
1.Waist	88.7
2.GLU	96.0
2.HbA1c	5.3
3.H-BP	133.0
3.L-BP	77.0
4.TG	164.0
4.HDL-C	53.0
	0.0
	0.0

** 診断結果 **
得点:9 点

Fig. 3. Example of a man (Left panels) and a woman (Right panels) whose MS criteria abdominal circumference (waist) and 2 other criteria exceed a reference value, as shown by the man's (a) and woman's (d) MS score bars plotted per year. The panels (b) and (e) are the component maps, for the man and the woman, and panels (c) and (f) are the respective score maps, all obtained with the proposed SOM method. The yellow trajectories in (b), (c) and (d), (f) connect the data points that correspond to the years listed along the horizontal axis of panels (a) and (d), respectively.

1. The existing method which starts from a high abdominal circumference has a risk of overlooking MS cases where the remaining 5 items in Section 1 are ignored whether they pose a high risk or not.
2. The existing method can overestimate certain cases because it only decides based on whether or not a reference value is exceeded.
3. For the existing method to judge a case as a metabolic one, the abdominal circumference must exceed a reference value. However, in the proposed method, criteria 1-4 in section 4 are evaluated in addition to the abdominal circumference. Since the proposed method has a less chance to overlook cases, extra medical expenses can be avoided.
4. By changing the boundary values in the csv-file of the tool, the metabolic degrees and component maps can be freely explored.



The carbohydrate metabolism,

(a) GLU:100, HbA1c:5.2

high blood pressure

(b) the changed standard values of GLU:110, HbA1c:5 .5

(a) The component map which corresponds to Fig. 2(a) where GLU:100, HbA1c:5.2

all items within reference MS criteria (health)

high carbohydrad region

(b) The component map which corresponds to Fig. 2(b) where GLU:110, HbA1c:5.5

high blood pressure region

non MS area

Fig. 4. Changes in the score bars (a,b) and the component maps (c,d) incurred when changing the reference values in the csv file of the tool from GLU:100, HbA1c:5.2 (a,c) to GLU:110, HbA1c:5.5 (b,d). Same convention for the yellow trajectories is as in Fig. 3. The figure was compiled using all available data in order to show the robustness of the tool.

Acknowledgements. The authors are very obliged to Prof. M. Van Hulle of KU Leuven, Belgium, for kindly reading and correcting our manuscript.

References

1. The metabolic syndrome diagnosis standard exploratory committee: The definition and the diagnosis standard of the metabolic syndrome. The Journal of the Japanese Society of Internal Medicine (J. Jpn. Soc. Int. Med.) 94, 188–213 (2005)

2. Tokutaka, H., Maniwa, Y., Kihato, P.K., Fujimura, K., Ohkita, M.: Application of SOM in a health evaluation system. In: WSOM 2007, Bielefeld Germany, September 3-6 (2007)
3. Tokutaka, H., Fujimura, K., Ohkita, M.: Cluster Analysis using Spherical SOM. In: WSOM 2007, Bielefeld Germany, September 3-6 (2007)
4. Nakatsuka, D., Oyabu, M.: Application of Spherical SOM in Clustering. In: Proceedings of Workshop on Self-Organizing Maps (WSOM 2003), pp. 203–207 (2003)
5. Kasezawa, N., Tohyama, K., Nakano, M., Hirota, K., Morishita, T., Tokutaka, H.: Usefulness of computer-based support tool based on SOM for metabolic syndrome checkup and aftercare in health evaluation and promotion. HEP 38, 574–583 (2011)
6. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer (2005)
7. Ohkita, M., Tokutaka, H., Fujimura, K., Gonda, E. (eds.): The Self-Organizing Maps and the tools, ch. 4. Springer-Japan Inc. (December 2008) (in Japanese)

Analysis of Farm Profitability and the Weighted Upscaling System Using the Self-Organizing Map

Mika Sulkava, Maria Yli-Heikkilä, and Arto Latukka

MTT Agrifood Research Finland, Economic Research,
Latokartanonkaari 9, FI-00790 Helsinki, Finland
{mika.sulkava,maria.yli-heikkila,arto.latukka}@mtt.fi

Abstract. Profitability and other economic aspects of farming in Finland are analyzed using the self-organizing map. The analysis of profitability bookkeeping data reveals several interesting relationships between the monitored financial variables. A weight optimization system is presented for upscaling financial figures of the sample of profitability bookkeeping farms to the whole country level. The self-organizing map is also used to assess the performance of the weighting system. It is confirmed that the most important large and medium-sized enterprises are represented well by the sample. Furthermore, it seems that the utilized arable area is the key factor in guiding the weight optimization process. These findings may turn out to be useful in developing the sampling of bookkeeping farms in the future.

Keywords: farm, profitability, bookkeeping, self-organizing map, upscaling, weight, sample, optimization, constraint, agriculture.

1 Introduction

Profitability of farm enterprises is very important as it makes it possible for the farms to stay in business in the long run and, thus, be a part of a stable food supply chain. Farm profitability has been fluctuating strongly in Finland during the recent years [1]. This may complicate the farmers' planning for the future.

In this paper, the self-organizing map (SOM) is used to analyze financial data of agricultural and horticultural enterprises. The data are collected from a sample of bookkeeping farms, and they are the source of many figures characterizing Finnish agriculture in the EconomyDoctor service of Agrifood Research Finland [2]. In addition, a weighted upscaling system for obtaining country-level results based on the sample is presented and analyzed using the SOM. The goal is to discover interrelations between financial variables and find out how different kinds of farms are represented by the sample based on the weighting, cf. [3].

The SOM has been successfully used in financial analysis, e.g., benchmarking of industrial companies [4]. A simple SOM analysis of the relationships within the bookkeeping farm data will be published in 2012 [5]. The data have also been analyzed with the aim of understanding input substitution and technological development of farms [6] and finding changes in productivity [7,8]. In addition,

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 285–294.
DOI: 10.1007/978-3-642-35230-0_29

neural networks have been used in predicting the sufficiency of internal financing of farms [9].

The organization of the rest of the paper is as follows: in the next section we present the data, in Section 3 the structure of the weighting system is introduced, in Section 4 the SOM and related parameters are explained, the results are shown in Section 5, and conclusions drawn in Section 6.

2 Profitability Bookkeeping Data

Annual profitability figures for Finnish agricultural and horticultural enterprises showing the average results of over 60 000 enterprises are calculated from the profitability bookkeeping organized by MTT Agrifood Research Finland. Profitability of Finnish farms is monitored using a sample of approximately 1 000 farms yearly. Data from the year 2010 are used in this study. In 2010 there were 940 bookkeeping farms. The original aim has been to represent the 40 000 largest enterprises of Finland, which is why the sample contains only a few small farms.

The form of bookkeeping data is similar to the data in the Farm Accountancy Data Network (FADN) [10]. There are thousands of variables in the bookkeeping data bank. The variables used in this study were selected by an expert. The aim was to select variables that have potential of providing a diverse picture of the economic performance – especially solvency and profitability – of farm enterprises. The following variables are used to characterize each bookkeeping farm i: economic size e_i, utilized arable area a_i, support payments, total gross return, entrepreneur's profit, livestock units, interest claim, equity ratio, return on assets, entrepreneurial income, profitability ratio, return on equity, hourly earnings, total assets, equity, interest rate, wage and interest claim, liability pay-back period[1], debt-%, working hours, rented arable area, type of farming, and support area.

The wage cost of own labor in 2010 is calculated using an hourly wage claim of 14 €. The interest cost of equity is calculated on the basis of a farm-specific interest rate, which is the sum of the risk-free interest rate and a farm-specific risk premium. When the compensations for labor input and own capital are deducted from entrepreneurial income, we obtain the entrepreneur's profit. The profitability ratio is defined as ${}^{E}/_{(W+I)}$, where E is the entrepreneurial income and W and I are the wage and interest claims, respectively [11]. When the profitability ratio is 1, all production costs have been covered and the entrepreneur's profit is zero [2].

In addition, structural data of agriculture containing the total number of farms and total utilized areas in the support areas, size classes, and types of farming have been calculated based on farm register data obtained from Information Centre of the Ministry of Agriculture and Forestry Tike.

According to a Regulation of the European Commission, there are 14 economic size classes of farms. In the EU farm production is divided into about 60 types.

[1] Liability pay-back periods above 50 years were considered uninformative and were, therefore, truncated.

Ten types of farming are present in Finland, some of which are combinations of more specific EU farm types. In addition, there are seven support areas in Finland. Table 1 shows the economic size classes, types of farming, and support areas from south (A) to north (C4).

Areas are reported in ha in the data and the currency unit is €. Livestock units are defined as grazing equivalents of dairy cows, i.e., small animals count for less than one livestock unit. See [12,2,9] for more information on the calculation of financial variables, and [13,12] on the determination of types of farming.

3 Weighting System

MTT Economic Research calculates annually the result and profitability development of Finnish agriculture and horticulture. In this total calculation the results for the whole country are obtained by summing up the weighted results of the bookkeeping farms [14]. A weighting system is presented in this section for obtaining reliable upscaling results based on the bookkeeping farms. The total results for the country's over 60 000 farms are, thus, calculated by summing up the weighted figures of the bookkeeping farms.

Weighting coefficients are calculated annually for each bookkeeping farm by numeric optimization so that when multiplied by the weighting coefficients and summed up the number of farms and cultivation areas correspond to the total number of farms and cultivation areas both in the whole country and in each support area. Within the support areas the weighting based on the number of farms is done according to farm size classes. By weighting according to the farm size classes the results can be made to correspond to the real farm size distribution in Finland.

The weighting is only based on the number of farms and total cultivation areas, on which there is aggregate information available for the whole country.

Table 1. Numbering of economic size classes, types of farming, and support areas

	Economic size (€)	Type of farming	support area
1	$e_i < 2\,000$	Cereal farms	A
2	$2\,000 \leq e_i < 4\,000$	Other crop farms	B
3	$4\,000 \leq e_i < 8\,000$	Horticulture, indoor	C1
4	$8\,000 \leq e_i < 15\,000$	Horticulture, outdoor	C2
5	$15\,000 \leq e_i < 25\,000$	Dairy farms	C2p
6	$25\,000 \leq e_i < 50\,000$	Cattle farms	C3
7	$50\,000 \leq e_i < 100\,000$	Sheep, goats and other grazing livestock	C4
8	$100\,000 \leq e_i < 250\,000$	Pig farms	
9	$250\,000 \leq e_i < 500\,000$	Poultry farms	
10	$500\,000 \leq e_i < 750\,000$	Non-classified	
11	$750\,000 \leq e_i < 1\,000\,000$		
12	$1\,000\,000 \leq e_i < 1\,500\,000$		
13	$1\,500\,000 \leq e_i < 3\,000\,000$		
14	$e_i \geq 3\,000\,000$		

These statistics are available in the Structural Development service of the MTT EconomyDoctor [2]. No financial variables are used in calculating the weighting coefficients, because no sufficiently reliable region or country-level aggregate figures are available. The weights are optimized separately for each year.

The weighting system consists of two phases. First, initial weights d_i are assigned to the bookkeeping farms i in the sample B. The initial weights are calculated for each farm using the weighting system of the FADN of the EU [10]. In this system the weighting coefficient of a farm in a specific year depends on how large a number of farms it represents in its own type of farming and economic size in its support area. The types of farming and economic sizes are determined for the whole period covered on the basis of the standard outputs introduced in the EU in 2010 [11].

Second, the weights w_i of the bookkeeping farms $i = 1, \ldots, N$ are adjusted with sequential quadratic programming to fulfil a set of constraints. The aim is to produce correct total values for certain variables which are known from other sources. The updating phase is a constrained optimization problem:

$$\min_{w_i} \sum_{i \in B} (w_i - d_i)^2 \tag{1}$$

subject to

$$\sum_{i \in S_j} w_i a_i \geq (1-t)A_j,\ \forall j \in \{1, \ldots, s\} \tag{2}$$

$$\sum_{i \in S_j} w_i a_i \leq (1+t)A_j,\ \forall j \in \{1, \ldots, s\} \tag{3}$$

$$\sum_{i \in E_k} w_i \geq (1-t)N_{E,k},\ \forall k \in \{1, \ldots, g\} \tag{4}$$

$$\sum_{i \in E_k} w_i \leq (1+t)N_{E,k},\ \forall k \in \{1, \ldots, g\} \tag{5}$$

$$\sum_{i \in T_m} w_i \geq (1-t)N_{T,m},\ \forall m \in \{1, \ldots, f\} \tag{6}$$

$$\sum_{i \in T_m} w_i \leq (1+t)N_{T,m},\ \forall m \in \{1, \ldots, f\} \tag{7}$$

$$w_i \geq 1,\ \forall i, \tag{8}$$

where S_j is the set of bookkeeping farms in support area j, t is the tolerance between the true and upscaled values, A_j is the total cultivated area in support area j, s is the number of support areas, E_k is the set of bookkeeping farms belonging to economic size group k, $N_{E,k}$ is the total number of farms in economic size group k, g is the number of economic size groups, T_m is the set of bookkeeping farms belonging to type of farming m, $N_{T,m}$ is the total number of farms of farming type m, f is the number of types of farming.

In other words, the weights are calibrated to match the cultivated area in each of the $s = 7$ support areas (Eqs. 2 and 3) and the numbers of farms in economic size groups (Eqs. 4 and 5) and all $f = 10$ types of farming (Eqs. 6 and 7). In Finland, the number of farms in the smallest and largest size classes is rather low. Therefore, and in order to reduce the number of constraints, only $g = 4$ size groups are used in the weighting system. This is achieved by combining size classes 1–4, 5–6, 7–8, and 9–14. The tolerance was set to $t = 0.01$ since with this value all constraints could still be fulfilled.

4 Self-Organizing Map

The self-organizing map (SOM) [15] is a useful tool in exploratory data analysis. It projects multidimensional data into a low-dimensional grid which is easy to visualize. In addition to nonlinear projection, the SOM also performs vector quantization. This representation can be used for visualization, clustering, and exploration of data. Conceptually, the SOM and its map units form a flexible net in the data space. This makes visualization of the grid useful in exploring the relationships of variables and the possible cluster structure of the data.

In this study training and analyzing the SOM was performed using the SOM Toolbox for Matlab [16]. Before training, the number of map units and the structure of the grid in the SOM are defined. The number of map units was chosen based on the default setting of SOM Toolbox, i.e., $\left\lceil 5\sqrt{N} \right\rceil$. We used hexagonal grid sheet structure and the default ratio of the side lengths: $\sqrt{\lambda_1/\lambda_2}$, where λ_1 and λ_2 are the two largest eigenvalues of the autocorrelation matrix.

The observations were normalized linearly before training, e.g., so that the mean of each variable is 0 and the variance is 1. The method used to normalize the data defines the distance between multidimensional vectors. For example, how should a change in return on asset percentage be related to a change in utilized area measured in hectares? Normalizing all the variances to unity solves this problem by defining that changes in different variables are equal if they are in equal proportion to their standard deviations. As a result, all variables have equal weights in this sense.

The map units are connected to neighboring units on the grid by the neighborhood function. Gaussian neighborhood function was used and $\sigma(t)$ corresponds to the width of the Gaussian function. The training is divided into a rough training phase and a fine-tuning phase. $\sigma(t)$ decreases during the rough training phase. The batch algorithm was used to train the SOM.

The map can be visualized using component planes, each of which shows the values of one of the original variables as colors on the grid. In addition, the map can be visualized with the unified distance matrix (U-matrix) [17], which shows the within-unit distances and distances between neighboring units on the grid.

The quality of the map can be measured with the quantization error, which is the average distance between each observation and its best-matching unit. In addition to quantization, the topology preservation of the projection can be measured with the topographic error [18]. It is defined as the percentage of observations for which the best-matching unit and the second-best-matching unit are not neighboring units on the grid.

5 Results

5.1 Farm Profitability

An economic map of the bookkeeping farms was produced using the SOM. Support area, type of farming, and the weight variables were not used to adapt the map in the training. These classification and weight variables were masked from the training because they do not characterize the economic status of the farms. Some types of farming are not visible on the corresponding component plane at all. This is partly due to the fact that there are fewer farms of those types present in the bookkeeping data as well as in Finnish agriculture in total. The quantization error of the map was 2.16 and the topographic error 0.06. The map is thus well organized. The U-matrix and component planes of the SOM are show in Figure 1.

The U-matrix suggests that there may be some cluster structure in the data, but the possible cluster boundaries are not very sharp. However, farms characterized by extreme conditions can be found separated from other farms in both top corners and the bottom left corner of the map.

Different economic types of farms can easily be spotted using the map. The top left corner corresponds to large, mainly dairy farms with the highest total assets and equity, the highest utilized and rented arable areas, and the most livestock. Also cereal, cattle, and pig farms are common in that part of the map, and the farms are often located in southern parts of the country. These farms receive the highest support payments and have the highest wage and interest claims. The entrepreneur's profit, however, is the most negative in this area. The hourly earnings are also negative despite the large number of working hours. Thus, the profitability of these farms is usually not very good, in fact it is even below 0.2. The high interest rates indicate that these enterprises are rather risky.

The top right corner of the SOM represents the largest farms with the highest incomes, returns, and profitability. On the other hand, the equity ratio of those farms is rather low and the liability pay-back period is very long. These farms are typically horticulture or dairy farms. Clearly smaller farms but similar in terms of solvency and profitability are in the middle of the right border of the map. These smaller farms are mostly classified as other crop farms.

In the mid-left part of the map medium-sized, medium-profitability farms can be spotted with above average debt-% and very low equity, equity ratios, and interest claims. The amount of labor is high on some of these farms but the utilized areas are low. These typically horticulture and dairy farms are concentrated more to the northern part of Finland.

Low-risk farms with low interest rates, debts, high equity ratios, and short liability pay-back periods are distributed in the middle of the map and the bottom right part. The main differences between these two are in economic size and types of farming. The former are mainly dairy farms and the latter – mainly cereal farms – are the smallest with the least working hours and smallest wage and interest claims. The small farms are also more commonly located in the

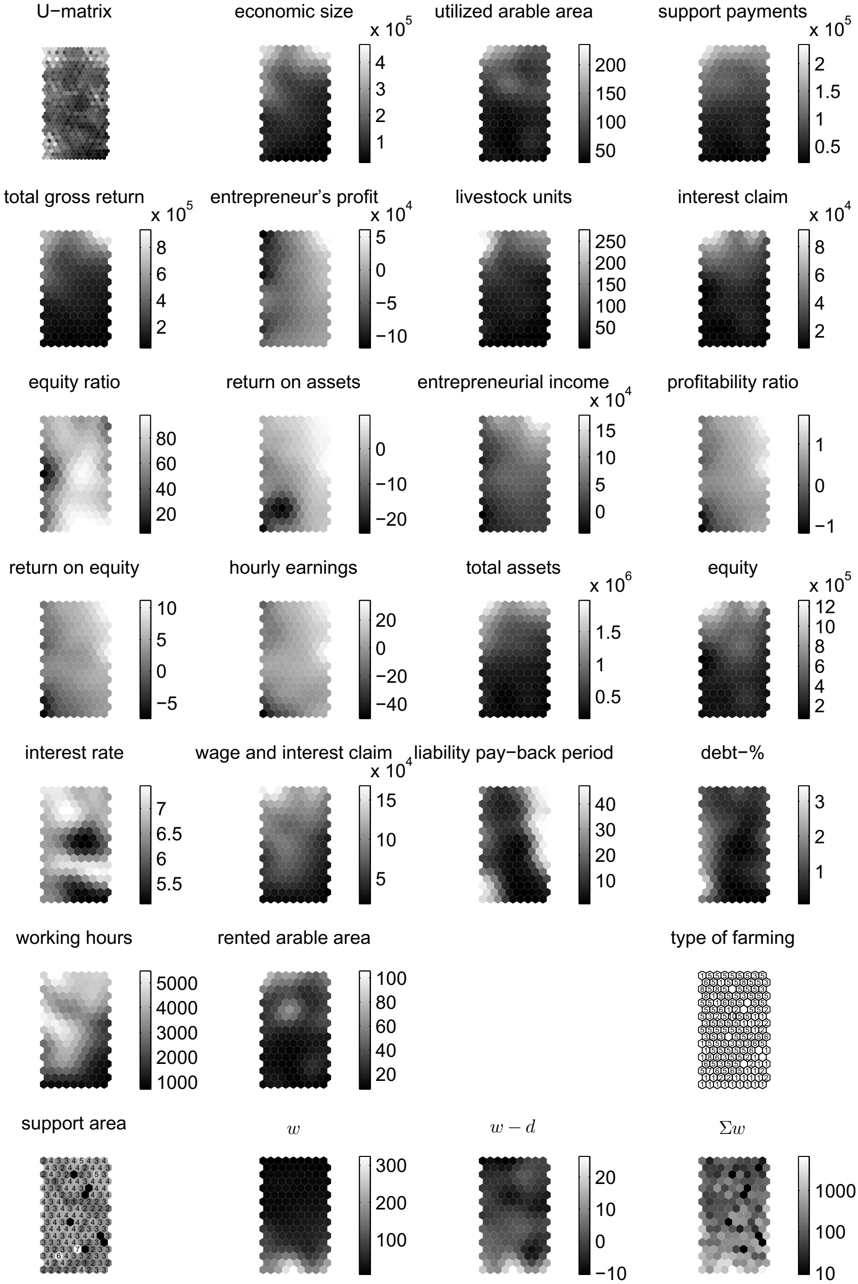


Fig. 1. U-matrix and component planes of the SOM of farm bookkeeping data and weights. The values in type of farming refer to the most common type of farming in the corresponding map unit. The values and colors in support area refer to the median value of the map unit. Type of farming and the variables in bottom row were not used to adapt the map in the training.

southern support areas. Both farm groups have average profitability ratios and are rather homogeneous based on the U-matrix.

The bottom left corner of the SOM has the least profitable farms. They are small low-equity farms with negative entrepreneurial income, hourly earnings, and return on equity. They have the highest debt percentages and longest liability pay-back periods. As regards types of farming most of these are cereal and dairy farms.

5.2 Upscaling Weights

The largest weights are located in the bottom of the map. Those map units represent two kinds of small farms, mostly in southern Finland. The first group has high equity ratios, short liability pay-back periods, and low interest rates indicating low business risk. The second group is characterized by the lowest returns, hourly earnings, and profitability ratios. Mostly these are cereal farms, but also other crop farms, dairy and cattle farms are represented.

The component plane of d is not shown because it looks essentially the same as the component plane of w. In its stead, the difference between w and d is presented. The differences between weights before and after optimization are relatively small, as can be expected due to the form of the optimized function. Interestingly, the largest increases during weight optimization occur in map units that also had the largest initial weights. So, the smallest bookkeeping farms need to represent an even higher number of farms than in the FADN weighting in order to fulfill the constraints (Eqs. 2–7).

The largest decreases in weights, on the other hand, can be found in four locations on the map: 1. top left corner, 2. slightly above and left from the center, 3. mid-part of right border, and 4. above and left from the bottom-right corner of the map. The two first locations correspond to mainly cereal and dairy farms that have large utilized arable area and a lot of rented lands. The main difference between these two is in economic size. The third and fourth locations correspond to small cereal and other crop farms. In the third location profitability is high but the pay-back periods of liabilities are long, whereas in the fourth location equity ratios are high. All of the four locations have higher utilized and rented arable areas than the surrounding map units.

The component plane of the sum of weights shows how many of the total 60 000 farms are represented by each map unit when the weighting system is used. The scale is logarithmic due to skewed distribution of the parameter. The highest number of farms represented by a single map unit is over 6 700. The distribution of $\sum w$ is rather uneven on the map. There is, however, a tendency that the sum of weights gets smaller towards the top of the map. When considering the whole population, the different types of larger farms on the top part of the map are rather well represented by the SOM. In contrast, a very high number of small farms is represented by a few map units at the bottom part of the map. This part of the SOM mostly contains rather similar prototype vectors based on the U-matrix and also the component planes. That is, these smallest farms do not seem to be very different from each other with respect to financial variables.

6 Conclusions

Using the SOM allowed us to analyze effectively interconnections between financial variables characterizing the performance of agricultural enterprises in Finland. Different kinds of farming could be easily distinguished on the map.

We studied five groups of farms with different profiles of profitability and solvency. Consequently, we came up with the following hypotheses concerning the financial status of Finnish farm enterprises. 1. Enterprises with the most livestock, arable area, highest assets, and equity are risky and have low profitability. 2. The largest horticulture and dairy farms and smaller other crop farms with low equity ratios have the highest profitability. 3. Farms with very low equity-ratios have average profitability. 4. The least risky enterprises are small dairy farms and very small cereal farms with high equity ratios. 5. Small cereal and dairy farms with high debts and long pay-back periods of liabilities are the least profitable.

The analysis of upscaling weights using SOM attests that the large farms in Finland are well represented by the sample of bookkeeping farms. The results obtained by weighting of course involve a degree of uncertainty, because the set of bookkeeping farms cannot fully reflect the highly varied population of Finnish farms and horticultural enterprises. Nevertheless, the distribution of weights is so uneven that not using the weighting would bias the total results.

Farms with above-average utilized arable areas within all size classes experience the largest decrease in weights during the weight optimization. It is likely that the area-related constraints have guided the optimization process. Therefore, it may turn out to be beneficial to increase the number of bookkeeping farms with smaller utilized arable area in the future.

The small farms with low or negative profitability have the largest weights and, thus, also the largest uncertainty in upscaling. The SOM analysis suggests, however, that these small farms are similar to each other, which would have a positive effect on uncertainty. More importantly, the contribution of the smallest farms to the total figures of agriculture is very limited. Therefore, the total calculation gives a comprehensive and coherent picture of the sector as whole. The weighting also enables the use of regularly updated forecasts, representative regional results and results according to production sectors as well as other calculations based on simulations. Analysis of the hypotheses above – as, e.g., in [19], cluster structure in the data, and temporal behavior of farm profitability are left as subjects for future research.

References

1. Rantala, O., Tauriainen, J.: Development of results and profitability of agriculture and horticulture. In: Niemi, J., Ahlstedt, J. (eds.) Finnish Agriculture and Rural Industries 2011. Publications, ch. 4.1, vol. 111a, pp. 52–56. MTT Economic Research, Agrifood Research Finland (2011)
2. MTT EconomyDoctor (June 2012), http://www.mtt.fi/economydoctor
3. Sulkava, M., Luyssaert, S., Zaehle, S., Papale, D.: Assessing and improving the representativeness of monitoring networks: The European flux tower network example. Journal of Geophysical Research – Biogeosciences 116, G00J04 (2011)

4. Eklund, T.: The Self-Organizing Map in Financial Benchmarking. D.Sc. thesis, Åbo Akademi University, Turku, Finland (December 2004)
5. Sulkava, M.: Exploring agricultural data using self-organizing maps. In: Proceedings of the 20th Pacioli Workshop, Rome, Italy (September/October 2012) (accepted for publication)
6. Ryhänen, M.: Input substitution and technological development on Finnish dairy farms for 1965–1991: Empirical application on bookkeeping dairy farms. Agricultural Science in Finland 3(6), 525–601 (1994)
7. Myyrä, S., Pihamaa, P., Sipiläinen, T.: Productivity growth on Finnish grain farms from 1976–2006: a parametric approach. Agricultural and Food Science 18(3-4), 283–301 (2009)
8. Kuosmanen, T., Sipiläinen, T.: Exact decomposition of the Fisher ideal total factor productivity index. Journal of Productivity Analysis 31(3), 137–150 (2009)
9. Latukka, A.: Predicting Financial Distress of Farms using Neural Network Application. Lic.Sc. thesis, University of Helsinki, Department of Economics and Management No. 22, Production Economics and Farm Management, Helsinki, Finland. (December 1998) (in finnish)
10. Farm accounting data network (June 2012), http://ec.europa.eu/agriculture/rica/index.cfm
11. Rantala, O., Tauriainen, J.: Development of results and profitability of agriculture and horticulture. In: Niemi, J., Ahlstedt, J. (eds.) Finnish Agriculture and Rural Industries 2012. Publications, MTT Economic Research, Agrifood Research Finland, ch. 4.1, vol. 112a, pp. 56–61 (2012)
12. Community Committee for the Farm Accountancy Data Network. Typology handbook. Technical Report RI/CC 1500 rev. 3, European Commission – Directorate-General for Agriculture and Rural Development, Brussels, Belgium (October 2009)
13. Committee for Corporate Analysis. The Guide to the Analysis of Financial Statements of Finnish Companies, Gaudeamus, Helsinki, Finland (2006)
14. Latukka, A., Sulkava, M.: Economic development of finnish agriculture and horticulture. In: Niemi, J., Ahlstedt, J. (eds.) Finnish Agriculture and Rural Industries 2012. Publications, ch. 4. 2, vol. 112a, pp. 62–65. MTT Economic Research, Agrifood Research Finland (2012)
15. Kohonen, T.: *Self-Organizing Maps*, 3rd edn. Springer Series in Information Sciences, vol. 30. Springer, Berlin (2001)
16. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: SOM Toolbox for Matlab 5. Report A57, Helsinki University of Technology, Laboratory of Computer and Information Science, Espoo, Finland (2000)
17. Ultsch, A., Siemon, H.P.: Kohonen's self organizing feature maps for exploratory data analysis. In: Proceedings of International Neural Network Conference (INNC 1990), pp. 305–308. Kluwer, Dordrecht (1990)
18. Kiviluoto, K.: Topology preservation in self-organizing maps. In: Proceedings of the International Conference on Neural Networks (ICNN 1996), vol. 1, pp. 294–299. IEEE Neural Networks Council, Piscataway (1996)
19. Sulkava, M., Tikka, J., Hollmén, J.: Sparse regression for analyzing the development of foliar nutrient concentrations in coniferous trees. Ecological Modelling 191(1), 118–130 (2006)

Professional Trajectories of Workers Using Disconnected Self-Organizing Maps

Etienne Côme[1], Marie Cottrell[2], and Patrice Gaubert[3]

[1] IFSTTAR - Bâtiment Descartes 2,
2, Rue de la Butte verte, 93166 Noisy le Grand Cedex, France
etienne.come@ifsttar.fr

[2] SAMM - Université Paris 1 Panthéon-Sorbonne
90, rue de Tolbiac, 75013 Paris, France
marie.cottrell@univ-paris1.fr

[3] ERUDITE, Université Paris 12,
61, avenue du Géneral De Gaulle, 94010 Créteil, France
patrice.gaubert@u-pec.fr

Abstract. Using the Panel Study of Income Dynamics (PSID) collected on the period 1984-2003, we study the situations of American workers with respect to employment. The data include all heads of household (men or women) as well as the partners who are on the labor market, working or not. They are extracted from the complete survey by computing a few relevant features which characterize the worker's situations.

To perform this analysis, we suggest to use a Self-Organizing Map (Kohonen algorithm) with specific topology. In this paper we present a new topology for SOM based on a planar graph with disconnected components (called D-SOM) which is especially interesting for clustering. Each component takes the form of a string and corresponds to an organized cluster.

From this clustering, we study the dynamics at the individual level, that is the trajectories of the individuals among the classes during the observed period. Then we estimate the transition probability matrices for each studied year and the corresponding stationary distributions.

Finally, we try to give an answer to the question: is there a significant change in 1992 (new economic policies after the Reaganomics).

Keywords: Kohonen algorithm, planar graphs, labor market, Markov chains.

1 Introduction

The aim of this study is to identify and to analyze the succession of situations occupied by workers on a modern labor market (1984-2001). The mainstream theory presents mechanisms to explain the level of labor furnished for a specified compensation, the stability of the relation between a firm and a worker and its evolution over time (a career). These mechanisms are not observed in the most

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 295–304.
DOI: 10.1007/978-3-642-35230-0_30

real situations. To identify the diversity of situations in terms of activity is the first step of the study.

A situation is defined by quantitative variables:

– global quality of a job, full time job for the whole year, wages, seniority in the same job versus
– positions with more or less precarious conditions: wages lower than the average, part time jobs, jobs for short periods, on-call jobs, current practice of a second job

Working on individual data we construct a classification of situations observed every 2 years on a specific labor market during two consecutive periods of nine years. With the characteristics of a small set of major situations, it is possible to define the successive localizations of each individual for each studied year of the two periods. That is what we called trajectories between situations.

We need to study the temporal changes, during both sub-periods: 1984-1992 and 1993-2001. It must be possible to answer some important questions linked to the evolution of the macroeconomic environment: in 1992 the end of Reaganomics and the beginning of Clinton period which leads to a global reduction of unemployment. What is the impact of this reduction of unemployment and is there a significant change at the individual level?

This article follows another paper [2] but contains necessary material (and possibly redundant) to be self-contained. It is organized as follows : first, in Section 2, the data and the notations used throughout the paper are presented. The methodology and the global architecture of the proposed procedure are described in Section 3. Each step is defined and results on real data are given in Sections 4 to 7.

2 The Data: First Period (1984, 86, 88, 90, 92) and Second Period (93, 95, 97, 99, 2001)

We use the PSID (Panel Study of Income Dynamics), dividing the observations in two sub-periods in order to solve the trade-off we meet: observe a number of workers large enough to obtain statistical indicators representative of the whole population, from one hand, and from the other hand, to keep only individuals present along the period to identify trajectories.

We create a sample for each period (1984-1992, 1993-2001) but, with the hypothesis that the main situations have the same characteristics in these periods, with differences in levels only, we make the classification with all the observations together.

In the PSID data, we select households for which the head (man or woman) is present every studied year of the period but separately for each sub-period. The administrative rule is that if there is a male in the household he is the head, if not the head is a woman. Fortunately quite the same variables concerning the activity on the labor market are available for the wife of the head, if there is one. Retrieving this information we constitute set of individuals (around 4 500 per year) observed every two years in each sub-periods, with a proportion of women close to the one observed in the whole population.

An observation consists of a couple (year, individual). It is described by 8 quantitative variables and 4 qualitative variables. See Table 1 for the list of variables and their meaning. j

Table 1. Variable name, description and type

Name	Description	Type
nbbhtrav	Number of worked hours per week	Quant
nbstrav	Number of worked weeks	Quant
nbschom	Number of unemployed weeks	Quant
nbsret	Number of weeks out of labor market	Quant
salhor	Wages per hour	Quant
nbex	Number of extra jobs	Quant
hortex	Number of hours worked in extra jobs	Quant
anctrav	Seniority in present work in monthes	Quant
sex	Sex	Qual
naiss	Year of birth	Qual
pro	Professional occupation	Qual
bri	Branch of industry	Qual

3 Disconnected Self-Organizing Maps, D-SOM

Following Come et al. (2010) [2], we use a light variant of the classical SOM ([3], in order to get a map which is composed of several disconnected one-dimensional strings. Each string will contains data which are similar at a rough level and that are displayed in ordered disposition.

To get this topology, it is necessary to define a neighborhood structure which is different from the classical one. Graph theory allows us to define such structures as noted by several authors like [1,4]. If the used graph can be represented in dimension 2, we will still have the advantages of Self-Organizing Maps for visualization and data mining.

See figure 1 an example of disconnected neighborhood structure that we define here.

This topology has a special interest: when the map consists of not connected parts, the "cooperation" step of the algorithm only concerns the units which belongs to the same component as the winning unit. The competition step is not modified, so that the algorithm complies a double goal :

1. to group the observations into macro-classes corresponding to the different connected components of the graph ;
2. to organize the units inside the macro-classes.

The code-vectors are denoted by m_{ij}, $i \in \{1, \ldots, K\}$, $j \in \{1, \ldots, n_i\}$, where K is the number of disconnected components and n_i is the size of component i.

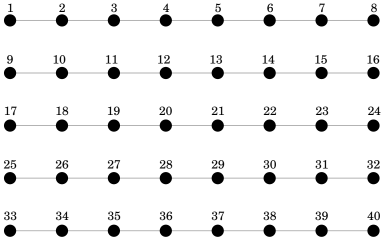


Fig. 1. Bi-dimensional representation of a disconnected map with 5 strings of 8 units

Then the distance $d\left((i,j),(i',j')\right)$ between classes (i,j) and (i',j') is define as the shortest path distance in the graph. It is equal to $+\infty$ if $i \neq i'$. The code-vectors which do not belong to the same macro-class as the winning unit are not updated by the cooperation step.

The algorithm can be written as below:

1. The code-vectors are randomly initialized in the data space ;
2. at each step t, the code-vectors are updated $\mathbf{m}_{ij}(t)$ in the following way :
   - one observation $\mathbf{x}_{t+1}$ is randomly drawn and we achieve two steps;
   - *Competition,* the winning unit is computed for the l'observation $\mathbf{x}_{t+1}$ by:

$$[i^*(t+1), j^*(t+1)) = \arg \min_{i\in\{1,\ldots,K\}, j\in\{1,\ldots,n_i\}} ||\mathbf{x}_{t+1} - \mathbf{m}_{ij}(t)||; \quad (1)$$

   - *Cooperation,* the code-vectors of the winning unit and of its neighbors (which necessarily belong to the same macro-class (i^*, j^*)) are updated by:

$$\mathbf{m}_{i^*j}(t+1) = \mathbf{m}_{i^*j}(t) + \alpha(t)h(t,(i^*,j^*),(i^*,j))\left[\mathbf{x}_{t+1} - \mathbf{m}_{i^*j}(t)\right], \quad (2)$$

   where t is the number of iteration, $\alpha(t)$ is the learning rate and $h(t,(i^*,j^*),(i^*,j))$ is the neighborhood function at step t between classes (i^*,j^*) and (i^*,j).

In conclusion, by imposing a limitation of the cooperation which only acts inside the macro-classes and by keeping a competition between all units, this algorithm allows us to get a classification into a given number of macro-classes which are themselves self-organized.

There exists other methods to get well-separated classes, see [5] for example. But our approach is different since we do not look for building an adjacency matrix between the code-vectors by repeating many runs of the SOM algorithm. Contrarily, we impose an a priori adjacency matrix which defines non-connected classes.

This kind of topology is well adapted in the frame of the labor market segmentation, since one looks for a segmentation into macro-classes well discriminated, easy to describe, split into organized classes. In a general case, the question of the choice of the number of macro-classes is guided by a priori argument if there exists theoretical reasons. In our case we chose 5 macro-classes which is the best choice to get contrasting and well identified situations.

Let us now describe the results that we get using this topology for our data.

4 The Map, Description of the Clusters

Figure 2 shows the about 45000 couples (year, individual) represented by a 8-vector, classified into 5 disconnected macro-classes, themselves composed of 8 units.

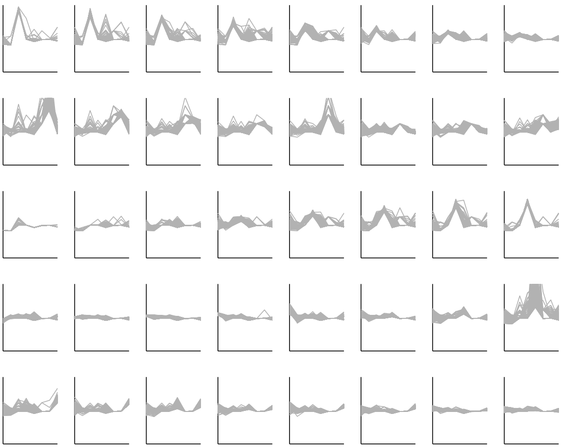


Fig. 2. D-SOM map with 5 macro-classes of 8 units

By computing the arithmetic means (see Table 2 and Table 3) of the eight variables used to make the classification, it is easy to emphasize the contrasts between the macro-classes, (the five strings):

- macro-class 1: precarious, part-time employment and unemployment
- macro-class 2: people having extra jobs (one or more) to obtain a sufficient standard of living
- macro-class 3: people most of the time out of the labor market (discouraged, ill, or for family reasons, and retired people in period 2)
- macro-class 4: full employment with very short seniority in the present place
- macro-class 5: full employment with the highest compensation and seniority (about 18 years)

These findings are obtained for the two sub-periods.

Figure 3 contains five subplots which present the evolution of the code-vectors along a macro-class from unit one to unit eight. All the variables are centered and

Table 2. Mean values for each variables by macro-class, period 1; the figures in bold are the maximum values for each variable, the figures between brackets are the class sizes

	C1 (1493)	**C2** (2736)	**C3** (3756)	**C4** (7443)	**C5** (6686)
nbhtrav	36.99	40.69	8.28	**42.03**	41.63
nbstrav	27.71	47.15	5.68	**48.50**	47.29
salhor	8.25	11.87	2.62	12.80	**14.28**
nbschom	**22.13**	0.54	0.29	0.14	0.11
nbsret	0.96	0.19	**9.48**	0.10	0.01
anctrav	28.95	82.43	5.58	30.20	**168.14**
nbex	0.05	**1.13**	0.01	0.00	0.00
hortex	8.58	**384.83**	1.35	0.77	0.12

Table 3. Mean values for each variables by macro-class, period 2; the figures in bold are the maximum values for each variable, the figures between brackets are the class sizes

	C1 (531)	**C2** (2171)	**C3** (6108)	**C4** (7099)	**C5** (7081)
nbhtrav	35.76	41.07	4.36	**42.25**	41.59
nbstrav	31.11	47.32	3.49	**48.13**	47.31
salhor	14.61	19.70	1.74	19.44	**20.25**
nbschom	**21.16**	0.34	0.06	0.10	0.06
nbsret	2.21	0.60	**3.27**	0.26	0.25
anctrav	29.94	108.62	3.50	29.36	**214.26**
nbex	0.04	**1.11**	0.00	0.00	0.00
hortex	7.99	**397.37**	0.31	0.61	0.00

reduced and are drawn on the same scale $[-5, 10]$. This representation confirms our description.

Figure 4 presents the 8 variables on the whole D-SOM map, with 5 macro-classes of 8 units each one.

5 Transitions

The study of trajectories followed by individuals observed over a period of nine years is obtained computing the transition matrix: it shows the probability to be in one class at year $t + 2$, starting from another class at year t. See Table 5 for the first period and Table 4 for the second one.

The most evident result is that the major part of a class has not moved between year t and year $t + 2$: the important exception to this rule is class one, in each sub-period. A large part of the precarious, unemployed, part-time workers have changed to good jobs two years later, and this phenomenon is even more important in second period.

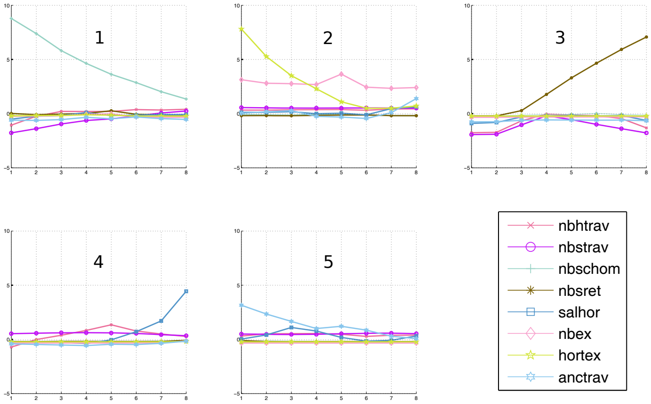


Fig. 3. Multivariate profiles of the different macro-classes

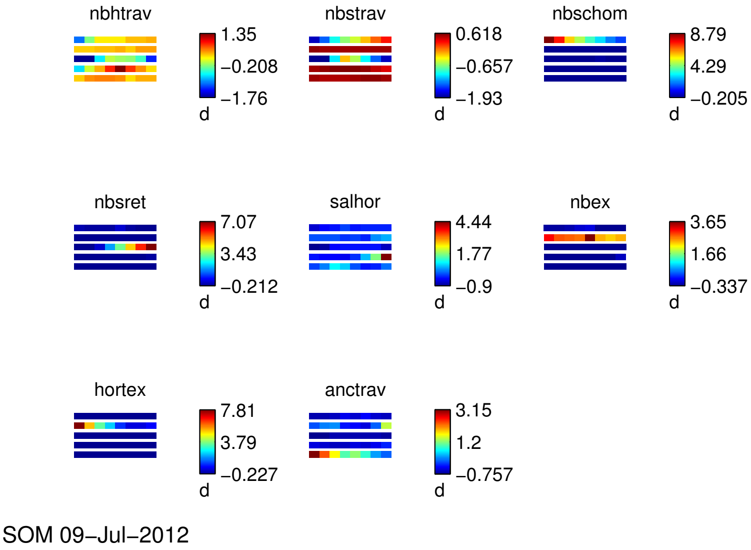


Fig. 4. 8 variables on the whole D-SOM map, with 5 macro-classes of 8 units each one

Of course, the most stable class over both sub-periods is class 5, the one with very stable jobs. The great proportion observed in period 2 of people staying in class 3 is probably due to the effect of people aging while they are observed and definitely leaving the labor market. It is interesting to notice that in the second period a proportion significantly smaller of the class 1 is staying in this class, that is the worst situation. It must be the effect the growing flexibility introduced in the US economy.

Table 4. Transition matrix, period 2, values in bold are maxima

	C1	**C2**	**C3**	**C4**	**C5**
C1	0.26	0.10	0.14	**0.40**	0.10
C2	0.03	**0.52**	0.03	0.26	0.17
C3	0.06	0.03	**0.66**	0.22	0.02
C4	0.06	0.08	0.07	**0.63**	0.16
C5	0.03	0.06	0.02	0.09	**0.79**

Table 5. Transition matrix, period 1, values in bold are maxima

	C1	**C2**	**C3**	**C4**	**C5**
C1	0.09	0.09	0.16	**0.59**	0.08
C2	0.02	**0.45**	0.04	0.31	0.19
C3	0.01	0.01	**0.85**	0.11	0.01
C4	0.02	0.08	0.08	**0.66**	0.16
C5	0.02	0.05	0.03	0.15	**0.75**

6 Limit and Empirical Distributions

For each period, we can compare the observed distributions of individuals across the five macro-classes to the theoretical limit distributions, computed under the hypothesis that everything in the environment stays unchanged. The limit distribution is estimated by iterating the transition matrix, which converges, as shown by Markov Chain Theory, to a matrix whose all rows are the same. So that the transition probabilities do not depend anymore on the starting value. We see Table 6 that there is a change between period 1 and 2. The theoretical and observed distributions are closer, one to the other, in period 2 than in period 1. This indicates that the system has become more stable, i.e. the successive distributions are approximately the same during period 2.

Table 6. Empirical and limit distributions, period 1 and 2

	C1	**C2**	**C3**	**C4**	**C5**
Empirical distribution for the first period	0.07	0.12	0.17	0.34	0.30
Limit distribution for the first period	0.06	0.12	0.13	0.31	0.38
Empirical distribution for the second period	0.02	0.09	0.27	0.31	0.31
Limit distribution for the second period	0.02	0.08	0.29	0.33	0.28

7 Some Results by Gender

Here is the distribution of men and women on the map. See Figure 5

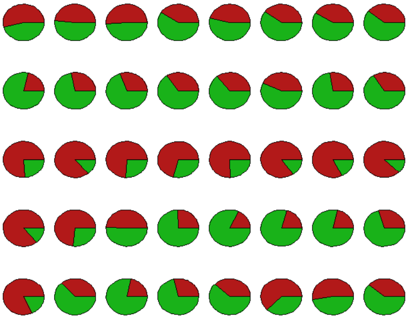


Fig. 5. Men in light grey (green) and women in dark grey (red), explanations are in the text

Knowing that men and women are in close proportions in the two samples (like they are in the whole population), it is easy to observe that men are more numerous than the average in the five last units of macro-class 4 and in the most part of macro-class 5 (macro-classes 4 and 5 correspond to the best situations).

They are also the main part of macro-class 2, the one where workers have one extra job or more.

At the same time women are a great proportion, from 2/3 to 4/5, of those who, for a while or definitely, are out of the market (macro-class 3).

If we look at the transitions matrix for the 2 periods and the genders, one can see that (we do not display them for lack of space):
- the major fact for both genders is the withdrawal from the market in the second period (people discouraged and/or not registered as present on the labor market or retired)
- men are leaving part-time jobs or true unemployment (macro-class 1) to obtain full-time unstable jobs (macro-class 4) in a greater proportion in second period, women move in a greater proportion towards the class 3, as in period 1
- from macro-class 4 of full-time jobs without seniority, women are more leaving towards the withdrawal (macro-class 3), while this move is very weak for men.

8 Conclusion

From this real-world example, we showed that using a Disconnected Self-Organized Map algorithm facilitates the clustering of numerous data, by providing a segmentation into easy-to-interpret clusters, themselves being divided into well-organized classes. Then the classification can be interpreted at two levels that are of interest.

The number of macro-classes is a priori defined, equal to the number of clusters that the experts have identified. This is the "supervised" part of the algorithm. The interest of SOM which is a non-supervised method is that each cluster can be described by the population it contains, and that we can retrieve its main characteristics.

At the second level, each cluster is, in turn, split into micro-classes, which are mainly organized according to the value of one of the input variables. This fact provides a refined description of each cluster's population. In the future, we want to study an automatic method to select the topology, the number of macro-classes, the size of the strings, in the framework of model selection.

References

1. Barsi, A.: Neural Self-organization Using Graphs. In: Perner, P., Rosenfeld, A. (eds.) MLDM 2003. LNCS, vol. 2734, pp. 343–352. Springer, Heidelberg (2003)
2. Côme, E., Cottrell, M., Verleysen, M., Lacaille, J.: Self organizing star (sos) for health monitoring. In: Proceedings of the European Conference on Artificial Neural Networks, Bruges (Belgium), pp. 1341–1346 (April 2010)
3. Kohonen, T.: Self-Organizing Maps. Information Sciences (1995)
4. Pakkanen, J., Iivarinen, J., Oja, E.: The elvoving tree - analysis and applications. IEEE Transactions on Neurals Networks 17, 591–603 (2006)
5. Resta, M.: Assessing the Efficiency of Health Care Providers: A SOM Perspective. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 30–39. Springer, Heidelberg (2011)

Understanding Firms' International Growth: A Proposal via Self Organizing Maps

Marina Resta* and Riccardo Spinelli

Department of Economics, University of Genova, via Vivaldi 5, 16126, Genova, Italy
{resta,spinelli}@economia.unige.it
http://www.economia.unige.it

Abstract. We discuss the use of Self Organizing Maps (SOMs) to address an issue the recently gained increasing relevance among economics scholars, i.e. how to measure firms level of international growth. We will demonstrate that SOMs can be a very powerful technique with superior features with respect to more traditional techniques (namely: K–means clustering). Our arguments will be supported by the empirical evidence, as we investigated the United Nations Conference on Trade and Development (UNCTAD) database.

Keywords: International growth, Self Organizing Maps, K–means clustering.

1 Introduction

Measures of corporate internationalization have recently gained crucial importance, as it is agreed that globalization is strongly related to firms international activities [4].

As reported in [7], measuring firms' internationalization may first have a phenomenalistic justification of its own; in other words, a measure is needed to get an operational definition of Multinational Corporations (MNCs), thus distinguishing them from domestic ones. Moreover, these indicators may be applied to extract more subtle distinctions of international involvement, such as the local, regional and global corporate orientation [8].

In this spirit, a growing number of studies focused on the proposal of *ad hoc* indexes, to establish the relation between firms' degree of internationalization and other corporate features such as financial performance, diversification, and managerial practices [13], [15], [16]. The rationale is to use either such measures or a combination of them as explanatory variable to model MNCs behavior [14].

With respect to the existing literature, in this paper, we suggest an innovative use of internationalization indexes, as we treat them as *proxies* of companies' structural features. Our aim is to classify MNCs via Self Organizing Maps (SOM) [9], in accordance to the extent and shape of their international projection, thus extracting homogeneous groups of firms which share a similar approach to foreign

* Corresponding author.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 305–314.
DOI: 10.1007/978-3-642-35230-0_31

markets. This true, in fact, our results could have a great impact in strategic terms, aiding managers to give better address to firms operative policy.

As far as we know, despite the variety of both economic and financial applications of SOMs (see for instance: [17], [18], [19], and [20] just to cite some), they have not yet been employed to study such an issue, even if SOMs could give a significant contribution and improve the quality of the results: our experience (that we are going to detail in next sections) shows that SOMs are superior in comparison with traditional clustering techniques like those employed in [6].

Holding this, the structure of the paper is as follows. In Section 2 we introduce the theoretical background underlying the measurement of corporate internationalization; in Section 3 we describe the set of data we examined and a brief insight on the methodologies we compared is given. Section 4 provides results and discussion issues. Section 5 concludes.

2 A Review on Corporate Internationalization Measures

The issue of corporate internationalization measurement is so relevant that a number of methodologies have been proposed to face it. Basically they can be distinguished with respect to three elements:

– the aspect of internationalization they aim to analyze;
– the variables they consider;
– their nature of individual or composite indicators.

The first issue points on the fact that there are several ways of assessing companies degree of internationalization, which depend on what patterns and aspects of we choose to emphasize. Within this strand, major aspects that are usually measured are the internationalization intensity (i.e. the share of foreign activities in total activities) and the internationalization breadth (i.e. the geographical dispersion of corporate foreign activities). This latter is sometimes coupled to a *cultural* dispersion analysis [21], [5]. Moreover, whereas some authors investigated intensity and breadth separately ([6]), others jointly examined them ([7], [1]).

The choice of the aspect to investigate, in turn, deeply conditions the selection of the variables to consider. To such purpose, it aids to remember that there is a large number of variables that can be considered when building the index, related to very different features of corporate activity: financial and economic indicators, management characteristics, geographical variables, etc. The basic issue, in this case, stands in whether or not composite indicators (i.e. made up by more than a single explicative variable) may be suitable to measure corporate internationalization. As argued in [4], they could be preferable for several reasons: firstly, internationalization is a multidimensional phenomenon so that limiting the measurement to one single item inevitably would lead to represent only a part of the whole phenomenon. As second remark, depending on what indicator is used, one could be lead to contradictory results. Moreover, individual indicators are much more subjected to measurement errors and contingent influences. Finally, individual measures are not so interchangeable, making it very difficult to draw comparisons between results derived from empirical studies using different proxies for the degree of internationalization. Composite indicators try to overcome these problems, since they condensed in a single index several variables.

With this in mind, we turned our attention to the Transnationality Index (TNI) and its components, firstly introduced by the United Nations Conference on Trade and Development (UNCTAD) in 1995. The TNI is an average among three variables: the foreign share in sales (FSTS), employment (FETE) and assets (FATA). It is calculated for the 100 biggest MNCs world-wide, and published annually in UNCTAD's World Investment Report.

The conceptual framework underlying this index is based on the dichotomy between foreign versus home country activities [11], and helps to assess the degree to which activities and interests of MNCs are embedded either in their home economy or in economies abroad. Obviously, TNI has a number of drawbacks: [10] argued that a high TNI value could be biased by a small home–country, which tends to overestimate the international projection; this is particularly evident for those TNCs coming from small industrial countries such as Switzerland or the Netherlands. Finally, the index is able to measure only the internationalization intensity without any distinction between companies whose foreign activities are concentrated in one or few countries and those whose activities are spread in many foreign countries [11].

Despite all its limitations, however, TNI is widely considered a good index for measuring firms' internationalization; even its detractors recognize that the individual variables it incorporates can sufficiently describe some aspects of the firms' degree of internationalization [10] (p. 706). We agree with this position, and in such a mood we are going to use the three TNI components instead than TNI as a whole.

3 Data and Methodology

3.1 Dataset

We used data provided by United Nations Conference on Trade and Development (UNCTAD) in the Annex to the World Investment Report 2011; the data refer to world's top 100 non–financial TNCs, ranked by foreign assets, in 2010. For each company the home economy is provided, together with the firm's reference industry and the values of foreign and total sales, employment and assets. Finally, TNI is given as the average of the following three ratios:

- foreign sales on total sales (FSTS);
- foreign employment on total employment (FETE);
- foreign assets on total assets (FATA).

In our analysis we employed FSTS, FETE and FATA as clustering variables, in order to identify groups of companies which share a common projection towards foreign markets: the value of every single indicator, in our opinion, could provide useful information to better understand the features of the international development of the considered companies. The rationale is that the value of such indicators can be assumed as a *proxy* of the degree of the firm's international projection. Consider, for instance, a company whose FSTS, FETE and FATA values are all around 0.75: we could conclude that this company concentrates three quarters of its activities outside its home economy, and, as a consequence, it is very intensively internationalized. Moreover, if we consider also the firm's country of origin, we could acquire even more information, thus deducting whether we are facing to a true multinational giant, or to a company which is *by force* multinational, provided its home economy small size. On the contrary, a company whose values are all around 0.25, is able to develop abroad only a quarter of its total activities, and considerations we have discussed in the above rows hold now specularly. Even more interesting, however, are those situations where the three indicators present values which are significantly different one from each other. Among the various possible combinations, consider, for instance, the case of similar FETE and FATA values, which, in turn, are much less than FSTS: in this case we would probably be observing a company which has its production capacity and infrastructure still based in the home economy, while its projection to foreign markets is mostly export-based. The symmetrical situation (high FETE and FATA and low FSTS), conversely, would lead us towards a company which has strongly delocalized its production abroad, but still keeps its domestic market as the most important one. Finally, different values of FETE and FATA could indicate dissimilar labor intensity per capital unit, being a consequence of different industry structure.

3.2 Methodology

Our analysis was performed by comparing of the results obtained running both K–Means clustering (KM) and Self-Organizing Maps (SOMs). Provided the practical focus of our paper, we will only provide a short insight on both methods; interested reader will be provided with proper references. As widely known, K-means [12] is one of the most commonly-used clustering algorithm. Its major pros rely on its simplicity, as one could easily see by observing the following pseudo–code that describes how it works. Let us assume to denote by $\{x_i\}, (i = 1, \ldots, N)$ the set of N input patterns to be partitioned into K clusters $C_1, \ldots, C_K$ in order to minimize the sum of within cluster dispersion (i.e. the Squared Error –SSE)

1. Initialize K center locations $(C_1, \ldots, C_K)$.
2. Assign each x_i to its nearest cluster center C_i.
3. Update each cluster center C_i as the mean of all x_i that have been assigned as closest to it.
4. Calculate:

   $$SSE = \sum_{i=1}^{K} \sum_{x \in C} \left[d(m_i, x)\right]^2 ,$$

   where d is a proper distance metric (usually the Euclidean norm), and $m_i, (i = 1, \ldots, K)$ is the i–th cluster mean.
5. If the value of SSE has converged, then return $(C_1, \ldots, C_k)$, else go to Step 2.

Over the time major improvements like parallel versions [3] or stochastic underlying framework [22] have been introduced to assure a better tuning of the KM procedure. Unfortunately, KM is so sensitive to the choice of initial starting points –centroids– that if the initial partitions are not chosen carefully, the computation will run the chance of converging to a local minimum rather than the global minimum solution.

This odd is superseded by Self Organizing Map (SOM) due to they way they work. In its simplest form SOM is a single layer neural network, where neurons are set along an n-dimensional grid: typical applications assume a 2-dimensions rectangular grid, but hexagonal as well as toroidal grids are also possible. Each neuron has as many components as the input patterns: mathematically this implies that both neuron and inputs are vectors embedded in the same space. Training a SOM requires a number of steps to be performed in a sequential way. For a generic input pattern x we will have:

1. to evaluate the distance between x and each neuron of the SOM;
2. to select the neuron (node) with the smallest distance from x. We will call it winner neuron or Best Matching Unit (BMU);
3. to correct the position of each node according to the results of Step 2., in order to preserve the network topology.

Steps 1.–3. can be repeated either once or more than once for each input pattern: a good stopping criterion generally consists in taking a view to the Quantization Error (QE), i.e. a weighted average over the Euclidean norms of the difference between the input vector and the corresponding BMU. When QE goes below a proper threshold level, say for instance 10^{-2} or lower, it might be suitable to stop the procedure. In this way, once the learning procedure is concluded, we get an organization of SOM which takes into account how the input space is structured, and projects it into a lower dimensional space where closer nodes represent neighboring input patterns.

4 Results Discussion

The application of the KM method to our dataset required the choice of the number of clusters to identify. This choice represents one of the main limits of the technique and it can bias the significance of the results. Two to six–cluster k–means solutions all showed significant F–tests for the three variables. To infer the correct cluster number, we run a pseudo–F test [2]. Pseudo-F increases up to the two-cluster solution, suggesting the latter as optimal. The two-cluster solution yielded F-values larger than 77.954 (all p-values .0000), as reported in Table 1.

Table 2 shows final centroids and proportions for the two clusters.

The cluster solution obtained with the KM technique shows evident limitations in its explicative capability. The two clusters, indeed, are almost equally populated and seem to gather, respectively, companies exhibiting medium or higher levels of internationalization; companies in both clusters belong to several

Table 1. For each variable we reported basic clusters statistics (Mean Square and degrees of freedom –df–). Similar records were provided for clusters error. Finally, latest two columns report the values of F–test and of Significance –Sig.–, respectively.

	Cluster		Error		F	Sig.
	Mean Square	df	Mean Square	Df		
Assets (FATA)	26892.192	1	147.056	98	182.871	,000
Sales (FSTS)	16435.282	1	210.832	98	77.954	,000
Employment (FETE)	17205.919	1	189.950	98	90.581	,000

Table 2. Final Cluster Centers

	CL01	CL02
% proportion	56%	44%
Assets (FATA)	51.11%	84.15%
Sales (FSTS)	57.49%	83.31%
Employment (FETE)	47.99%	74.42%

different industries and come from various home country. Limited information can be consequently inferred by the proposed solution.

The solutions with three and more clusters – discarded due to the minor value of the pseudo-F test – seem to show the presence of further latent groups of companies, which could be more interesting because of the more articulated values of the three indicators. Nevertheless, these groups have disappeared in the accepted solution, as they merged into the only two significant clusters.

We then moved to apply SOMs to the same dataset, to assess whether or not this method is able to provide more information about the companies in our sample, and about the way in which their international projection is shaped. In particular, we examined both raw data and a sigmoid transformation of the dataset; we refer to those experiments by the labels EXP1 and EXP2, respectively. The rationale under this choice may be easily motivated. Looking at the frequency distribution of the three variables under examination one could observe that the values are too much concentrated either on the right or the left hand side of the histogram. By applying a sigmoid transformation we avoid this problem, thus obtaining a values distribution close to a uniform distribution.

Moving to the discussion of the results, in the case of raw data (EXP1) three clusters emerged, whose main statistics are reported in Table 3. SOM overall appearance is provided in Figure 1.

One can note that clusters C1 and C2 are practical equally representative, since they contain approximately 40% of the whole dataset. The third cluster C3 appears to be somewhat residual, and it accounts for the remaining 15% firms in the data sample. Additionally, it is noteworthy to observe that C1 shows both FASA and FESE closer to 54% value, whereas FSTS is nearer to 65%. This can be interpreted as the signal that firms in the cluster exhibit an average internationalization level; on the other hand, FSTS value indicates that those

Table 3. Clusters main statistics in the case of EXP1

Segment	Frequency	Assets (FATA)	Sales (FSTS)	Employment (FETE)
C1	43.00%	0.545	0.6549	0.5324
C2	42.00%	0.850	0.8316	0.7543
C3	15.00%	0.433	0.3844	0.3364

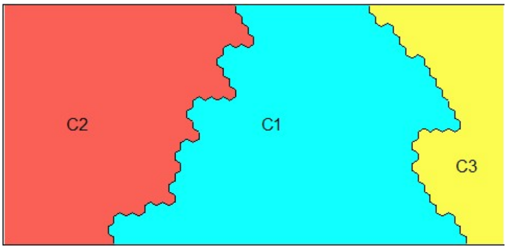


Fig. 1. Clusters in SOM working with raw data

firms have propensity to export higher than their attitude to delocalize capital–intensive activities such as production.

For what is concerning cluster C2, it seems to gather strongly internationalized firms, since all clusters members are firms with variables values definitely greater than 0.75. Besides, since the FETE value is lower than both FASA and FESE scores, this suggests that the cluster contains firms whose internationalization model is characterized by lower labour intensity per unit of foreign assets or sales. To conclude, C3 gathers together firms with relatively low internationalization intensity. In this case, values of the three variables are substantially aligned. However, one could not forget that we analyzed data referring to top 100 firms, as per foreign assets; this adds new light on cluster scores: the values mirror the evidence that cluster C3 hosts larger firms that although projected on the international scene have still maintained a prior role into the domestic reference market.

With respect to the results obtained by running K-means algorithm, SOM seems to provide a better solution, since it evidenced a more articulated and complex data structure. We then run EXP2, whose rationale has been explained in previous rows. We detailed basic simulation statistics in Table 4, while the final SOM is provided in Figure 2.

In this case the solution provided by SOM is, if possible, more explicative than that obtained in EXP1. We have now five different clusters. The first cluster includes 30 firms with higher values for all examined variables. Moreover, values of different variables are very similar one to each other. This means that we are in presence of strongly internationalized firms, a kind of *world champion* firms for which the original market weights lower than 20% of the whole activity. Some of these firms (for instance: Vodafone, Hutchison Whampoa, Perno-Ricard) are

Table 4. Clusters main statistics in the case of EXP2

Segment	Frequency	Assets	Sales	Employment
C1	30.00%	0.861	0.8490	0.8130
C2	19.00%	0.489	0.7523	0.4418
C3	15.00%	0.787	0.7764	0.6019
C4	22.00%	0.579	0.5496	0.6040
C5	14.00%	0.428	0.3821	0.3229

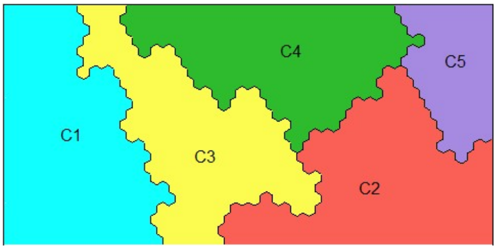


Fig. 2. Clusters obtained in a SOM working with a sigmoidal transformation on data

global champions from large economies, while others (in detail: Nestlè, Nokia, Philips) come from home countries which are far smaller than the optimal scale for their activities and, as previously stated, this boosts their international projection. Cluster C3 is spatially neighbor of C1 (Figure 3), and it seems reasonable, provided that variables values mirror C1 features at lower scale. However, looking at Table 4, it sticks immediately out that FETE scores are definitely lower than those of both FATA and FSTS (as already seen in C2 from EXP1). Like in that case we already discussed for EXP1, also here we assume it as a signal of firms whose internationalization strategy concerns activities with lower labor intensity. Indeed, many of the firms in this cluster belong to capital intensive industries such as oil and gas (Exxon Mobil, Total, to cite some), pharmaceuticals (GlaxoSmithKline, Roche), and aircraft. FETE values are practically aligned in both C3 and C4, but in this latter case, FETE is similar to both FATA and FSTS, on a leverage level; this suggests we are dealing with firms whose internationalization strategy is well balanced across the three strategic aspects. Many of them are utilities companies, such as EON, Iberdrola and GDF Suez. C5, on the other hand, is quite similar to C3 of EXP1: it has residual features and includes firms with the lowest internationalization profile. Remarkable is the presence of two major players (Wal–Mart and Tesco) in the retail sector, whose industrialization is not yet so intense as in other major industries, due to structural and normative issues. Finally, C2 joins together firms with high export intensity, like in the case of C2 in EXP1. In this latter case, however, the sample has been more refined, thus assuming a more sharp profile with respect to export propensity. Emblematic is the presence in this cluster of many car manufacturers (Renault, FIAT, Toyota, Nissan, BMW, Daimler), which share an approach where a truly world market is addressed with a very strong production-base in the home economy.

5 Conclusion

In this paper we applied Self Organizing Maps (SOMs) to study main structural features of the internationalization projection of the largest Multinational Corporations (MNCs) worldwide. By comparison with the results of K-Means clustering, SOM methodology has resulted to be much more performing, opening the way to further refinements of this approach to international business studies. An interesting research strand would include the extension of the analysis to a larger sample of firms; in particular, it would be of major significance to compare the structure of the United Nations Conference on Trade and Development (UNCTAD) top-100 firms with the corresponding sample of MNCs from developing economies, to assess whether or not *emerging* multinationals follow the same paths of international development than their correspondents from developed economies. A second research vein could concern the width of the datasets to be analyzed, since additional financial, structural, cultural and managerial indicators could be inserted in order to characterize the international activities of the firms; indeed, the use of a wider set of clustering variables could lead to a more refined and explicative definition of groups of firms sharing a similar approach to foreign markets.

References

1. Aggarwal, R., Berrill, J., Hutson, E., Kearney, C.: What is a multinational corporation? Classifying the degree of firm-level multinationality. Int. Bus. Rev. 20, 557–577 (2011)
2. Calinski, T., Harabasz, J.: A dendrite method for cluster analysis. Commu. Stat. 3(1), 1–27 (1974)
3. Dhillon, I.S., Modha, D.S.: A Data-Clustering Algorithm on Distributed Memory Multiprocessors. In: Proc. KDDWS on High Performance Data Mining (1999)
4. Dörrenbächer, C.: Measuring Corporate Internationalisation. A Review of Measurement Concepts and their Use. Interec. 35, 119–126 (2000)
5. Fisch, J.H., Oesterle, M.–J.: Exploring the Globalization of German MNCs with the Complex Spread and Diversity Measure. Schmalenbach Bus. Rev. 55, 2–21 (2003)
6. Fortanier, F., van Tulder, R.: Internationalization trajectories - a crosscountry comparison: are large Chinese and Indian companies different? UNU–MERIT Working Papers 2008–54, Maastricht (2008)
7. Geisler Asmussen, C., Pedersen, T., Petersen, B.: How Do We Capture Global Specialization When Measuring Firms' Degree of Globalization? Man. Int. Rev. 47(6), 791–813 (2007)
8. Geisler Asmussen, C.: Local, regional, or global? Quantifying MNE geographic scope. J. Int. Bus. St. 40, 1192–1205 (2009)
9. Kohonen, T.: Self–Organizing Maps. Springer, Berlin (2002)

10. Hassel, A., Höpner, M., Kurdelbusch, A., Rehder, B., Zugehör, R.: Two Dimensions of the Internationalization of Firms. J. Man. St. 40(3), 705–723 (2003)
11. Ietto–Gillies, G.: Different conceptual frameworks for the assessment of the degree of internationalization: an empirical analysis of various indices for the top 100 transnational corporations. Transnat. Corp. 7(1), 17–40 (1998)
12. McQueen, J.B.: Some methods of classification and analysis in multivariate observations. In: Proc. Fifth Barkley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
13. Majocchi, A., Zucchella, A.: Internationalization and Performance Findings from a Set of Italian SMEs. Int. Small Bus. J. 21(3), 249–268 (2003)
14. Nguyen, T.–H., Cosset, J.–C.: The measurement of the degree of foreign involvement. App. Ec. 27, 343–351 (1995)
15. Olsen, B., Elango, B.: Do Multinational Operations Influence Firm Value? Evidence from the Triad Regions. Int. J. Bus. Ec. 4(1), 11–29 (2005)
16. Pangarkar, N.: Internationalization and performance of small and medium–sized enterprises. J. World Bus. 43, 475–485 (2008)
17. Resta, M.: Early Warning Systems: an approach via Self Organizing Maps with applications to emergent markets. In: Proceedings of the 2009 Conference on New Directions in Neural Networks: 18th Italian Workshop on Neural Networks, WIRN 2008. IOS Press, Amsterdam (2009)
18. Resta, M.: Assessing the Efficiency of Health Care Providers: A SOM Perspective. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 30–39. Springer, Heidelberg (2011)
19. Sarlin, P., Eklund, T.: Fuzzy Clustering of the Self-Organizing Map: Some Applications on Financial Time Series. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 40–50. Springer, Heidelberg (2011)
20. Martín, B., Serrano Cinca, C.: Self Organizing Neural Networks for the Analysis and Representation of Data: some Financial Cases. Neur. Com. Appl. 1(2), 193–206 (1993)
21. Sullivan, D.: Measuring the Degree of Internationalization of a Firm. J. Int. Bus. St. 25(2), 325–342 (1994)
22. Wu, F.X.: Genetic weighted K-means algorithm for clustering large-scale gene expression data. BMC Bioinformatics 9 (2008)

Short Circuit Incipient Fault Detection and Supervision in a Three-Phase Induction Motor with a SOM-Based Algorithm

David Coelho[*] and Claudio Medeiros

Instituto Federal de Educação Ciência e Tecnologia do Ceará, Fortaleza, Ceará, Brasil
Davidcoelho89@gmail.com, claudiosa@ifce.edu.br

Abstract. An offline two-dimensional SOM neural based algorithm is used in order to supervise a three-phase squirrel-cage induction motor and detect short-circuit incipient fault condition. A sinusoidal PWM inverter is used to feed the motor and some components of the current motor frequency spectrum are used as input variables. A special electrical structure was built to emulate incipient short-circuit at the stator windings of the induction motor. The data were acquired with the motor operating under different frequencies, load level and fault extent. Through the generated data base, the algorithm was tested and a high mean success rate combined with a good visualization of the problem was achieved. In near future, this algorithm can be used as base for an online supervisory system for this kind of motor failure.

Keywords: Three-Phase Induction Motor, Fault Detection, Self-Organizing Map (SOM), Short-Circuit.

1 Introduction

Three-phase induction motors are widely used in industry due to their robustness, efficiency and simplicity [1]. Fans, blowers, conveyors, crushers, compressors, cranes and pumps are some examples of these machines applications [2].

In order to match all the applications of three-phase induction motors, many studies in speed and torque control have been made [3], [4]. Among these, inverter drives are widely used because they reduce maintenance and improve reliability [3].

However, due to machine aging and environment conditions, the induction motor is subject to various faults [1]. Most of these failures are caused by combination of various stresses acting upon the winding, rotor, bearings and shaft [5]. Among these faults, the insulation breakdown in the stator winding corresponds to nearly 40% of the total motor failures [6] and, in general, initiates as a high impedance fault [7]. Then, the fault current can cause a local heating, making the failure spread quickly in

[*] Corresponding author.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 315–323.
DOI: 10.1007/978-3-642-35230-0_32

the winding [8]. If this fault is detected at the beginning of its occurrence, maintenance team can actuate and save production costs or the induction machine can be reused after the motor rewinding [9].

A lot of studies in fault detection have been made. As examples one can mention applications using supervised [1] [10] and unsupervised [6] [11] artificial neural networks (ANN). An important characteristic of fault detectors using ANNs is that they do not suppose the existence of any motor mathematical model [6], but it is necessary a consistent and significant amount of data that can represents properly the problem in hands.

Another important characteristic of ANNs is the ability to treat nonlinearity and ambiguity in the input space. Particularly, in inter-turn short-circuit detection, it is common to use some components of the current frequency spectrum to compose the input vectors. But these frequency components can exist previously in the electric system or be affected by more than one kind of fault. These conditions can create severe difficulties for fault detection. Furthermore, the current signals may be embedded in strong background noise [1].

In this research the authors use a Self-Organizing Map (SOM) [12] for its ability to perform clustering and preserve the topology [11]. The purpose is not only classify an inter-turn short-circuit fault but also to show the failure evolution in the output space.

2 Test Bench and Data Acquisition

To emulate inter-turn short-circuit faults, a three phase induction motor was rewound, and a mechanical and an electrical structure were built. Each part of this system is described as follows.

2.1 Rewound Motor and Emulation System

A standard three-phase delta connected squirrel-cage induction motor has been used as base. Its main characteristics are 0.75 kW, 220/380 V, 3.02/1.75 A, 79.5% efficiency, 1720 rpm, Ip/In = 7.2, and 0.82 power factor. There are 348 turns per phase distributed in two groups of three concentric bobbins with fifty-eight turns each one. Originally, only two terminals are available per phase.

The motor has been rewound and, after that, eight extra terminals per phase are available, exposing derivations of the first concentric bobbin in the first group. So, it is possible emulate many inter-turn short-circuit levels. For this research, three different levels of short-circuit were used. In the lowest level, 5 turns were short-circuited, totaling 1.41% of the turns of one phase. In the medium level, 17 turns (4.8%) were short-circuited. Finally, in the highest level, 32 turns (9.26%) were short-circuited.

Besides this, an auxiliary command system was built to execute two kinds of short-circuits schemes. The first one, which imitates the initial condition of the short-circuit process, is achieved connecting a great parallel resistor to specific derivation terminals. So, a little part of the phase current flows through the resistor, which characterize a high impedance short-circuit scheme. The other one, called low impedance short- circuit scheme, is illustrated in Figure 1. At this picture, it can be seen an extra bobbin formed due the short-circuit effect. It is important to note that there is a resistor limiting the fault current in order to preserve the motor integrity.

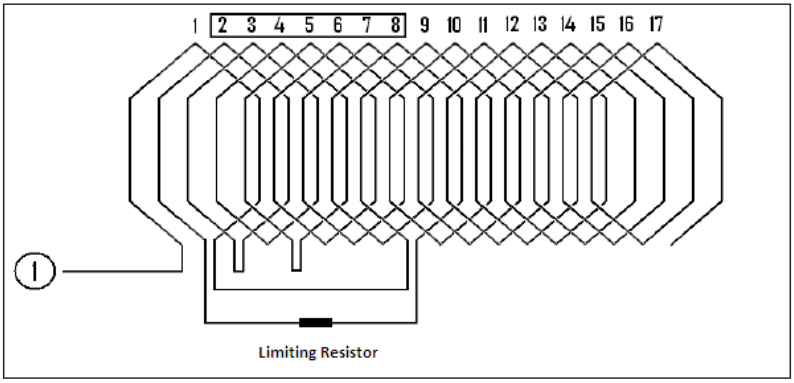


Fig. 1. Emulation of seven inter-turn short-circuit

2.2 Mechanical Structure

The mechanical load applied to the motor is based on Eddy current brakes [13]. So, two coils, performing eighteen thousand turns, and an iron magnetic circuit are used to produce a magnetic flux through an aluminum disc coupled to the motor's shaft. A controlled single-phase rectifier is used to vary the load level to the motor under operational conditions.

Several sort of mechanical load profiles can be emulated by this system, but only constant load related to the rotation speed is used to generate the data base.

Special attention was given to the structure alignment, since excessive vibration can introduce undesirable frequency components in the acquired signals.

In figure 2 the mechanical structure built for this research is shown.

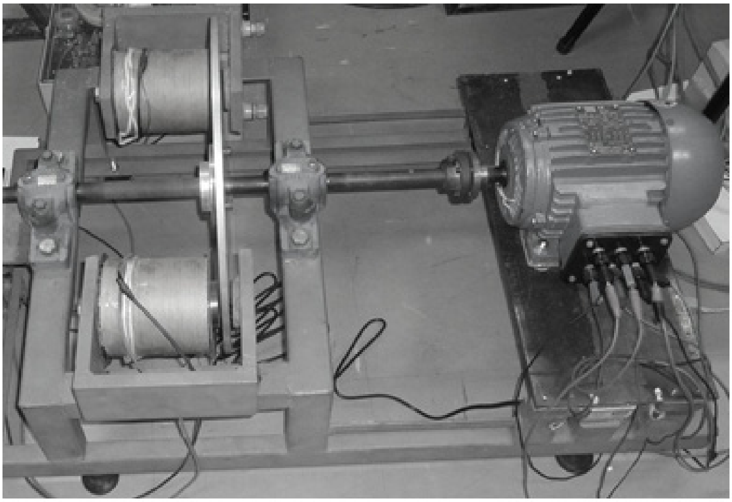


Fig. 2. Mechanical Structure for applying load to the motor

2.3 Hall Effect Sensor

As the frequency-domain analysis appears to be the most popular computational approach for fault detection [9], this feature extraction methodology is adopted.

One Hall's effect sensor is connected to each line cable between frequency inverter and delta connected motor. The sensor input range is -50 A to +50 A, and its respective output varies between 0 V and 5 V. As the motor rate current is 3.05 A, it is necessary to utilize a conditioning electronic circuit. The sensor output signal is initially amplified and passes through a second order low-pass Butterworth filter before to be applied to data acquisition module. The filter cutoff frequency is 1 kHz.

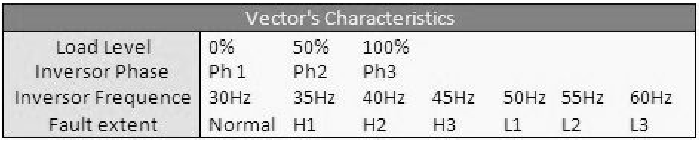
Three channels of Agilent's "U2352A" data acquisition module are used to acquire the line current signals. The 16-bit resolution analog input channels are configured to differential mode acquisition in a -5 V to +5 V range. The frequency sample rate is 10 kHz.

2.4 Data Base

The data acquired consists of 441 time domain vectors: 63 represent normal conditions, 189 represent high impedance faults and 189 represent low impedance faults. Each vector is the result of 10 seconds of acquisition. So, 100000 samples are available. After applying fast Fourier transformation (FFT) to these vectors, only seven components of each resultant frequency spectrum are used to compose 441 characteristics vectors.

As can be seen in Table 1, four status information are added to each vector: load level, phase identification, frequency of the voltage applied to the motor, and the "fault extent". Letters "H" and "L" represent, respectively, high impedance and low impedance short-circuits. The number next to them represents the level of the fault: 1 for 5 turns short-circuited, 2 for 17 turns short-circuited, and 3 for 32 turns short-circuited. These additional information are relevant to evaluate the final fault detector in two aspects: classification rate and ability to show the evolution of the failure. It is important to remember that the failure starts from a high impedance condition, and goes to a low impedance condition.

Table 1. Vector´s Characteristics



Vector's Characteristics							
Load Level	0%	50%	100%				
Inversor Phase	Ph 1	Ph2	Ph3				
Inversor Frequence	30Hz	35Hz	40Hz	45Hz	50Hz	55Hz	60Hz
Fault extent	Normal	H1	H2	H3	L1	L2	L3

3 Methodology

Firstly, the FFT is used to transform the time-domain vectors into frequency-domain vectors. As the sampling rate is 10 kHz, the range of the frequency spectrum is from 0 to 5 kHz. In order to reduce the amount of characteristics in the input vectors, the authors have investigated for salient multiple components of the fundamental frequency (f) of the inverter output voltage. Considering all frequency spectrums obtained as a result of the application of specific fundamental frequency of the inverter output voltage, independently of the load level and fault extent, the authors have calculated the variances for all frequency components. Then, these variances were evaluated and it was verified that the most significant frequencies, by the power classification point of view, are the following components: 0.5f, f, 1.5f, 2f, 3f, 5f and 7f. So, the feature vectors are composed of 7 features and 4 status information: load level, inverter phase, frequency of the output inverter voltage, and fault extent.

In Figure 3 two bi-dimensional projections of the input vectors are shown. For both, the horizontal axis represents the 3f feature and vertical axis represents the 5f feature, and "triangles" represent projections related to no fault conditions. In Figure 3(a) the black dots represents high-impedance short-circuit fault conditions. It can be seen that these two classes seems to be difficult to distinguish. On the other hand, in Figure 3(b) black dots represents low-impedance short-circuit fault conditions and it can be seen that the data distribution of these two classes suggest that the classification problem is not complex.

These projections suggest what one can imagine: data representing high-impedance faults (incipient faults) are closer to the faultless data than data from representing low-impedance faults.

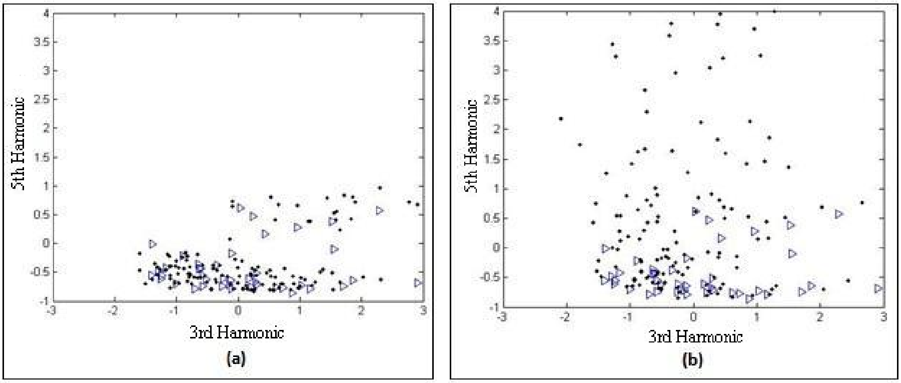


Fig. 3. 2D data projections (a) faultless and high-impedance fault; (b) faultless and low-impedance fault

The data base is normalized twice. Primarily, the frequency components are normalized by the magnitude of their fundamental frequency. This is necessary to avoid the influence of the load level. Finally, each feature is normalized by its global mean and standard deviation.

Thereafter, it is necessary to set the dimensions of the neuron matrix. In order to make this choice, the data base was separated into three basic sets: data representing normal operational conditions, data representing high impedance inter-turn short-circuit fault conditions and data representing low impedance inter-turn short-circuit fault conditions.

For each one of these sets, the authors have trained an unsupervised neural network using “Winner Takes-all” algorithm. The number of neurons was varied from 1 to 12, and the quantization error was evaluated. The best quantization error was obtained with 10 neurons for each previous set, performing 30 neurons, and so, a two-dimensional 6x5 SOM neural network topology was chosen.

Finally, with the normalized data base and with the chosen topology, the algorithm was defined. The general equation utilized to update the neurons is as follows [14]:

$$w_i(t+1) = w_i(t) + \eta(t)h(i^*, i, t)[x(t) - w_i(t)], \quad (1)$$

in which $w_i(t+1)$ is the new neuron weight, $\eta(t)$ is the learning rate and $h(i^*, i, t)$ is the neighborhood function. The learning rate varies according to

$$\eta(t) = n_0\left(1 - \frac{t}{t_{max}}\right), \quad (2)$$

in which $0 < n_0 < 1$, t is the current iteration and t_{max} is the total number of iterations.

Primarily, the neighborhood function varied according to:

$$h(i^*, i, t) = \exp\left(-\frac{\|r_i(t) - r_{i*}(t)\|^2}{2\propto^2(t)}\right), \quad (3)$$

In which i^* is the current winner neuron, $\|r_i(t) - r_{i*}(t)\|$ is the squared Euclidian distance between the current neuron and the current winner neuron, and $\propto^2$ determines the influence of the winner neuron over the others. The results obtained by the application of this methodology were not considered satisfactory. So, two other kinds of neighborhood functions are used during the training. The first one is utilized in the first eight epochs. In these epochs a decreasing rectangular neighborhood is used. The initial number of neighbors is 4, and after every 2 epochs, the number of neighbors is reduced by one. At these first epochs, $h(i^*, i, t)$ is “1” if the current neuron is a neighbor of the winner neuron and $h(i^*, i, t)$ is “0” if the current neuron is not a neighbor of the winner neuron. Then, from the ninth epoch until the last, the number of neighbors is 1 and the Gaussian Function, represented in equation 3, is utilized.

In addition, the function used to decide which neuron is the winner, is the Euclidian Function:

$$\|x(t) - w_i(t)\| = \sqrt{[x(t) - w_i(t)]^T[x(t) - w_i(t)]} = \sqrt{\sum_{j=1}^{n}[x_j(t) - w_{ij}(t)]^2} \quad (4)$$

In equations 1 and 4, it is shown the two processes of SOM ANN [17]: adaptation of neurons at equation 1 and competition at equation 4.

Beyond that, initially, the number of epochs used was 100 (chosen randomly). But, it was noticed that the mean quantization error converges in less than 20 epochs. So, this number of epochs is utilized at this work.

Moreover, after each training process, the SOM ANN is tested with the data that are not utilized for training. The percentage of data utilized for training process and test is, respectively, 75% and 25%.

Considering the training process finished, the follow procedure is used to label the neurons:

- A 3D matrix (6x5x3) is created only with "zeros". The first two dimensions represent the position of each neuron, and the third dimension represents the votes for each class.
- The training feature vectors are presented, once more, to the trained neural network, and each time that a neuron is considered "the winner" for an individual feature vector from a specific class, a vote is "added" for the position of the matrix that represented this class and this neuron.
- After all input vectors have been presented, each neuron is labeled as a representative for the class that it has more votes for.
- If there is a draw between the votes for two classes, the Euclidian distance between this neuron and the feature vectors of each class is calculated. The neuron is labeled as a representative of the class which has the lowest Euclidian distance.

This technique is used to avoid neutral neurons, that is, neurons that have the same number of votes for more than one class and could not be labeled.

4 Results

To evaluate SOM's ability to perform clustering and, simultaneously, classify the problem properly, 10 trainings and tests were performed. Numerical results are shown in Table 2, in which *Nn* is the number of neurons which compose the neural architecture used, *Nc* is the number of classes related to the problem, *CRtrain* (average) and *CRtest (average)* stand for the average classification rate for the training and testing data, respectively, *CRtrain* (Max) and *CRtest (Max)* stand for the maximum classification rate for training and testing data respectively. Only data from two phases of the motor were used to these trainings and tests totalizing 42 data from normal condition motor, 126 data from high impedance short circuit fault and 126 data from low impedance short circuit fault.

So as to compare SOM's ability to classify the problem, trainings and tests are made up with a 7-9-1MLP, that is, it contains nine neurons in the single hidden layer and one neuron in the output layer.

Table 2. Classification results

Algorithm	Nn	Nc	*CRtrain (average)*	*CRtest (average)*	*CRtrain (Max)*	*CRtest (Max)*
MLP	9	2	70.74	70.58	79.10	85.29
SOM	30	2	87.86	86.79	88.63	90.54
SOM	30	3	65.25	56.18	65.4	66.2

Two experiments are performed. In the first one 3 classes are used: normal condition, high impedance short-circuit incipient fault and low impedance short-circuit incipient fault. There is a good result in clustering, but the average classification rate of 65.25% in training data set is not considered satisfactory. Just 30.37% of normal condition data are recognized. Moreover, 81.31% of high impedance fault data are recognized and just 55.18% of low impedance fault data are recognized. That happens mainly due to the existing ambiguity between normal condition data and high impedance data. In the second experiment, when just the classes from normal condition and fault condition (embodying high and low impedance fault) are used, the average classification rate rises to 87.86% in the training and presents an average classification rate of 86.79% in the test. The recognition rate of normal condition data remains low (30.37%). So, one can notice that an ambiguity between low and high impedance fault data also exists seeing that there is a classification rate increase. A 90.54% maximum classification rate in testing data is the best result achieved.

To compare the results of the proposed methodology, a Multi-Layer Perceptron (MLP) with one hidden layer containing 9 neurons is used. The average classification rate in test is 70.58% and the maximum classification rate in test is 85.29%. The results with the MLP are not considered satisfactory, confirming that this problem is not easy to solve.

Figure 4 shows a 6x5 two dimensional SOM Neural Network that achieved 90.54% classification rate. The full symbols are neurons and the empty symbols are the input data. The squares represent high impedance fault data, the triangles represent low impedance fault data, and the circles represent normal condition data. It can be seen that the SOM can cluster each class in different regions of the map. The mapping also reflects the percentage of the used data, for it attributes the lower number of neurons to the lower data set.

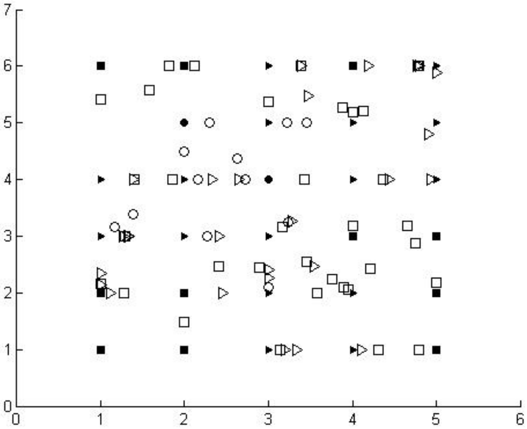


Fig. 4. SOM Neural Network

5 Conclusion

This paper shows that a two-dimensional SOM-Based algorithm is capable of achieving a classification rate of 88.63% and 90.94% in the training data set and test data set respectively in classification between normal condition and short-circuit faults. The preliminary results, when one distinguishes incipient faults (high impedance) from severe faults (low impedance), are not considered satisfactory yet. Also, with the visual analysis of SOM, it can be noticed that this ANN can cluster the input data into regions of the map. In near future, with the improvement of this initial offline ANN, an online supervisory of short-circuit faults can be built and the classification rate can be improved.

Acknowledgments. The authors thank FUNCAP for its financial support.

References

1. Ghate, V.N., Dudul, S.V.: Optimal MLP neural network classifier for fault detection of three phase induction motor. Expert Systems With Applications 37 (2010)
2. Vico, J., Hunt, R.: Projection Principle for Electrical Motors in the Cement Industry. In: 2010 IEEE-IAS/PCA 52nd Cement Industry Technical Conference (2010)
3. Sawa, T., Kume, T.: Motor Drive Technology – History and Visions for the Future. In: Annual IEEE Power Electronics Specialists Conference, vol. 35, Auchen, Alemanha (2004)
4. Nirali, R., Shah, S.K.: Fuzzy Decision Based Soft Multi Agent Controller for Speed Control of Three Phase Induction Motor. International Journal on Soft Computing (IJSC) 2(3) (2011)
5. Bonnet, A. H.: Root Cause Failure Analysis for AC Induction Motors in the Petroleum and Chemical Industry. In: Petroleum and Chemical Industry Conference (PCIC) (2010)
6. Martins, J.F., Pires, V.F., Pires, A.J.: Unsupervised Neural-Network-Based Algorithm for an On-Line Diagnosis of Three-Phase Induction Motor Stator Fault. IEEE Transactions on Industrial Electronics (2007)
7. Nataranjan, R.: Failure identification of induction motors by sensing unbalanced stator currents. IEEE Transactions on Energy Conversion 4(4), 585–590 (1989)
8. Tallam, R.M., et al.: A survey of methods for detection of stator related faults in induction machines. In: Proceedings of the IEEE International Symposium on Diagnostics for Electric Machines, Power Electronics and Drives (SDEMPED 2003), pp. 35–46 (2003)
9. Thomson, W.T.: On-line mcsa to diagnose shorted turns in low voltage stator windings of 3-phase induction motors prior to failure. In: Proceedings of the IEEE International Conference on Electric Machines and Drives (IEMDC 2001), pp. 891–898 (2001)
10. Asfani, D.A., et al.: Temporary short circuit detection in induction motor winding using combination of wavelet transform and neural network. Expert Systems with Applications 39 (2012)
11. Wu, S., Chow, T.W.S.: Induction machine fault detection using SOM-based RBF neural networks. IEEE Transactions on Industrial Electronics 51(1) (February 2004)
12. Kohonen, T.: Self-Organizing Map. In: Proceedings of the IEEE 78(9) (1990)
13. Maurya, V.K., et al.: Eddy Current Braking Embbeded Circuit. International Journal of Applied Engineering and Technology 1(1), 104–113 (2011) ISSN: 2277-212X, `http://www.cibtech.org/jet.htm`
14. Haykin, S.: Redes Neurais: Princípios e Práticas. Ed. Bookman, Porto Alegre (2001)

The Finnish Car Rejection Reasons Shown in an Interactive SOM Visualization Tool

Jaakko Talonen[1], Mika Sulkava[2], and Miki Sirola[1]

[1] Aalto University, Department of Information and Computer Science, Espoo, Finland
{jaakko.talonen,miki.sirola}@aalto.fi
[2] MTT Agrifood Research Finland, Economic Research, Helsinki, Finland
mika.sulkava@mtt.fi

Abstract. In this paper a new SOM visualization tool is introduced. It is shown how Collaborative Filtering can be used as preprocessing before the SOM training. Our goal was to provide for the user a possibility to analyze car differences by component planes which is not possible by original published tables. In addition it is possible to explore how different flaws are related in time or with other variables. The effects of the driver dependent components, such as tires, can be filtered out from rejection probability using the component plane codebooks. The interactive SOM visualization is very useful when a large number of labels is present. We developed a function to generate the needed files for a Processing language based tool. Our tool can be used simultaneously with the SOM Toolbox.

Keywords: SOM Visualization Tool, Collaborative Filtering, Car Inspection Data, SOM Toolbox.

1 Introduction

In this paper multidimensional data is used to draw conclusions on the structure of car inspection data. The Self-Organizing Map (SOM) preserves the data topology of multidimensional data and the given data can be explored visually in a lower dimension. Collaborative filtering (CF) was initially proposed to find preferences for users [1]. In this paper, we show that combining CF and SOM provided us new valuable information. Data which contain missing entries can be explored more efficiently.

Our goal is to visualize filtered data on a low dimension to preserve as much information as needed. During our experiments, we couldn't find a proper tool for our visualization problem, although there are several SOM software packages [2]. We developed a simple add-in for the SOM Toolbox [3], which is flexible, general-purpose software library created by the SOM Programming Team of Helsinki University of Technology [4]. Exploring large maps *(200<map size<10 000)* with many labels is made efficient with the interactive SOM component visualization tool. Attention is paid to interactive label selection where a user can define by several ways which labels are shown.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 325–334.
DOI: 10.1007/978-3-642-35230-0_33

2 Data

A-Katsastus is the largest private provider of vehicle inspections in northern Europe. In 2011 this company published Finnish rejection statistics in the same format for the third time [5]. The statistics are published in dozens of tables on the basis of the year of introduction into use, make and model. In this paper the word "car" means this combination, e.g. MAZDA-6 (2007). Last year they inspected almost one million passenger cars in Finland [1]. As even the size of the raw data set is very large, only the aggregated data were published.

The average rejection rate r [percentage] and thousand kilometers driven [10^3km] were published, if a certain car is inspected more than *100* times. If some car is inspected somewhat more than the limit, it can cause missing values in another year. In addition, the rejection reasons (RR) were listed, when the same RR was listed more than *10* times. Only a maximum of three of the most common reasons per a car was listed. In theory probability for certain RR is $p \in]0,1]$.

In Finland, new cars are inspected on third and fifth year and older cars yearly. This causes missing values to our data set. Newer cars have fewer rejections causing zero values to our data set. Therefore there is less information about new cars than old ones.

In the original publication, data rejections are divided into *13* different classes, such as chassis, brakes, steering and control devices. RR data is quantified for the analysis using the rejection reasons. We define a car matrix as

$$C = \begin{bmatrix} c_{1,1} & \cdots & c_{1,n_c} \\ \vdots & \ddots & \\ c_{n_{RR},1} & & c_{n_{RR},n_c} \end{bmatrix}, \tag{1}$$

where the size of matrix **C** is the number of RR times the number of different cars inspected in the years *2009-2011*. A cell value in matrix **C** is one if certain RR is listed and zero otherwise. A similar matrix **M** is defined to represent missing values. $m(i,j)=0$ when car information is missing, otherwise *1*. Average kilometers driven d, rejection rate r and car age a are scraped to data vectors from the published A-Katsastus documents. More about the input matrix setup is discussed in Section 5.

3 Methods

In this Section it is shown how to use Collaborative Filtering (CF) as a preprocessing method before training a SOM. We got intuition for this from the concept of a recommender system, which is an important application of machine learning. There are many websites or systems that try to recommend new products for user. Examples include Amazon (recommends new books), Netflix (try to recommend new movies to user) and eBay (shopping website). [1]

In our research, the recommender system is used for missing value imputation and filtering the given data. With a suitable regularization parameter selection, we were able to filter both zero and one values more reliably by “collaborating” with the RR information of other cars. This method is explained carefully by using our car inspection data to ensure that it is understood correctly.

3.1 Collaborative Filtering

Our task is to predict the probability of rejection reason (RR) *i* for each car *j*. In this section, a content based recommendation system is introduced. First, let us define matrix **Y** and **R** as

$$Y^* = \begin{bmatrix} C_{2011} \\ C_{2010} \\ C_{2009} \end{bmatrix}, Y = Y^* - \mu, R = \begin{bmatrix} M_{2011} \\ M_{2010} \\ M_{2009} \end{bmatrix}, \tag{2}$$

where the size of **Y** and **R** matrices is *((#RR * #years published) x #cars)* and μ is the mean vector of **Y*** excluding missing values. Each RR in this new matrix **Y** has an average value of *0*.

In practice each car “rates” only some set of the RRs whenever *r(i, j) = 1*. If *RR(i)* was mentioned for car *j* in the published data table, *y(i, j) = 1*, otherwise *0*. The task of the recommender system is to fill in the missing values of the RR data, denoted by "?" in Table 1.

Table 1. Data is published in dozens of tables. However, car inspection data can be shown in one large table where rejection reasons and year published are shown as rows and all cars, with more than 100 inspections in at least one year as columns.

Rejection Reason (RR)	car 1	car i	...	car N
2011: Chassis	1	1		?
2010: Chassis	1	0		?
...				
2009: Identification number	0	?		0

So, both **Y** and **R** matrices contain only zero and one values. Before the SOM visualization, missing and zero values are filtered to ensure that RR dependencies are visualized effectively. Without filtering, a large number of car labels are situated in the same nodes in the SOM.

A set of features *x* is defined for each RR and a parameter vector *θ* for each car. This is basically a linear regression problem, so the predicted values here are as close as possible to the values that we observed in our data set. Our optimization objective for learning the parameters of car *j* is

$$J_x = \min_{\theta^{(1)},\ldots,\theta^{(n_c)}} \frac{1}{2}\sum_{j=1}^{n_c} \sum_{i:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_c}\sum_{k=1}^{n}(\theta_k^{(j)})^2, \tag{3}$$

where *n* is the number of features. This is used to make predictions for all of the cars. So *J* in Eq. (3) is an optimization objective which is minimized by a gradient descent update equation as

$$\theta_k^{(j)} := \theta_k^{(j)} - \alpha \frac{\partial}{\partial \theta_k^{(j)}} J(\theta^{(1)},\ldots,\theta^{(n_c)}). \tag{4}$$

The algorithm has a very interesting property – feature learning. It can start to learn by itself what features to use. Similarly to Eq. (3), we can learn the features of RR *i* with objective

$$J_\theta = \min_{x^{(1)},\ldots,x^{(n_{RR})}} \frac{1}{2}\sum_{i=1}^{n_{RR}} \sum_{j:r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_{RR}}\sum_{k=1}^{n}(x_k^{(i)})^2. \tag{5}$$

In initialization, we randomly set some values for the car parameters. Now based on the initial random guess for the *θ*, we learn features for the different RRs. We can keep iterating by going back and forth and optimizing both parameter sets. This will actually produce reasonable set of features for RRs for each car. This is a basic collaborative filtering algorithm. The term collaborative filtering refers to the observation that when the algorithm is performed with a large set of data, all these cars are effective by some sort of collaboration to get better RR estimates.

It is more efficient to find the optimal solution for *θ* and *x* simultaneously. The new optimization objective *J* is defined as

$$\begin{aligned} J = \min_{x^{(1)},\ldots,x^{(n_{RR})},\theta^{(1)},\ldots,\theta^{(n_c)}} & \frac{1}{2} \sum_{(i,j):r(i,j)=1} \left((\theta^{(j)})^T x^{(i)} - y^{(i,j)}\right)^2 \\ & + \frac{\lambda}{2}\sum_{i=1}^{n_{RR}}\sum_{k=1}^{n}(x_k^{(i)})^2 + \frac{\lambda}{2}\sum_{j=1}^{n_c}\sum_{k=1}^{n}(\theta_k^{(j)})^2, \quad x,\theta \in R^n, \end{aligned} \tag{6}$$

where the first term is a sum over every pair of car and RR with the condition that the data are published, i.e. *r(i,j) = 1*. The second and the third terms are regularization terms, see Eqs. (3, 5).

The first step in CF is the initialization of *x* and *θ* to small random values like usually done in neural network training. The optimization objective *J* in Eq. (6) is minimized by using the partial derivatives of the cost function defined as

$$\begin{aligned} x_k^{(i)} &:= x_k^{(i)} - \alpha \frac{\partial}{\partial x_k^{(i)}} J(x^{(1)},\ldots,x^{(n_{RR})},\theta^{(1)},\ldots,\theta^{(n_c)}), \\ \theta_k^{(j)} &:= \theta_k^{(j)} - \alpha \frac{\partial}{\partial \theta_k^{(j)}} J(x^{(1)},\ldots,x^{(n_{RR})},\theta^{(1)},\ldots,\theta^{(n_c)}). \end{aligned} \tag{7}$$

J is minimized by gradient descent. Finally, given a car, if a car has parameters *θ*, and if there is a RR with learned features *x*, we would then predict that RR would be given a probability of $\theta^T x$. The CF algorithm learns simultaneously features for all the RRs as well as parameters for all the cars. The predictions for those reasons how different cars would fail in any car inspection year are achieved.

The predicted ratings, the optimized values of CF, are used as a part of input matrix **P** as

$$P = \hat{Y} = \begin{bmatrix} (\theta^{(1)})^T(x^{(1)}) & (\theta^{(2)})^T(x^{(1)}) & \cdots & (\theta^{(n_c)})^T(x^{(1)}) \\ (\theta^{(1)})^T(x^{(2)}) & (\theta^{(2)})^T(x^{(2)}) & \cdots & (\theta^{(n_c)})^T(x^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ (\theta^{(1)})^T(x^{(n_{RR})}) & (\theta^{(2)})^T(x^{(n_{RR})}) & \cdots & (\theta^{(n_c)})^T(x^{(n_{RR})}) \end{bmatrix} + \mu, \tag{8}$$

where we have estimated probabilities for each car and RR pair. For example, RR probabilities for a new car based on our recommender system is *p(new car) = μ*, because the first term plays no role at all in Eq. (6). Besides the matrix **P**, additional features are derived in Section 5.

3.2 Self-Organizing Map

High dimensional data can be visualized in low dimensional views by using the Self-Organizing Maps (SOM). The method has two main phases, training and mapping, like in most artificial neural networks. The SOM consists of neurons which are usually initialized with small random values. In the iterative training phase, one sample vector *x* from the input data set is chosen randomly. The distance measurement between it and all the weight vectors of the SOM is derived. The weight vectors are updated so that the closest neuron *c* weight vector with input *x* (BMU) is moved closer to the input vector in the input space. The topological neighbors of the BMU are moved in a similar way weighted by the neighborhood function. This update process is smoothing the codebook values where the new model values are derived as

$$m_i(t+1) = m_i(t) + \alpha(t)h_{c,i}(t)\left[x(t) - m_i(t)\right], \tag{9}$$

where *α(t)* is a scalar factor defining the size of model *i* correction. The smoothing kernel *h(t)* takes care of neighbor updating. It decreases when the distance between the models increases, with the maximum value of *1* when *c = i*. [4, 6]

4 Visualization Tool

A lot of SOM visualization tools are available, e.g. Viscovery SOMine, proSOM, SomVis [2, 7]. During our experiments an add-in for an existing tool was designed. We developed a function which is used with Matlab SOM Toolbox [3]. Our tool was implemented using the Processing language version 1.5.1. For recent developing, we had used version 2.06a to enable the use of our program on the Android platform. Therefore, some of the functionalities such as zooming and component plane selection are based on the mouse/touch positions. The Processing visualization tool is shown in Fig. 1. This tool is mainly useful in cases where the data consist of a large amount of label information or dozens of component planes.

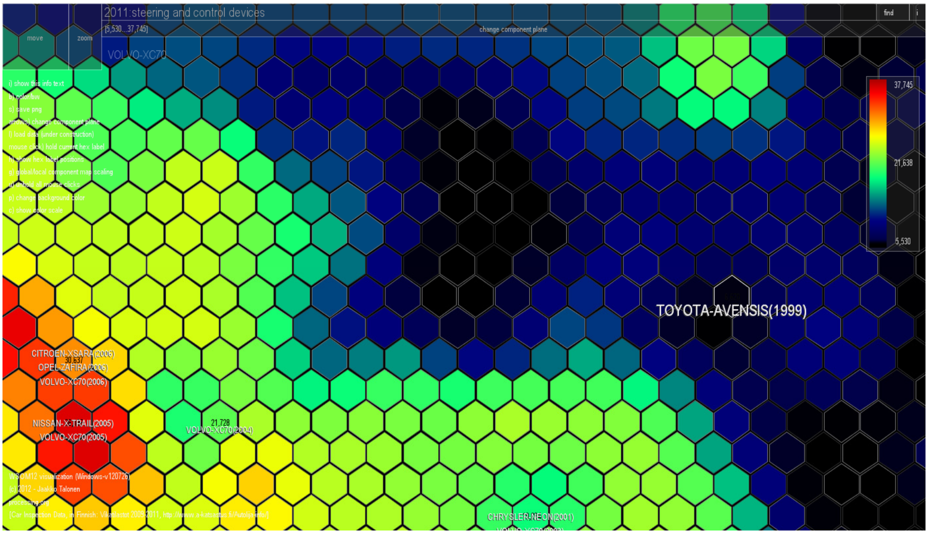

2011:steering and control devices
[5,530...37,745]
move
zoom
VOLVO-XC70
change component plane
find
i
i) show this info text
b) color/bw
s) save png
arrows) change component plane
l) load data (under construction)
mouse click) hold current hex label
h) show hex label positions
g) global/local component map scaling
u) unhold all mouse clicks
p) change background color
c) show color scale
37,745
21,638
5,530
TOYOTA-AVENSIS(1999)
CITROEN-XSARA(2006)
30,637
OPEL-ZAFIRA(2006)
VOLVO-XC70(2006)
NISSAN-X-TRAIL(2005)
VOLVO-XC70(2005)
21,728
VOLVO-XC70(2004)
CHRYSLER-NEON(2001)
WSOM12 visualization (Windows-v120726)
(c) 2012 - Jaakko Talonen
processing.org
[Car Inspection Data, in Finnish: Vikatilastot 2009-2011, http://www.a-katsastus.fi/Autolija-info/]

Fig. 1. Navigation on the component planes are made possible by the areas at the top left corner. Long top area is used for the feature selection. Labels can be selected manually from the map or with the find command. Also some additional features are programmed to the tool. [8]

The tool can be easily taken into use. There is no need to install software, and only with two lines of Matlab code (bolded) the interactive visualization tool can be used:

```
sM = som_make(sD,'munits', 1060,'mask',[ones(1,42) 0]');
sM = som_autolabel(sM,sD,'add1'); % all labels are stored
som_createprocessingdata(sM); % I: (som_make - output)
!WSOMprocessing; % opens an interactive tool
```

The first bolded line calls function and generates txt-data files. Information of codebooks, labels and component planes are stored into these files. Data can be modified with any text editor. The last line opens our tool [8]. Visualizations can be explored even Matlab or Octave is not installed. Car inspection data are included in the tool, but data can be overwritten and other data sets can be explored. A hexagonal grid of nodes is preferable for visual inspection, so the rectangle shape is left out from our tool [4].

5 Experiments and Results

All available data tables were combined to one data matrix **Y**. This procedure caused missing values, so we defined matrix **R**, see Eq. (2). The sizes of both matrices are *39 x 1060*.

In the Collaborative Filtering method some parameters were selected based on our experiments. Parameter selection was based mainly on intuition and on the quality of SOM visualizations. For example, with two features *n* SOM visualization was very simplified. In practice, RR *i* either correlated with a car *j* or not. Finally, we decided to have 39 features *n*, so the final feature vector dimension θ in our experiments was the same as the number of RRs (13) times years of published data (3). Optimized RR feature matrix size is then of size *39x39*, which was used to derive matrix **P**, see Eq. (8).

A regularization term λ helps to prevent overfitting, see Eq. (6). If λ is large, then predicted values have high bias and matrix **P** underfits this data set. If we have a very small value of λ, say $\lambda = 0$, it is usually an overfitting setting. With some intermediate value of λ, a reasonable fit to this data is achieved. In our experiments, we wanted to avoid overfitting by selecting $\lambda=50$. With a rather large λ, more reliable estimates for 4th to 13th rejection reason for each car *j* were reached. With small λ or without it, filtered $p(i,j)$ values were close to zero for $r(i,j) = 1$.

The output of CF matrix **P** was then visualized using SOM, see Eq. (8). Without additional features many cars (1060) had approximately same properties (39) and no differences were detected between those cars.

In this paper, the SOM input matrix is a combination of CF estimates and the original and derived numerical data. It was ensured that matrix **P** has positive values and then **P** was transposed and scaled into percentages by setting the sum of the RRs of each year (*2009, 2010, 2011*) to *100.* Matrix **S** is defined as estimated probabilities of each RR *i* and car *j* pair. For example, the 10th most common RR in 2010 for each car is estimated. The SOM input matrix is defined and input matrix units are shown in square brackets as

$$F_{SOMinput} = \left[S[\%] \quad d\left[10^4 km\right] \quad r[\%] \quad a[years] \quad \frac{r[\%]}{d\left[10^5 km\right]} \right], \tag{10}$$

where matrix **S** is the RR probability matrix, *d* is an average vector of *ten thousands* kilometers driven, *r* is the rejection rate, a is the car age (2011 – "introduced to use") and in the last column of matrix **F** is a derived mask variable, which is not used for training.

The scaling of the vector components was based on Kohonen's instructions [4]. As it is discovered there does not exist any simple rule to determine what kind of scaling is the best. Our selection for vector component units is shown in Eq. (10). There is no need for additional scaling, because the feature vector units are selected to be approximately in the same scale. Cars are mainly classified based on their RRs, because matrix **S** has the most weight and columns in the matrix **F**. The vector component "age" has the lowest weight in the training. These additional variables are not set in our final experiments as *mask* variables to ensure cars with rejection reasons (RR) to be settled in the adjacent cells in SOM output.

Besides the heuristic map size selection built in the SOM Toolbox, the map size is based on intuition that each car should have one cell in the map, so the selected map size was *1060*. The actual size was somewhat smaller, because the function is based on the side lengths so that their product is as close to the desired number of map units.

The top left part of "2011 tires" -component plane is shown in Fig. 2. An average rejection rate per average kilometers driven is used as an estimate to find out which cars perform bad or well in all car inspections, see Fig. 3.

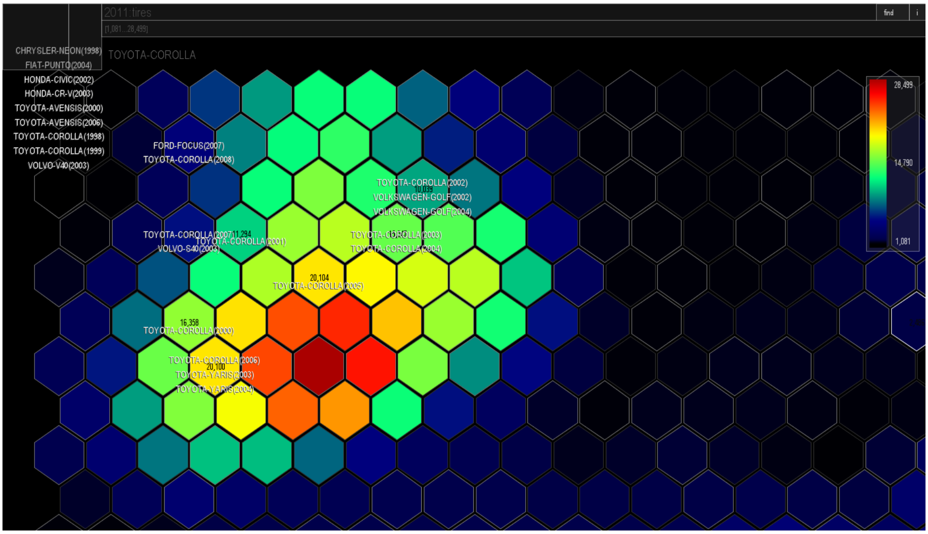


Fig. 2. A part of component plane of tires (2011) which contain Toyota Corolla labels shown by "find" -command. The sum of all 39 codebook values for each cell is approximately *# years data published*100%*. In practice it means that probabilities for driver dependent RRs are then smaller. By the feature vector selection, the user can easily explore dependencies in time and between the flaws.

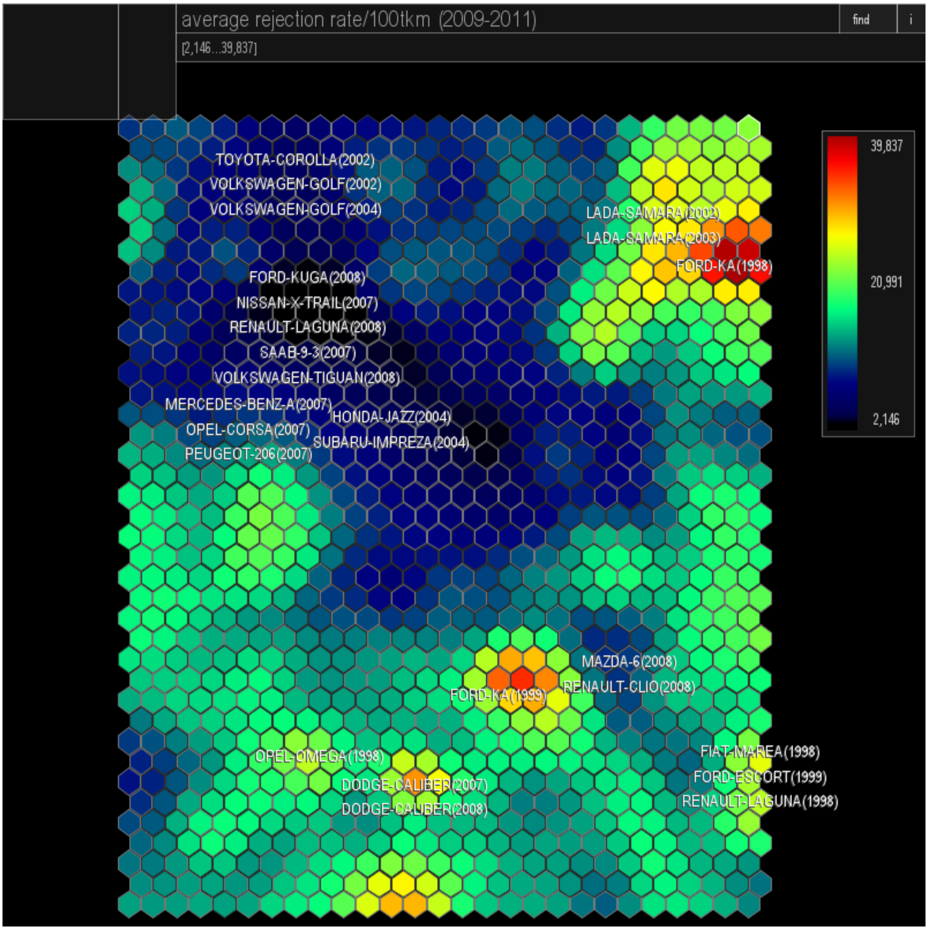


Fig. 3. The car labels which had either small or large mask variable value are shown in this derived component plane. It is the proportion of RRs and average kilometers using codebook values. The best cars based on this feature are situated near to the center and on the top left parts of the map. Vector components *d*, *r* and *a* have some influence in the mapping. Grey hex borders allocate label(s) in a cell.

In addition to the component plane visualizations, we used the SOM codebook information to filter out the effect of tires. In practice this fault is completely driver dependent. Of course, some drivers can break other parts too, steering etc. New estimation for filtered rejection reason $r*$ is derived by reducing three codebook values (tires) from the rest and multiplying it by the rejection rate codebook values. After this procedure, these values are sorted. A list from the best to the worst car was achieved. For example, Toyotas shown in Fig. 2 got a better ranking after this procedure than without it. For example, from the complete listing it is seen that Honda HR-Vs (1999-2001) got better ranking points than many cars which were introduced into use in 2008.

6 Conclusions

In our research, we show that SOM is an effective and very informative method for exploring this type of problem. Data exploration experience can be improved by selecting a suitable interactive visualization tool; therefore we developed it during our research. Even though the provided data was not complete, by the suitable preprocessing procedure, the problem of missing values and discrete data input was filtered by Collaborative Filtering. The main results are reported in this paper. In addition, complete car inspection data exploration is made possible for the reader, who may download our visualization tool with this data set [8].

Our work is still in progress and in future work, we will apply K-Means clustering to classify cars to "good", "average" and "bad" cars related to the car inspection data. More reliable results can be achieved by adding car inspection data provided by other companies. We will also develop our tool, to ensure efficient SOM exploration experience for other researchers.

Acknowledgment. We would like to thank Aalto University, Helsinki Doctoral Programme in Computer Science - Advanced Computing and Intelligent Systems (Hecse) and A-Katsastus for providing the data. In addition, we would like to thank Mikko Korpela for proof-reading this paper.

References

1. Xiaoyuan, S., Taghi, M.K.: A Survey of Collaborative Filtering Techniques. In: Advances in Artificial Intelligence, Article ID 421425, 19 pages. Hindawi Publishing Corporation (2009), doi:10.1155/2009/421425
2. Stefanovič, P., Kurasova, O.: Visual analysis of self-organizing maps. Nonlinear Analysis: Modelling and Control 16(4), 488–504 (2011)
3. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: Self-organizing map in Matlab: the SOM Toolbox. In: Proceedings of the Matlab DSP Conference, vol. 99, pp. 16–17 (1999)
4. Kohonen, T., Honkela, T.: Kohonen network. Scholarpedia 2(1), 1568, revision #122029 (2007)
5. A-Katsastus, http://www.a-katsastus.com/ (retrieved on July 2012)
6. Kohonen, T.: The Self-Organizing Map. Proceedings of the IEEE 78(9), 1464–1480 (1990)
7. Abeel, T., Saeys, Y., Rouzé, P., Van de Peer, Y.: ProSOM: core promoter prediction based on unsupervised clustering of DNA physical profiles. Bioinformatics 24(13), i24–i31(2008)
8. Talonen, J.: An Interactive Self Organizing Map (SOM) Visualization Tool, https://sites.google.com/site/somvisualizationtool/

A Model for Mortality Forecasting Based on Self Organizing Maps

Marina Resta[*] and Marina Ravera

Department of Economics, University of Genova, via Vivaldi 5, 16126, Genova, Italy
{resta,ravera}@economia.unige.it
http://www.economia.unige.it

Abstract. In this paper we introduce a general model framework based on Self Organizing Maps (SOMs) to explore the behavior of populations mortality rates and life expectancy. In particular, we show how to employ SOM clustering capabilities to construct coherent mortality rates, i.e. mortality rates that can be applied unchanged to a wide range of countries. To such purpose, we will employ various countries mortality data downloaded from the Human Mortality Database. Our aim is two–fold. On the one hand, we are going to prove that a data mining approach can be meaningful to build mortality forecasts in a way which is less pretending (in terms of both computing time and parameters to estimate) than traditional techniques. This issue is very important, provided that mortality forecasts are widely employed to develop insurance products. On the other hand, we will show that SOM clustering can be very effective to extract similar mortality patterns from apparently very different countries, thus highlighting non–linear hidden features that are missing for more standard techniques.

Keywords: Longevity risk, Self Organizing Maps, Clustering, Mortality forecasting.

1 Background

Mortality forecasting is an important topic, as it may considered the basis of social and economic planning, and fundamental to many other forecasting exercises as well. In particular, in this paper we are concerned with the link existing between mortality trends and insurance contracts, namely those contracts providing individuals with annuities, pensions and other benefits paid during their lifetime (the so–called *living benefits*).

The main issue is of financial (and balancing) nature: on the one hand, paying benefits implies that insurance companies must have a proper *reserve*, i.e. a fund from which money can be retrieved; on the other hand, pensions and annuities are usually paid depending on proper amounts of money (premium) the individuals have conveyed throughout their active (i.e.: at work) life. The balance between such different amounts of money is guaranteed if and only if the behavior of

[*] Corresponding author.

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 335–343.
DOI: 10.1007/978-3-642-35230-0_34

mortality rates is correctly estimated. However, since mortality rates in many countries are persistently decreasing, the systematic misunderstanding of such behavior could lead to serious financial consequences in the longer term, as far as their premiums and reserves are concerned. This focus has led to identify *longevity risk* [9] as a new type of risk affecting the management of annuity and pensions portfolios.

Provided the importance of the issue, a number of methodologies have been proposed to model (and forecast) the dynamics of mortality rates, although it aids to remember that choosing of methodology is not without controversy, since it can lead to very marked difference in forecasts [7], [8]. Actually more popular models are trend–based, and they can be viewed as belonging to the research vein pionereed by the Lee–Carter model –LCM–[4], we will explain in detail in Section 2. In a nutshell, LCM assumes to represent mortality rates as functions of age x and time t, identifying a single time index which summarises past trends, which affects mortality at time t at all ages simultaneously, and which can be modelled with a view to extrapolation. Over the past decades several weaknesses of LCM have been highlighted, and various modification of the original model have been suggested (see among others: [3], [1], [6]).

Despite of the wide literary corpus, however, the techniques actually in use are of heavily statistical type, and soft computing approaches are rather unexplored. With this is mind, we are going to introduce a general model framework based on Self Organizing Maps (SOMs) [2], to explore the behavior of populations mortality rates. In particular, we will focus on so–called coherent models, and we will explore mortality data of various countries (downloaded from the Human Mortality Database–HMD) in search of similar mortality experiences. In this way we will be able to show how to employ SOM clustering capabilities to construct coherent mortality rates, i.e. mortality rates that can be applied unchanged to a wide range of countries. Our aim is two–fold. On the one hand, we are going to prove that a data mining approach can be meaningful to build mortality forecasts in a way which is less pretending (in terms of both computing time and parameters to estimate) than traditional techniques. On the other hand, we will show that SOM clustering can be very effective to extract similar mortality patterns from apparently very different countries, thus highlighting non–linear hidden features that are missing for more standard techniques.

The structure of the paper is therefore as follows. In Section 2 we will introduce definitions and notational conventions related to the notion of mortality trend, to move then to the description of the Lee-Carter model. Section 3 will be devoted to the presentation of our simulations and to the discussion of related results. Section 4 will conclude.

2 Mortality Trends and Related Issues

2.1 Understanding Actuarial Notations

Modelling the dynamics of mortality rates over time implies to understand the data we are dealing with. Assume the random variable $D_{x,t}$ to denote the number of deaths in a population at age x and time t. Corresponding realizations are generally denoted by $d_{x,t}$, and represent the observed number of deaths, while $e_{x,t}$ generally refers to the matching exposure (in person-years) to the risk of death. The probability of death at age x for a given time t is then given by: $q_{x,t} = \frac{d_{x,t}}{d_{x-1,t}}$. Finally, empirical mortality rates are given by: $m_{x,t} = \frac{d_{x,t}}{e_{x,t}}$ whose stochastic counterpart is the hazard rate (or force of mortality) for age x and time t: $\mu_{x,t}$. In order to provide a cross classification, one can fix a calendar year t in the range $[t_1, t_n]$, and an age x in the interval $[x_1, x_k]$, either grouped into k ordered categories, or by individual year (range k). The main issue an actuary must face is how to model $\mu_{x,t}$ for every $t \in [t_1, t_n]$ and $x \in [x_1, x_k]$.

2.2 The Lee–Carter Model

As said in Section 1, Lee and Carter [4] suggested a framework to model the force of mortality $\mu_{x,t}$ for age x and time t:

$$\ln \mu_{x,t} = \alpha_x + \beta_x \,\kappa_t + \epsilon_{x,t}, \tag{1}$$

subject to the constraints:

$$\sum_{t=t_1}^{t_n} \kappa_t = 0, \text{ and: } \sum_{x=x_1}^{x_k} \beta_x = 1 \tag{2}$$

Here α_x is a fixed parameter exploiting the age profile; by Eqs.(1)–(2) it is possible to prove [4] that the least squares estimator of α_x is given by:

$$\hat{\alpha}_x = \ln \prod_{t=t_1}^{t_n} \mu_{x,t}^{1/h}, \; h = t_n - t_1 + 1. \tag{3}$$

In this way α_x expresses the fixed general shape of the logarithmic transformation of the age–specific mortality rates. For what it concerns remaining parameters, κ_t describes the underlying time trend, while (constant) β_x is the sensitivity of $\ln \mu_{x,t}$ at age x to the time trend represented by κ_t. Finally, $\epsilon_{x,t}$ renders age and time specific effects not captured by the model, and it is assumed to be an independent, identically distributed random variable.

In order to fit the model, [4] proposed a three–steps procedure detailed on following.

Step 1. Estimate α_x as from Eq.(3) above.

Step 2. Compute the matrix of statistics $[Z_{x,t}] = [\ln m_{x,t} - \hat{\alpha}_{x,t}]$ and then estimate κ_t and β_x as, respectively, first right and first left singular vectors in the Singular Value Decomposition (SVD) [10] of the matrix $[Z_{x,t}]$ subject to the above constraints.

Step 3. Adjust the estimated κ_t such that, for each t:

$$\sum_{x=x_1}^{x_k} d_{x,t} = \sum_{x=x_1}^{x_k} e_{x,t} exp\left(\hat{\alpha}_x + \hat{\beta}_x \hat{\kappa}_t\right), \text{ for all } t \tag{4}$$

By running the procedure one can get proper estimates for $\mu_{x,t}$, and hence it will be able to derive any other related actuarial variable.

3 Simulation and Results

3.1 Experimental Settings

We build a framework aimed to develop coherent mortality forecasts. This choice may be easily justified: over the past two decades the populations of the world have become more closely linked by communication, transportation, trade, technology, and disease [5]. It is then reasonable and perfectly straightforward to forecast mortality for a pool of countries (and hence populations), taking advantage of commonalities in their historical experience and age patterns. Obviously populations that are sufficiently similar to be grouped together might have somewhat different mortality histories; however, such past differences should not lead to continuing long-run divergence in the future.

With this in mind we employed data extracted from the Human Mortality Database (HMD)[1], that contains original calculations of death rates and life tables for national populations (countries or areas), as well as the input data (death counts from vital statistics, census counts, birth counts, and population estimates from various sources) used in constructing those tables. Six data types are available from the HMD: births, deaths, population size (annual estimates), exposure to risk of death, death rates, and life tables. At present the database contains detailed data for 37 countries: Table 1 lists the countries as well as the acronym we employed to refer to them in our simulations.

Table 1. Countries included in the Human Mortality Database and related abbreviations

Country & ID	Country & ID	Country & ID
Australia (AUS)	Germany (GER)	Norway (NOR)
Austria (AUT)	Hungary (HUN)	Poland (POL)
Belarus (BIE)	Iceland (ICE)	Portugal (POR)
Belgium (BEL)	Ireland (EIRE)	Russia (RUS)
Bulgaria (BUL)	Israel (ISR)	Slovakia (SLK)
Canada (CAN)	Italy (ITA)	Slovenia (SLO)
Chile (CHI)	Japan (JAP)	Spain (SP)
Czech Rep. (CR)	Latvia (LAT)	Sweden (SWE)
Denmark (DEN)	Lithuania (LIT)	Switzerland (SWI)
Estonia (EST)	Luxembourg (LUX)	Taiwan (TW)
Finland (FIN)	Netherlands (NL)	United Kingdom (UK)
France (FRA)	New Zealand (NZ)	U.S.A. (USA)
		Ukraine (UKR)

[1] http:\www.mortality.org

In our simulations we employed life tables: we can think to them as matrices whose components are time (t), age (x), observed number of deaths ($d_{x,t}$), exposure to risk of death ($e_{x,t}$), probability of death ($q_{x,t}$), and empirical mortality rates ($m_{x,t}$): while generally it is $x \in [0, 110]$, since all ages from birth ($x = 0$) to extremal age (i.e. the highest age at which someone in the population is still living, e.g.: $x = 110$) are represented, t depends on the year from which the country's demographic bureau began to collect data. In the case of Sweden, for instance, data began to be collected since 1751, so that the available life table has more than $28,000$ entries (obtained as 111×258, i.e. 111 years for each collection time $t = 1751, \ldots, 2009$). Moving to Russia and Ukraine, on the other hand, the dataset is sensitively smaller (approximately $6,000$ rows), because data began to be collected after 1953. In order to make meaningful comparisons, we use as starting time $t = 1960$, thus having for each country an input matrix of 5439 rows. Moreover, although it is possibile to access and examine separated life tables for both male and female populations, we considered global life tables, giving statistics for the population as whole.

We then implemented a three steps procedure running as follows.

Step 1. For each country's lifetable we run a separate SOM, with rectangular topology, initialization at random, and logarithmic transformation of all input variables (with the exception of time and age that have been used to label the data and hence have not been processed).

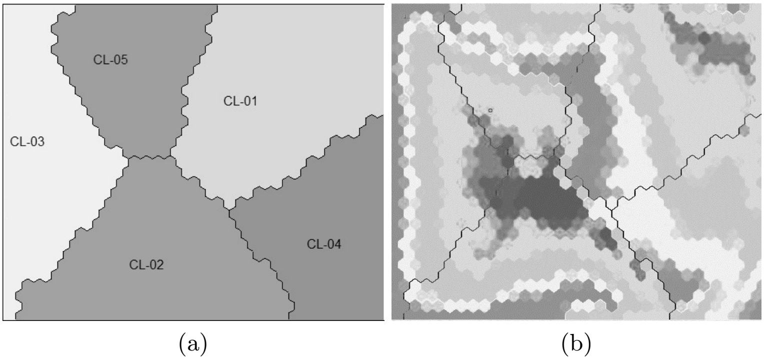
Step 2. We then examined the similarity among maps obtained in the previous step, thus getting a 37×37 symmetric scores table SCT, whose generic i, j entry represents the degree of similarity between the i–th and j–th map. Using SCT values we were then able to group countries hence defining the number of populations sharing common mortality features.

Step 3. For each group defined in Step 2. we have then built mortality forecasts, according to formulas already provided in Eqs. (1)–(4).

3.2 Discussion

As said in previous rows, SOMs operate in two stages over three of the implemented procedure. For what is concerning **Step 1.**, Figure 1, representing Australian life tables, offers some insights about the kind of information SOMs can provide.

From left to right, the first picture in Figure 1 represents age–time clusters for the Australian population in the period: 1960–2009. Note that five cluster emerged: data were at least equally distributed among them. Independently from the reference time t, Cluster 1 (CL01) collects data for population aged in the interval $[75 - 97]$, Cluster 2 (CL02) gathers individuals whose age is in the range $[98, 111]$, Cluster 3 (CL03) refers to ages $x \in [31, 60]$, Cluster 4 (CL04) to ages $x \in [0, 30]$, and Cluster 5 (CL05) considers $x \in [61, 74]$. Moving to the second picture, it offers a view into the map organization by time, that is how life tables data referring to different years are spread on the SOM: various gray tones



(a) (b)

Fig. 1. From left to right: age–time clusters, and map organization by time in a sample country (AUS). Various gray tones represent different years.

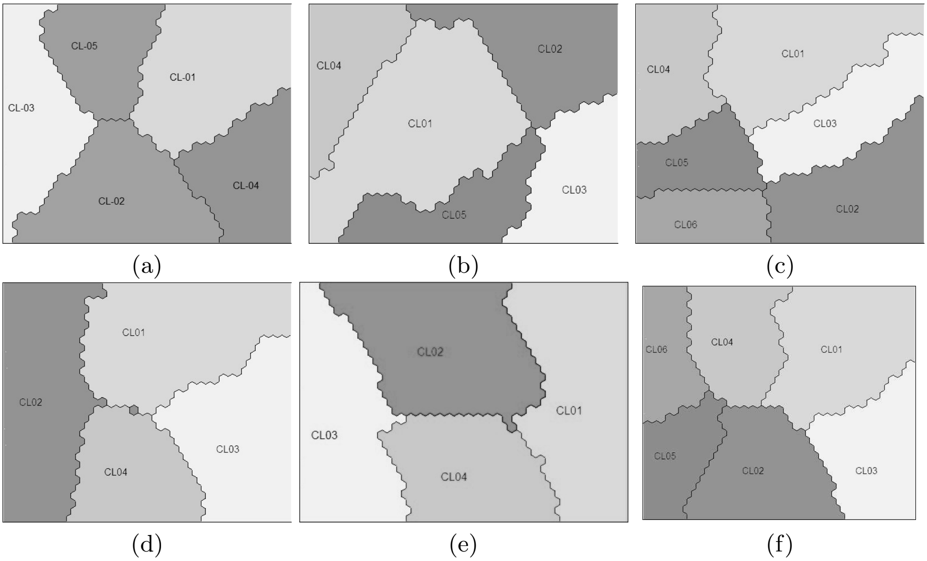
(from white to black) represent different years (in the interval 1960–2009), so that one can easily view that latest years statistics are mainly concentrated on the left hand side of the map, years around later 20th century and earlier 21th century are essentially represented in the internal part of the SOM, while in the center of the map we find data referring to initial years of the sample.

In the second step, we turn to evaluate the similarity among the various maps. This was done by looking at the following factors: (*i*) number of clusters; (*ii*) representativeness of each cluster; (*iii*) ages collected in each cluster. In this way we were able to find out six homogeneous groups (given in Table 2), for which it is then possible to move to **Step 3**, and hence to coherent mortality forecastings.

Table 2. Groups identified by SOM for coherent mortality forecasts. The underlined country is the group central country.

Group 1	Group 2	Group 3	Group 4	Group 5	Group 6
AUS	DEN	BIE	AUT	CHI	CR
CAN	FIN	BUL	BEL	ICE	HUN
EIRE	NOR	EST	FRA	ISR	POL
NZ	SWE	LAT	GER	POR	RUS
UK		LIT	ITA	TW	SLO
USA		UKR	JAP		SLK
			LUX		
			NL		
			SP		
			SWI		

The groups evidence strong coherence among anglo–saxon countries (Group 1), Northern Europe countries (Group 2), Baltic countries (Group 3), (mainly) Western Europe countries (Group 4) and Eastern Europe countries (Group 6). Group 5 appears of residual nature. In order to stress the difference among



(a) (b) (c)

(d) (e) (f)

Fig. 2. From top to bottom and from left to right: SOM organization corresponding to Groups 1 to 6 central countries identified by our procedure. In the top row, moving in clockwise sense, the picture labelled by (a) is associated to Group 1 central country SOM, the picture labelled by (b) corresponds to Group 2 central country SOM, and so on. In the second row, once again in clockwise sense, the picture labelled by (d) is associated to Group 4 central country SOM, and son on up to the picture labelled by (f) which represents Group 6 central country SOM.

countries in the groups, Figure 2 shows the SOM appearance for the central country of each group.

Using data from central countries, we then performed the final stage of our procedure, i.e. mortality forecasting. The main gain deriving from our technique is primarily in the fact that instead of needing to provide different estimations for 37 countries, we are now asked to give six estimations, at each age x, and for every time t in a proper time range. This means obviously a gain in terms of both time and computational efforts.

Figure 3 shows thirty-year life expectancy forecasts $(e_{x,t})$ obtained in the final stage of our procedure for each group central country.

4 Final Remarks

In this paper we introduced a SOM–based framework to model and forecast mortality rates dynamics.

The importance of the topic is related to the emergence of longevity risk, as a new type of risk affecting the management of annuity and pensions portfolios, due to misundertandings in the behaviour of mortality.

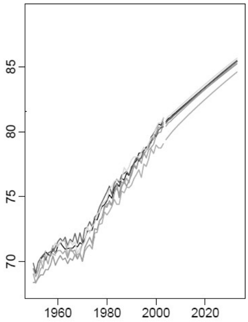


Fig. 3. Coherent life expectancy forecasts for each group central country

The main issue faced by existing methods relies in the fact that in order to provide forecasts at a given time t in future and every age $x \in [0, 110]$, they need a very big amount of information going back in time as much as possible. Moreover, according to the traditional approach, each country must be considered as a unique experience, so that generally forecasts for a population cannot be *tout–court* applied to people in a different geographical area.

Our contribution moves in the research vein of coherent mortality forecasts, assuming that if countries share proper common features (e.g. geographic, politic or economic ones) then they are coherent and hence they can also share mortality statistics and forecasts. We then introduced a three–stages procedure which offers a way to create coherent groups. SOM operate in two of three steps, since in the first phase they are employed to get a representation of countries lifetables, while in the second step the clusters originated by SOMs (in particular: their number, as well as their stastistical representativeness) are used to build coherent groups. Data of central country groups are then employed to provide mortality forecasts.

We tested our approach on 37 countries dataset, as resulting from the Human Mortality Database (HMD). The procedure lets us to identify six meaningful groups, whose composition seems to mirror mainly geopolitic differences: we have groups gathering Anglo–Saxon countries (Group 1), Northern and Eastern Europe countries respectively (Groups 2 and 6), Baltic countries (Group 3), and Western Europe lands (Group 4). Group 5, on the other hand, appears of residual nature, collecting areas with apparently no immediate connections.

The results we have obtained prove the effectiveness of a data mining approach to build mortality forecasts. Besides in this way the estimation procedure is less pretending (in terms of both computing time and parameters to estimate) than traditional techniques. This issue is very important, provided that mortality forecasts are widely employed to develop insurance products. Finally we have shown that SOM clustering can be effective to extract similar mortality patterns from apparently very different countries, thus highlighting non–linear hidden features that are missing for more standard techniques.

References

1. Haberman, S., Renshaw, A.: A cohort–based extension to the LeeCarter model for mortaility reduction factors. Insur. Math. Econ. 38, 556–570 (2006)
2. Kohonen, T.: Self–Organizing Maps. Springer, Berlin (2002)
3. Koissi, M.C., Shapiro, A., Hognas, G.: Evaluating and extending the LeeCarter model for mortality forecasting: bootstrap confidence interval. Insur. Math. Econ. 38, 1–20 (2006)
4. Lee, R., Carter, L.: Modelling and forecasting US mortality. J. Am. Stat. Assoc. 87, 659–671 (1992)
5. Li, N., Lee, R.: Coherent mortality forecasts for a group of populations: An extension of the Lee-Carter method. Demography 42(3), 575–594 (2005)
6. Li, S.H., Chan, W.S.: The Lee-Carter Model for Forecasting Mortality Revisited. Presented at the Living to 100 and Beyond Symposium Sponsored by the Society of Actuaries, Orlando, Fla., January 12-14 (2005)
7. Oeppen, J., Vaupel, J.: Broken limits to life expectancy. Sc. 296, 1029–1031 (2002)
8. Olshansky, J., Passaro, D., Hershaw, R., Layden, J., Carnes, B., Brody, J., Hayflick, L., Butler, R., Allison, D., Ladwig, R.: A potential decline in life expectancy in the United States in the 21st Century. New Engl. J. Med. 352, 1138–1145 (2005)
9. Pitacco, E.: Longevity risks in living benefits. In: Fornero, E., Luciano, E. (eds.) Developing Annuity Market in Europe, pp. 132–167. Edward Elgar, Cheltenham (2004)
10. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: Singular Value Decomposition. In: Numerical Recipes in FORTRAN: The Art of Scientific Computing, 2nd edn., pp. 51–63. Cambridge University Press, Cambridge (1992)

Paths of Wellbeing on Self-Organizing Maps

Krista Lagus[1], Tommi Vatanen[1], Oili Kettunen[2], Antti Heikkilä[1], Matti Heikkilä[3], Mika Pantzar[4], and Timo Honkela[1]

[1] Aalto University School of Science
Department of Information and Computer Science
Espoo, Finland
[2] Sports Institute of Finland, Vierumäki, Finland
[3] Sykettä Elämään Tmi, Orimattila, Finland
[4] National Consumer Research Center, Helsinki, Finland
{krista.lagus,tommi.vatanen,timo.honkela}@aalto.fi

Abstract. In this article, we introduce the concept of pathways of wellbeing and examine how such paths can be discovered from large data sets using the self-organizing map. Data sets used in the illustrative experiments include measurements of physical fitness and subjective assessments related to diagnosing work stress.

1 Introduction

Research on human health is becoming an increasingly multidisciplinary and interdisciplinary endeavor. In addition to the traditional view on health as a biological and medical phenomenon, its cognitive, psychological, social and societal dimensions have been acknowledged as well. A sign of this kind of broadening of research focus is the use of the term “wellbeing” instead of the term “health”. Human health and wellbeing is a dynamic phenomenon that is influenced by a number of variables. In this paper, we present a framework for wellbeing informatics using the self-organizing map (SOM) [10]. We illustrate our approach through an analysis of the development of fitness and its relationship with other wellbeing variables. The SOM has been widely used in the analysis of health-related data (e.g. diseases [6], gene data [17], mental health [14], public health and health care policy [1,2,15,19], nutrition and health [16], and social factors of health [8]).

In a recent study, it was found that being involved in an aerobic training regime in the elderly increased the size of certain parts of hippocampus by about 2%, and also resulted in clear improvements in spatial memory [5]. It has also been shown that long-term stress or single-time very strong stress decreases the size of the hippocampus, eventually leading to work exhaustion or depression [25]. Controlled studies like these strengthen the view that examining the dependencies between work condition, stress, and different types of physical exercise is a relevant research topic.

In this paper, we study wellbeing as a process that develops over time through some states. With this respect, our approach closely resembles methodologically the work that has been conducted in the SOM-based analysis of economic

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 345–352.
DOI: 10.1007/978-3-642-35230-0_35

[3,4,9,13] and industrial [22] processes. In addition to the aspect of modeling process-like data, our focus is in the analysis of individuals and networks of individuals using the self-organizing map (for our earlier related research, see [11,18,24]). In general, our objective is not to present any specific methodological technical improvements. Rather, we present an overall framework for using the SOM in the analysis of wellbeing data, and provide an illustrative experiment of a data analysis in the area of fitness. Moreover, we discuss interdisciplinary connections between computational modeling, physiology, psychology, sociology and information systems design in the domain of wellbeing informatics.

2 Wellbeing as Paths

A multitude of data are nowadays being collected regarding our activities related to individual wellbeing in one way or another. Such data sources include objective measurements such as testing of physical fitness, or subjective questionnaires for assessing, e.g., levels of work stress or depth of potential depression. Current mobile devices related to wellbeing or our daily activities add a further dimension regarding wellbeing over time. In general, everyday practices have become an increasingly popular object of research, and sophisticated qualitative and quantitative methods have been developed to increase understanding of them [20].

Traditionally, results of assessments are only given to the individual at hand, for instance, used for diagnostic purposes, as in the case of depression, or for coaching purposes, as in the case of physical fitness examination. The feedback may entail a report or diagnostic scale on the levels of fitness, stress or depression that the individual is experiencing, but typically no more information regarding the distribution of data or of dependencies between different variables.

However, by looking at the collective data from such tests from a large number of individuals, it becomes possible to obtain a richer view of the situation and to provide a richer feedback to the individual. Data from others, when used anonymously, can also serve as a point of additional learning or coaching. Identifying wellbeing paths from various collected data sets can therefore be considered a relevant enterprise.

Wellbeing informatics can be described as the activity where observations regarding individual wellbeing are collected and analyzed using methods suitable for finding value in large data sets. We define final purpose of wellbeing informatics as to identify, and share socially this information to the relevant other individuals whom it concerns.

Instead of a singular wellbeing path we speak of a multitude of paths of wellbeing. This reflects the viewpoint that there are many quite different ways to lead a wellbeing life, a multitude of approaches and a multitude of states in the state space which can mean wellbeing for different individuals. However, the paths also coincide, they are not totally separate. Bases of coincidence may be, for example, similar life circumstances, similar personality, or similar physical or emotional makeup.

We consider wellbeing as an ability or set of skills to live a happy, fulfilling, good life. Looking at wellbeing from this point of view entails viewing life as a continuous process of learning and development. We learn based on our own experiences and based on the experiences of others. Social sharing is such of great importance as part of learning the skills required for a wellbeing life.

Viewing oneself in the context of other individuals, and their paths, can be a life-changing experience. It can provide a point of reflection on one's own life, a place of facing a painful observation regarding own situation, or a source of hope. It can also serve as a learning experience, lead by the motivating question: what happens to me if I continue on this path where I am now?

3 Maps of Fitness and Stress

Previously, we have studied the relationships between different aspects of physical fitness using fitness test measurements conducted at the Sports Institute of Finland in Vierumäki, Finland [23]. Over the past decades, the Sports Institute of Finland has measured the fitness of approximately one hundred thousand people. In our study, we included about 37,000 subjects who have taken the tests during 2006–2009. The fitness test consists of various measurements aiming to assess aerobic fitness, muscular strength and elasticity. Aerobic fitness was tested using a standard cycle ergometer test which results in age-corrected seven-step test score according to [21]. Muscular strength was assessed using three separate tests measuring leg, arm and abdominal muscle strength. Elasticity measurements are most ambiguous of the three and aim to assess flexibility of sides, hips and shoulders. Additionally, age, gender, body mass index (BMI) and percentage of fat were used in the analysis.

In this paper, we shift our focus from analyzing the relationships between fitness variables in a population to a longitudinal study of individual development. To illustrate our point, we use a data collection that contains a subset of the people included in the previous study. For these 371 people (230 women, 141 men), additional information over fitness variables has been collected. The data collection is based on an intervention lead by one of the authors (O.K.) as a part of her ongoing PhD research, where the subjects took part in five consecutive fitness measurements together with three standardized surveys assessing their stress level, ability to work and somatic symptoms. These measurements and surveys took place over a period of two years during which the participants were also given personalized health and wellbeing advice. In this paper, we do not report the results of the intervention study itself, but use a portion of the data in order to illustrate the use of the self-organizing map in analyzing and visualizing wellbeing data that has been collected as a time series. More specific results of the intervention study will be published separately.

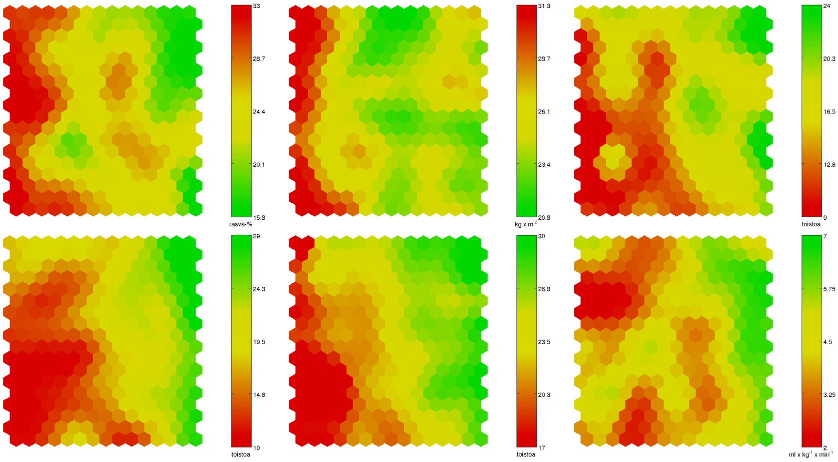


Fig. 1. Distribution of fitness variables on a self-organizing map among female subjects. The variables included in this figure are body fat, body-mass index (BMI) and stomach (upper row, from the left), arms, legs and fitness class (lower row). Green color is used to indicate preferred values of each variable.

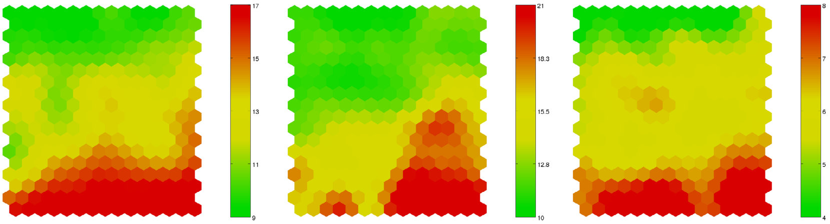


Fig. 2. Distribution of stress variables on a self-organizing map among female subjects: stress symptoms, somatic symptoms and mental resources. Green color is used to indicate low stress and symptom levels and high level of mental resources.

In order to analyze and visualize the data using the SOM, 11 variables were used. The test subjects are anonymous and therefore we cannot show a map of people as such, e.g. relating the position of the people on the map with their background variables. However, the relationships between the variables used in the analysis become visible by showing the distributions of these variables on the map (see Fig. 1 for variables related to the fitness and Fig. 2 for the variables related to the stress). Each measurement was used as a separate data point allowing us to examine movement of the subjects on the map during the intervention.

The structure of the map was analyzed using K-means clustering algorithm. Nine clusters were extracted, shown in Fig. 3. The clusters have been labeled to indicate most important distinctive characteristics of each cluster.

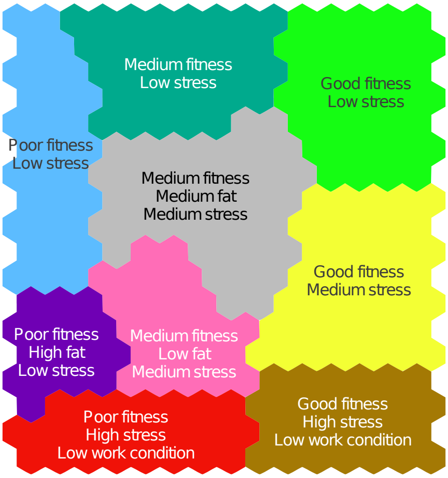


Fig. 3. Clustering structure of the wellbeing map

4 Paths of Wellbeing as Trajectories

Looking at the wellbeing map, where each point describes one state in the wellbeing state space, an individual's wellbeing path then can be viewed as that individual's trajectory over time on the map. The trajectories can be useful in answering questions such as:

- If my fitness is now here, what directions are realistically available for me when data concerning other similar cases is taken into consideration?
- If there are typical trajectories out of my current place, where do they lead?

When different map areas have been identified, based of large number of samples, as clear "crisis zones", the map can also be utilized for identifying worrisome situations and paths. The map can quite concretely and visually be utilized for showing what are the common outcomes from the current situation. In Fig. 4, some real-world examples of paths or trajectories on the wellbeing map are shown.

It is often challenging to motivate oneself to conduct a life change that is required to elicit a clear improvement in wellbeing. Motivation cannot be given from the outside, it cannot be required nor coaxed. It requires reflecting on the current situation, realistically looking into the mirror and understanding what the consequences of current state and current life are. And how would those outcomes feel. For this reason, providing visualizations that accurately and clearly show one's life in the context of other lives, including projected outcomes, can potentially be a powerful tool in creating increased wellbeing.

By looking at the maps and paths of the developments, it is also possible to become conscious of the large variety between different individuals. Every life

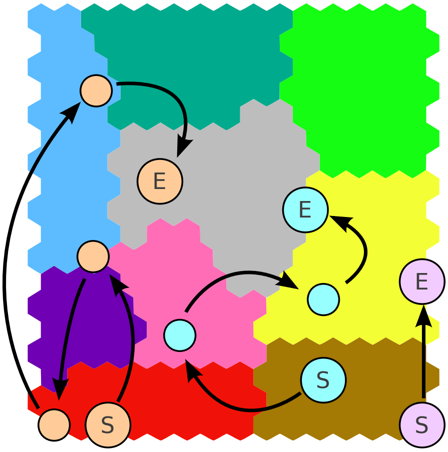


Fig. 4. Examples of trajectories on the wellbeing map. The clustering structure has been shown in Fig. 3.

story is unique, although there may be shared parts of the paths. The paths selected for the visualization, see Fig. 4, were ones that had a change for the better. However, in many paths there were no change for the better, and sometimes changes were for the worse. What we may conclude from this is that a) it is quite difficult to change one's life, as many paths just stayed in the same spot throughout the intervention, b) changes for the better may occur, but back-steps are quite normal as well, and c) also changes for worse do occur.

The map display and viewing one's own progress on it also helps raise new questions: When I moved from the "red state" to the "green state", what was happening in my life then? Becoming conscious of what really happens, and asking questions about why it happens, is what can lead to deep change in behavior. In this way, the SOM of wellbeing regions and pathways can be viewed as a tool for reflecting and becoming conscious of oneself, in order to make better decisions that lead one to one's own goals in life.

5 Conclusions and Discussion

In this paper, we have described a methodological and conceptual framework for supporting wellbeing, based on peer information. The basic idea is to analyze large number of trajectories of the development of wellbeing among individuals to indicate potential paths. These kinds of paths can facilitate well motivated and realistic examples of development. Moreover, if information on interventions is available, personalized data-driven advice can be provided.

The self-organizing map has been used in this work to analyze and visualize the data. It is well suited to the visualization of the wellbeing paths. Also other related methods such as Generative Topographic Mapping (GTM) or Latent Direchlet Allocation (LDA) could be used but the choice of the method does not affect the basic framework. From the point of view of providing specific personalized advice, using probabilistic modeling may be considered useful. One interesting possibility is to conduct analysis of different scenarios, testing the effect of different interventions in an individual case. The number of potentially relevant variables in this domain is potentially huge including biomedical and psychological data as well as data on everyday practices.

The focus of this paper is in the analysis of numerical data. It was shown how the analysis can help as a reflection tool and to provide information on the paths of wellbeing. Additional peer support can be obtained from qualitative sources. The SOM has been used to analyze the contents of document collections (see e.g. [12] as an example of an early work). Recently, we have used independent component analysis and sentiment analysis in text mining of wellbeing-related discussions [7]. An information system designed on the basis of these two areas, quantitative and qualitative, would integrate the facts based on one's own and others' measurements and contextual information with reflections and qualitative comments and peer advice available in textual form. We foresee that this kind of system could promote wellbeing in a substantial manner.

References

1. Basara, H., Yuan, M.: Community health assessment using self-organizing maps and geographic information systems. International Journal of Health Geographics 7(1), 67+ (2008)
2. Cattinelli, I., Bolzoni, E., Barbieri, C., Mari, F., Martin-Guerrero, J., Soria-Olivas, E., Martinez-Martinez, J., Gomez-Sanchis, J., Amato, C., Stopper, A., Gatti, E.: Use of self-organizing maps for balanced scorecard analysis to monitor the performance of dialysis clinic chains. Health Care Management Science 15(1), 79–90 (2012)
3. Cottrell, M., Bodt, E.D., Grégoire, P.: Financial application of the self-organizing map. In: Proceedings of EUFIT 1998, 6th European Congress on Intelligent Techniques Soft Computing, vol. 1, pp. 205–209 (1998)
4. Eklund, T., Back, B., Vanharanta, H., Visa, A.: Using the self-organizing map as a visualization tool in financial benchmarking. Information Visualization 2(3), 171–181 (2003)
5. Erickson, K.I., Voss, M.W., Shaurya, R., Basak, C., Szabo, A., Chaddock, L., Kim, J.S., Heo, S., Alves, H., White, S.M., Wojcicki, T.R., Mailey, E., Vieira, V.J., Martin, S.A., Pence, B.D., Woods, J.A., McAuley, E., Kramer, A.F.: Exercise training increases size of hippocampus and improves memory. Proceedings of the National Academy of Sciences 108(7), 3017–3022 (2011)
6. Gaetano, L., Di Benedetto, G., Tura, A., Balestra, G., Montevecchi, F.M., Kautzky-Willer, A., Pacini, G., Morbiducci, U.: A self-organizing map based morphological analysis of oral glucose tolerance test curves in women with gestational diabetes mellitus. Stud. Health Technol. Inform. 160(pt. 2), 1145–1149 (2010)
7. Honkela, T., Izzatdust, Z., Lagus, K.: Text Mining for Wellbeing: Selecting Stories Using Semantic and Pragmatic Features. In: Villa, A.E.P., Duch, W., Érdi, P., Masulli, F., Palm, G. (eds.) ICANN 2012, Part II. LNCS, vol. 7553, pp. 467–474. Springer, Heidelberg (2012)

8. Honkela, T., Koskinen, I., Koskenniemi, T., Karvonen, S.: Kohonen's Self-Organizing Map in Contextual Analysis of Data. In: Information Organization and Databases: Foundations of Data Organization, pp. 135–148. Kluwer (2000)
9. Kiviluoto, K.: Predicting bankruptcies with the self-organizing map. Neurocomputing 21(1-3), 191–201 (1998)
10. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer (2001)
11. Lagus, K.: Map of WSOM 1997 abstracts - alternative index. In: Proceedings of WSOM 1997, Workshop on Self-Organizing Maps, pp. 4–6. Helsinki University of Technology, Neural Networks Research Centre (1997)
12. Lagus, K., Honkela, T., Kaski, S., Kohonen, T.: Self-organizing maps of document collections: A new approach to interactive exploration. In: Simoudis, E., Han, J., Fayyad, U. (eds.) Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp. 238–243. AAAI Press (1996)
13. Lendasse, A., Lee, J.A., Wertz, V., Verleysen, M.: Forecasting electricity consumption using nonlinear projection and self-organizing maps. Neurocomputing 48(1-4), 299–311 (2002)
14. Mabruk, A.F., Yousif, J.H.: Self-organizing map approach for identifying mental disorders. International Journal of Computer Applications 45(7), 25–30 (2012)
15. McGaugh, M.: A practical application of self-organizing maps in public health. In: 1st International Conference on Innovation and Entrepreneurship in Health. Oklahoma State University (2012)
16. Mehmood, Y., Abbas, M., Chen, X., Honkela, T.: Self-Organizing Maps of Nutrition, Lifestyle and Health Situation in the World. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 160–167. Springer, Heidelberg (2011)
17. Oja, M., Sperber, G.O., Blomberg, J., Kaski, S.: Self-organizing map-based discovery and visualization of human endogenous retroviral sequence groups. Int. J. Neural Syst. 15(3), 163–179 (2005)
18. Paju, P., Malmi, E., Honkela, T.: Text Mining and Qualitative Analysis of an IT History Interview Collection. In: Impagliazzo, J., Lundin, P., Wangler, B. (eds.) History of Nordic Computing 3. IFIP AISC, vol. 350, pp. 433–443. Springer, Heidelberg (2011)
19. Resta, M.: Assessing the Efficiency of Health Care Providers: A SOM Perspective. In: Laaksonen, J., Honkela, T. (eds.) WSOM 2011. LNCS, vol. 6731, pp. 30–39. Springer, Heidelberg (2011)
20. Shove, E., Pantzar, M., Watson, M.: The dynamics of social practice: Everyday life and how it changes. Sage (2012)
21. Shvartz, E., Reibold, R.C.: Aerobic fitness norms for males and females aged 6 to 75 years: a review. Aviat. Space Environ. Med. 61(1), 3–11 (1990)
22. Simula, O., Vesanto, J., Vasara, P.: Analysis of industrial systems using the self-organizing map. In: Proceedings of KES 1998, Knowledge-Based Intelligent Electronic Systems, pp. 61–68 (1998)
23. Vatanen, T., Heikkilä, M., Honkela, T., Kettunen, O., Lagus, K., Pantzar, M.: Kuntotiedot kartalle - erilaiset hyvä- ja huonokuntoisten ryhmät näkyviin. Liikunta & Tiede (Sports & Science), pp. 48–53 (2012)
24. Vatanen, T., Paukkeri, M.S., Nieminen, I.T., Honkela, T.: Analyzing authors and articles using keyword extraction, self-organizing map and graph algorithms. In: Proceedings of the AKRR 2008, pp. 105–111 (2008)
25. Wosiski-Kuhn, M., Stranahan, A.M.: Opposing effects of positive and negative stress on hippocampal plasticity over the lifespan. Ageing Research Reviews 11(3), 399–403 (2011)

Exploring Social Systems Dynamics with SOM Variants

Marina Resta

Department of Economics, University of Genova, via Vivaldi 5, 16126, Genova, Italy
{resta}@economia.unige.it
http://www.economia.unige.it

Abstract. We use variants of Self Organizing Maps (SOMs) to simulate how agents interact in social systems. Our efforts were mainly concentrated to model agents learning and psychological relationships, as well as the way those latter can affect the system general behavior. As main result, we developed a suitable environment to simulate economic systems and to simulate its dynamics.

Keywords: Self Organizing Maps, Social Systems, Dichotomous Growth.

1 Introduction

Over the past decade economic dynamics has been intensively simulated by way of soft computing tools. The current interest on such topic may find explanations under different points of view; however, here we agree with [6], who emphasized the importance of viewing to the economy as an evolving network. In such a context, interaction emerges as the leading aspect of modern economic systems, where the individual behavior is perceived like the synthesis of both previous personal experience and partnership effects: our actions affect those of other people; those, in turn, can affect our welfare. This true, reasonable simulations of interaction should take into account at least three interrelated levels of issue: (*i*) the individual level, driven by personal interests; (*ii*) the aggregate level, where global behavior not necessarily emerges as simply cumulation of individual activities; (*iii*) the level of the bi–directional flow, linking individual to aggregate behavior, and vice-versa, so that the macro and micro levels may influence themselves reciprocally. Apart from considerations about its efficacy, an exhaustive dynamical description would hence require the assumption of a system of Partial Differential Equations (PDEs), as wide as the number N of individuals in the model. This obviously makes the problem not easy to handle, especially for larger values of N. In order to overcome this issue, heavy computational methods have been introduced to model phase transition in economic systems: shell models [2], coupled map lattices [5] and cellular automata have been suggested as suitable methods helping to understand social systems basic mechanisms, thus building a bridge between traditional statistical descriptions, and dynamical representation in phase space. The aforementioned methods, in fact, share the common feature

P.A. Estévez et al. (Eds.): *Advances in Self-Organizing Maps*, AISC 198, pp. 353–362.
DOI: 10.1007/978-3-642-35230-0_36

to reproduce economic dynamics by means of some kind of discretization (varying depending on the methodology in use), where the PDEs governing the process are transformed into a set of Ordinary Differential Equations (ODEs). Obviously different capabilities of representation and generalization can be combined in order to get more or less simplified models of the observable world. Starting from the pioneering work of [4], the contemporary literature mainly focused on the use of cellular automata to simulate economic systems and interplays among individuals within them, although there is a serious danger to confuse spontaneous switches, inherited in the algorithm, with endogenous ones [12]. On the other hand, the extensive use of Artificial Intelligence paradigms such as Genetic Algorithms [1], or their ibridisations [9], [10] was criticized already in [3], who proved that sometimes results are due mostly to randomization issues of the Genetic Algorithm, rather than to mating or crossover features inherited into the model itself.

Holding this, our paper analyzes a different approach to economic systems modeling, and addresses the specific field of simulation of interactions by means of spatial connections. This is possible thanks to the particular algorithm in use, which is merely inspired by the idea underlying Self Organizing Maps (SOMs) [7] to retrieve neighborhood interaction through traditional spatial relationships (in our case induced by either Moore, or von Neumann neighborhoods), as well as by means of a Voronoi tessellation of system variables space. In such sense, connections have been assumed relevant both to condition the level of human capital (and hence production), and also propensities to save and to study. The structure of the paper is therefore as follows. Section 2 briefly introduces some technical details concerning the variants of SOMs we employed; Section 3 focuses both on the description of the economic assumptions we made to develope our model and on its algorithmic implementation. Section 4 discusses simulation results, while Section 5 contains some conclusive remarks and outlooks for future works.

2 SOM Variants and Their Significance for Economics Simulations

As widely known, Self Organizing Maps (SOMs, Kohonen maps), are unsupervised neural models, which consist of a number of neurons generally arranged into a two-dimensional grid, driven to preserve topological relationships over the input space, while performing at the same time a dimensionality reduction of the above representation space. The Kohonen algorithm assumes to iteratively modify the map nodes by way of a set of rules; we focused on a slight modification of the original algorithm as suggested in [8]. Consider first a finite set $X = \{u(t)\}_{t=1,...,T}$ of $d-$dimensions input data items: $X \subset \Omega \subset \mathbb{R}^d$. Besides, let us assume that M is the $m \times k$ bi–dimensional projection grid defined into a discrete bi-dimensional output space $\mathbb{Z}^2$, and $\underline{w}_i \in \Omega$ to be the pointer associated to neuron (unit, node) i in the map ($i = 1 \ldots, m \times k$). The initial stage starts in the topological map M whose neurons are arranged in a disordered manner, i.e. at random. At each step t, the input $u(t)$ from a continuous space $\mathbb{R}^d$ is presented to the net, and the algorithm describes a mapping Φ from $\mathbb{R}^d$ to $\mathbb{Z}^2$, according to which a winner or leader neuron is selected in the map when it is: $arg\min_{k \in M} ||\underline{w}_k - \underline{u}||$.

This makes possible to order neurons according to their similarity with the input, as well as to similarity criteria among themselves:

$$p(i) > p(j) \Leftrightarrow (||\underline{u} - \underline{w}_i|| > ||\underline{u} - \underline{w}_j|) \vee (||\underline{u} - \underline{w}_i|| = ||\underline{u} - \underline{w}_j||) \wedge (i > j) \quad (1)$$

where $p(i)$ is the position in M of the winner neuron at time $t+1$. Hence, both the pointer $\underline{w}_i$ associated to leader neuron, and all the pointers $\underline{w}_j$ belonging to a convenient (according to Eq. (1)) neighborhood in the map are modified with the following rule:

$$\Delta\underline{w}_i = h_{ij}\{\alpha, d_{map}[p(i), j, i]\}(\underline{u} - \underline{w}_i) \quad (2)$$

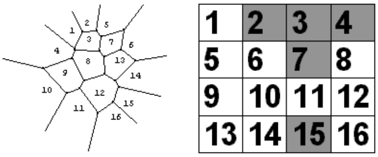
with α being a fixed constant, $d_{map}[p(i), j, i]$ a distance function, and $(\underline{u} - \underline{w}_i)$ is the error between the input and each pointer. Finally, $h_{ij}(\cdot)$ is the neuron interaction function between the nodes: it depends on the distance in the map d_{map} between each node, as well as by the constant α. Throughout our simulation we will assume:

$$h_{ij}[\alpha, d_{map}(p(i), j, i)] = exp\,(-\alpha\,, d_{map}(p(i), j, i)) \quad (3)$$

The learning phase is completed after the whole dataset (if the number of input patterns is finite) has been presented to the map.

It is noteworthy to observe that the variant of SOM algorithm we have therein discussed takes into account spatial relationships at least twice and in quite different ways. At each step, in fact, neurons are ordered both according to Eq. (1), and to Eq. (2). While, in the former case, the Voronoi tessellation of input space (or better its evolution over time) is captured, in the latter the proper learning phase takes place, with information retrieval and exchange both between neurons and the input pattern, and among nodes themselves. To make the concept clearer, consider Figure 1.

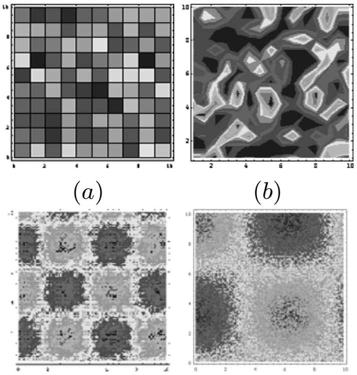
The neighborhood structure deriving from Voronoi tessellation of neural space is generally quite different from the one which comes from ordinary proximity



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Fig. 1. A sketch proof of the organization as resulting both from the Voronoi tessellation of neural space (left), and by applying a von Neumann (cross-shaped) neighborhood, when edges of the neural lattice are pasted together (right)

relationships. To make an example, consider the cross-shaped neighbor with radius one centered on the cell labeled by number 3: the map edges have been pasted together, to avoid border effects. As one can see, the neighborhood of cell 3 includes neurons: 2, 4, 7, and 15. On the other hand, the Voronoi tessellation of neural space assigns to cell 3 different neighbors from the ones previously indicated. A second remark is then noteworthy: if one looks at Eqs. (2)–(3), he might note that keeping α closer to 0 (e.g. $\alpha < 0.01$), the impact of additional information which comes from input tends to be widely spread from the leader nodes to nearest neurons. On the contrary, if α is maintained nearer to 1 (e.g. $\alpha > 0.7$), then neurons within a $\vartheta-$wide ($\vartheta \in \mathbb{N}$) neighborhood amplitude from the leader will be less sensitive to new information than in previous case. Figure 2 shows this idea in a more intuitive fashion.



(*a*) (*b*)

(*c*) (*d*)

Fig. 2. From top to bottom in clockwise sense, a 10 × 10 SOM map at initial step (*a*) and after 1000 iterations (*b*). Neurons are colored according to their similarity (Euclidean distance) to neighbors. Neighborhood effects on the 10 × 10 SOM when ϑ is maintained closer to 1 (*c*), or to 0 (*d*).

The aforementioned features make SOM a quite promising instrument to model human behaviour and interactions into an economic system. The extreme flexibility which is possible to gain by operating over α and d_{map}, in fact, offers the opportunity to reproduce swarm effects, as well as its antonym i.e. the individual specification as sole identity. This in practice means that by properly varying either the value of α, or the shape of the function d_{map}, it is possible to control the learning phase, so that either neighborhoods with same shapes (e.g. cross) have different sensitivity to information spread over them (α is varied) or equal information intensity (α unchanged) may be spread over different shaped areas, thus enforcing (or penalizing, depending of the constant value of α) the effect of original input over the map.

3 The Computational Model

3.1 Preliminar Economic Statements

We consider a two sector growth model, embedded into an overlapping generation system. This means that in our model each individual lives for two periods; at each step he/she chooses how to allocate potential labor between work and study, and hence how to divide wages between consumption and savings. Those, in turn, will be invested in physical capital, to be used in the second part of individuals life. New generations acquire from elders previous situation, and change it through learning and neighborhood effects. The efforts of this study have been primarily focused to model such effects, as well as the impact over decision variables deriving from the existence of notable spatial connections. We have examined different neighborhood topologies, in order to represent both neighborhood effects in strictly geographical sense, and also collective behaviors, induced by affinity and by other psychological motivations. Giving a deeper look to the model, the function ruling the production of tangible good at time t is a Cobb-Douglas function of the type:

$$Q_t^{(i)} = \left[L_t^{(i)} \cdot K_t^{(i)} \cdot H_t^{(i)}\right] \tag{4}$$

Where $Q_t^{(i)}$, $L_t^{(i)}$, and $H_t^{(i)}$ are, respectively, production, labor services, tangible and human capital for the i–th agent. We assumed that young individuals begin their life with an equal amount of potential labor σ, which has to be divided between work and study. Labor services $L_t^{(i)}$, depends then on initial potential labor disposal, as well as on individual propensity to study $v_t^{(i)}$: $L_t^{(i)} = \sigma \left[1 - v_t^{(i)}\right]$. Analogously, physical capital $K_t^{(i)}$ is a function of propension to invest into physical capital $z_t^{(i)}$, and of residual propensity to study bring out from previous step: $K_t^{(i)} = \sigma^2 \, z_t^{(i)} \left[1 - v_t^{(i)}\right]$. Finally, for what is concerning the human capital made available to each agent, it is given by: $H_t^{(i)} = (1-\tau)H_{t-1}^{(i)} + g|v_t^{(i)}|H_t^{*(i)}$. Here τ is a constant value in the interval $[0, 1)$, $H_t^{*(i)}$ allows for positive labor externalities into the model, being the average human capital into the spatial neighborhood of each agent, and $g|v_t^{(i)}|$ is a conditional (non increasing and continuous) function which associates diminishing returns to in incremental efforts in human capital formation. Finally individual's utility function is given by:

$$U_t^{(i)} = \left[\sigma \, v_t^{(i)}\right]^{2/3} Q_t^{(i)} \left\{z_t^{(i)} \left[1 - z_t^{(i)}\right]\right\}^{1/2} .$$

3.2 The SOM–Based Model

Starting from the assumptions discussed in previous paragraphs, a computational model involving SOMs has been developed. We build a rectangular grid of agents, with border joined together, to form a torus with agents lying over a continuous surface. Each individual is associated to a reference vector:

$$x_t^{(i)} = \left\{ v_t^{(i)}, z_t^{(i)}, Q_t^{(i)}, H_t^{*(i)}, H_t^{(i)}, U_t^{(i)} \right\} \tag{5}$$

representing agent's condition. Here $v_t^{(i)}$, $z_t^{(i)}$, $Q_t^{(i)}$, $H_t^{*(i)}$, $H_t^{(i)}$ and $U_t^{(i)}$ have the meaning discussed in previous rows. Both v and z have been assumed as control variables, i.e. those parameters whose evolution can decisively influence the behavior of production, work, and hence the formation of individuals utility profile. At a generic step t the procedure works as follows:

STEP 1. Selection of the best performing unit according to the definition given below.

Definition 1. *(Best performer). The best performer (BPF) unit at time* t *is the agent whose utility has resulted at highest level at step* $t-1$*:*

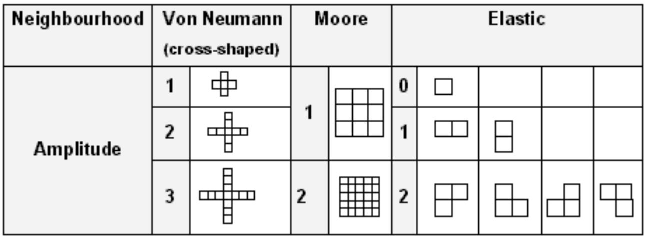
$$BPF_t = arg\max_{i \in M} U_{t-1}^{(i)}.$$

STEP 2. A Voronoi tessellation of neural space is performed, by ordering neurons according to their distance from the couple v_{BPF}, z_{BPF}, of control parameters associated to the Best Performer. This order is then retrieved in the learning procedure through Eq. (3). In this way each agent will acquire new propensities to study and to save, which in turn are used to calculate step values for production Q, utility U, and labor H.

With respect to the SOM variant that we have described in Section 2, we included an additional random perturbation ξ, in order to avoid that tuning nodes position to that of the BPF might lead to super-positions among nodes; in the observable world, in fact, mimicking the behavior of other agents rarely lead to reach the goal exactly, but rather it is a task severely affected by noise. Hence once the shape of d_{map} is properly chosen, it will be possible to force the net to give more or less emphasis to the proximity of neurons.

4 Results Discussion

Our simulation relied on Self Organizing Maps made by 400 neurons arranged into a rectangular 20x20 lattice, with edges pasted together to avoid border effects. Each neuron was structured as explained in Sec.3, i.e. it was associated to a 6th-uple like that of Eq. (5), originally set at random. In order to model nodes (agents) proximity we considered three different types of neighborhood : (a) von Neumann or cross shaped, (b) Moore, and (c) *elastic* neighborhood: Figure 3 shows some examples of different kinds of neighborhood. From now we will refer to those types of proximities by means of the labels: $VN(r)$, $MN(r)$, and $EN(r)$ to indicate von Neumann, Moore, and elastic neighborhood respectively, with radius amplitude r. Note that the expression *elastic* means that we introduced a system of clique typologies, hence giving each neuron the chance to mutate the shape (and the amplitude) of its neighborhood, according to its fitness respect on the whole system. Within this context the meaning of fitness must be intended in the sense of welfare level.



Neighbourhood	Von Neumann (cross-shaped)		Moore		Elastic				
Amplitude	1		1		0				
	2				1				
	3		2		2				

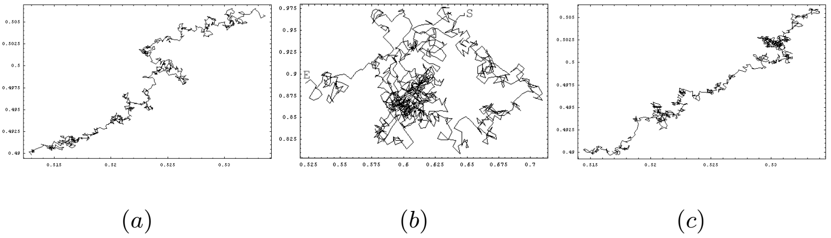
Fig. 3. A snapshot on neighborhood used throughout the simulation

Table 1. Parameters set

	σ	τ	α	ξ
Value	10	0.02	0.6	0.01

System parameters have been maintained constant as shown in Table 1.

Simulating economic system dynamics offers a huge number of challenging issues. Being this work a first time application, here we are mainly interested to show the potential inside this approach. For this reason (and because there is not enough room to go into deepest detail), we are going to discuss only a snapshot of the results we have obtained. Starting from the dynamics of the average distribution of propensities to study v, and to invest into physical capital z, Figure 4 shows their behavior, when the initial values are fixed as (a) extracted from random variables uniformly varying in the range $(0, 1)$, (b) closer to one, or (c) closer to zero.

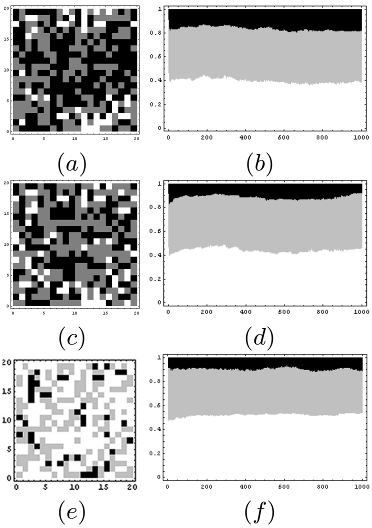


(a) (b) (c)

Fig. 4. Paths towards 1000 runs for the couple (v, z), when both v and z are initialized at random (a) uniformly in the range $(0, 1)$, (b) closer to 1, (c) closer to 0

One can note that when (v, z) are set as uniformly distributed random variables in the range $(0, 1)$ their values tend to maintain closer to average values (i.e. 0.5) over the whole procedure (Figure 4 (a)). Indeed, whereas the couple

(v, z) is forced to maintain closer either to one or to zero (Figure 4 (b) and (c) respectively), the couple (v, z) spans a path which drives them to reach values around $0.5 - 0.52$. These results appear stable, in the sense that those appear to be limit values to which simulations converge, independently either by the kind of neighborhood adopted, or by the particular initial conditions. Additional simulation results are then summarized in Figure 5, where the first row refers to the SOM behavior when the Von Neumann neighborhood ($VN(1)$) with radius amplitude 1 is applied; in the second row we find results for Moore neighborhood with radius one ($MN(1)$); finally the third row shows the results for the elastic neighborhood. In this latter case, simulations were driven assigning to proximities with radius 0 and 1 much more probability to be chosen by agents, i.e. more attention has been focused on the simulation of *egoistic* politics. In this way, although in a still schematic fashion, it has been attempted to capture the capability of the model to emulate different human behaviour, when less or more structured crowd effects are present (like in the case of VN(r) and MN(r)), or when the sole identity dominates over all possible behaviours (EN(r)). Besides, each row addresses two different issues. The first one concerns the position in the map of agents with higher-low fitness in terms of production. Different tones of gray indicate different levels of production, black and white representing opposite situations, that is the highest and the lowest values of production reached by single cells; A second discussion issue (see in Figure 5, pictures labelled by (b), (d) and (e), depending on the neighborhood in use) concerns the distribution (in percentage terms) of agents with high-low welfare levels. Agents are grouped according to their welfare: black stand for the poorest, white for the richest, gray levels for intermediate conditions. By way of the Voronoi tessellation of input space we induced affinity relations rather than proximity ones. This is in agreement with the existence of agents sharing equal levels for propensities to study as well as to save although they are spatially placed on different regions. However, while (v, z) are driven over a path which bring them to converge on similar steady values, independently from initial conditions, this, on the other side, does not always holds for welfare. In almost 90% of monitored cases, in fact, the original distribution of welfare still maintains unchanged over the whole simulation: i.e. despite from changes in propensity to both study and work rich people remains rich and the same holds for poorer. We can then conclude that if the affinity is not accompanied by proper spacial conditions, the affinity by itself is not able to modify existing situations. This brings to conclude that equally trained agents tend to show different productivity, according to the particular spatial context they are placed in. At the same time, the distribution of wealth appears variously structured, in accordance to the neighborhood shape which prevails in the simulation. In other words, the adoption of egoistic rules should produce limited imitation effects and dichotomization; since the regional dimension has been inflated in the model through the conditioning of labor externalities (namely through H*), it should be reasonable to deduce that , when the shape of neighborhood is wider enough, those play a major impact on the welfare level.



(*a*) (*b*)

(*c*) (*d*)

(*e*) (*f*)

Fig. 5. From top to bottom in clockwise sense, each row shows two pictures. The first one in each row always represents the final organization (after 1000 iterations) of the 20×20 SOM map in the $VN(1)$ (a), $MN(1)$ (c), and $EN(1)$ (e) case respectively. Neurons are colored according to their similarity (Euclidean distance) to neighbors.

5 Conclusion

This work has focused on the plausibility of computer simulation to reproduce the dynamic of economic systems. In particular, Self Organizing Maps (SOMs) were not considered relevant as an analysis tool of social or economic data, as, for example, in [11], but they have been introduced as operative tool: thanks to their extreme flexibility, by properly varying control parameters, it is possible to drive them to reproduce a wide variety of situations, useful to emulate (obviously in a simplified way) real world dynamics. To this purpose, it has been pointed on how SOMs inherited features could be use to represent both affinity among individuals (and hence their psychology), and regional proximity. Starting from this point, various simulations were implemented, using a variety of possible neighborhood, in order to test the emergence of dichotomous growth, and some possible explications of such phenomenon. From the simulations, it has emerged, that, although psychologically similar, agents may be strongly influenced by regional factors. Spatial connections, in turn, are not always significant at the same level, and either systems dominated by strongly community relationships, or systems where individual politics prevail may show dichotomization in growth and development. This makes possible to think to the existence of an "optimal threshold" for radius neighborhood amplitude, beneath which proximity effects tend to be soften. This conclusion has been supported by looking at the dynamics of the simulated artificial world with proximity affinities induced through propensities to study v and to invest into physical capital z, and regional neighborhood structure inflated through the presence of labor externalities.

References

1. Arifovic, J.: Strategic Uncertainty and the Genetic Algorithm Adaption. In: Amman, H., Rustem, B., Winston, A. (eds.) Computational Approaches to Economic Problems. Kluwer (1997)
2. Bohr, T., Jensen, M.H., Paladin, G., Vulpiani, A.: Dynamical Systems Approach to Turbulence. Cambridge University Press, Cambridge (1998)
3. Geisendorf, S.: Genetic Algorithms in Resource Economics Models, A Way to Model Rationality in Resource Exploitation? Draft Paper (1998)
4. Hegselmann, R.: Modelling Social Dynamics by Cellular Automata. In: Liebrand, B.G., Nowak, A., Hegselmann, R. (eds.) Computer Modelling of Social Processes, pp. 37–64. Sage Publications (1998)
5. Kaneko, K.: Overview of Coupled Map Lattices. Chaos 2(3), 279 (1992)
6. Kirman, A.: The Economy as an Evolving Network. J. Evol. Ec. 7 (1997)
7. Kohonen, T.: Self–Organizing Maps. Springer, Berlin (2002)
8. Martinetz, T., Schulten, K.: Topology Representing Networks. Neur. Net. 7(3) (1994)
9. McCain, R.A.: Localized Romer Externalities and Dichotomous Development: Simulations with a Cellular Genetic Automaton. Working Paper (1998)
10. McCain, R.A.: Backwash and Spread, Effects of Trade Networks in a Space of Agents Who Learn by Doing. Working Paper (1999)
11. Niemelä, P., Honkela, T.: Analysis of Parliamentary Election Results and Socio-Economic Situation Using Self-Organizing Map. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 209–218. Springer, Heidelberg (2009)
12. Wuensche, A.: Classifying Cellular Automata Automatically: Finding Gliders, Filtering, and Relating Space-Time Patterns, Attractor Basins, and the Z Parameter. Compl. 4(3), 7–66 (1999)

Author Index