Jorma Laaksonen
Timo Honkela (Eds.)

LNCS 6731

Advances in Self-Organizing Maps

8th International Workshop, WSOM 2011
Espoo, Finland, June 2011
Proceedings

Springer

Lecture Notes in Computer Science 6731

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Jorma Laaksonen Timo Honkela (Eds.)

Advances in Self-Organizing Maps

8th International Workshop, WSOM 2011
Espoo, Finland, June 13-15, 2011
Proceedings

Springer

Volume Editors

Jorma Laaksonen
Aalto University School of Science
Department of Information and Computer Science
00076 Aalto, Finland
E-mail: jorma.laaksonen@aalto.fi

Timo Honkela
Aalto University School of Science
Department of Information and Computer Science
00076 Aalto, Finland
E-mail: timo.honkela@aalto.fi

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-21565-0 e-ISBN 978-3-642-21566-7
DOI 10.1007/978-3-642-21566-7
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011928559

CR Subject Classification (1998): F.1, I.2, D.2, J.3, H.2.8, I.4, I.5

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

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

Preface

The 8th Workshop on Self-Organizing Maps, WSOM 2011, was the eighth event in a series of biennial international conferences that started with WSOM 1997 at the Helsinki University of Technology.

WSOM 2011 brought together researchers and practitioners in the field of self-organizing systems, with a particular emphasis on the self-organizing map (SOM). When academician Teuvo Kohonen was conducting his pioneering work with a small number of colleagues in the 1970s and 1980s, the prospects of neural network research were not widely acknowledged. The main focus was on artificial intelligence research based on symbol manipulation methodologies. As notable exceptions, Teuvo Kohonen as well as Stephen Grossberg, Shun-ichi Amari and Christoph von der Malsburg continued their efforts regardless of criticism that was often based on short-sighted interpretations of the book *Perceptrons* published in 1969 by Marvin Minsky and Seymour Papert.

For a long time, and regrettably also often these days, the term neural networks was considered to be synonymous with multilayer perceptrons. However, multilayer perceptrons have given ground to more advanced forms of supervised learning including support vector machines. Actually, among the three classic neural network paradigms – multilayer perceptrons, Hopfield nets and SOMs – only the last one has remained in a strong position. The persistent interest in the SOM algorithm can perhaps be explained by its strength as an unsupervised learning algorithm and by its virtues in analyzing and visualizing complex data sets.

Presently, research on artificial neural networks is a well-established scientific discipline and an area of technological development with a large number of applications. Artificial neural network research can be divided into three main strands: (1) explicit modeling of biological neural circuits and systems, (2) neurally inspired computing, and (3) statistical machine learning research that has mostly abandoned its biologically inspired roots. This classification cannot be considered clear-cut, but rather a continuum. In his banquet keynote talk at the IJCNN 2007 conference, Michael Jordan emphasized the importance of the neural network research for its role in facilitating the path to current statistical machine learning research. Obviously, the biological inspiration helped in abandoning some restricting assumptions that were commonly held in classic statistical computing.

There are hundreds of different kinds of variants of the basic SOM algorithm, each typically proposing some advantage by giving up an aspect of the original formulation, such as computational efficiency, capabilities in visualization, implementational simplicity or biological relevance. In general, the SOM has inspired a lot of methodological research and provided a tool for a large number of real-world cases.

The WSOM 2011 event covered the results of research in theory and methodology development as well as selected examples of applications. When applications of the SOM are considered, it is good to keep in mind that the thousands of uses of the SOM in different fields of science are usually reported in the specific fora of each discipline. Moreover, the commercial projects based on the SOM are typically not reported publicly, but there are many indications that the entrepreneurial use of the SOM and its variants in data analysis, knowledge management and business intelligence is widely spread.

The technical program of WSOM 2011 consisted of 36 oral or poster presentations – by a total of 96 authors – that highlighted the key advances in the area of self-organizing systems and more specifically in SOM research. We warmly thank all the authors of the contributed papers. We also gratefully acknowledge the contribution of the plenary speakers. The plenary presentations were given by Barbara Hammer (University of Bielefeld, Germany) and Teuvo Kohonen (Academy of Finland and Aalto University, Finland). The event celebrated the 30th anniversary of the first report in which Kohonen presented the basic principles of the SOM, and the 10th anniversary of the 3rd edition of his book *Self-Organizing Maps*. We also celebrated that the number of SOM-related scientific papers has reached approximately 10,000.

We warmly thank the highly respected international Steering and Program Committees whose roles were instrumental for the success of the conference. The Program Committee members and the reviewers ensured a timely and thorough evaluation of the papers. We are grateful to the members of the Executive Committee. In particular, the experience of Olli Simula as the Local Chair and the skillful efforts of Jaakko Peltonen as the Publicity Chair contributed greatly toward the success of the event.

WSOM 2011 was co-located with the ICANN 2011 conference. We wish to thank the organizers of ICANN 2011, especially General Chair Erkki Oja, Local Chair Amaury Lendasse and Finance Chair Francesco Corona. The smooth collaboration with them facilitated the success of WSOM 2011. Last but not least, we would like to thank Springer for their co-operation in publishing the proceedings in the prestigious *Lecture Notes in Computer Science series*.

The organizers had a chance to welcome the participants to the new but prestigious Aalto University School of Science. Namely, from the beginning of 2010, the 100-year-old university changed its name and form. Three universities, Helsinki University of Technology, Helsinki School of Economics, and University of Art and Design Helsinki, merged into Aalto University which became the second largest university in Finland.

April 2011

Timo Honkela
Jorma Laaksonen

Organization

WSOM 2011 was held during June 13–15, 2011, organized by the Department of Computer and Information Science, Aalto University School of Science, and co-located with the ICANN 2011 conference.

Executive Committee

Honorary Chair	Teuvo Kohonen (Academy of Finland, Finland)
General Chair	Timo Honkela (Aalto University, Finland)
Program Chair	Jorma Laaksonen (Aalto University, Finland)
Local Chair	Olli Simula (Aalto University, Finland)
Publicity Chair	Jaakko Peltonen (Aalto University, Finland)

Steering Committee

Teuvo Kohonen	Academy of Finland, Finland
Marie Cottrell	Université Paris 1 Panthéon-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

Guilherme Barreto	Federal University of Ceará, Brazil
Yoonsuck Choe	Texas A&M University, USA
Jean-Claude Fort	Université de Toulouse, France
Tetsuo Furukawa	Kyushu Institute of Technology, Japan
Colin Fyfe	University of the West of Scotland, UK
Barbara Hammer	Bielefeld University, Germany
Samuel Kaski	Aalto University, Finland
Jorma Laaksonen, Chair	Aalto University, Finland
Krista Lagus	Aalto University, Finland
Amaury Lendasse	Aalto University, Finland
Ping Li	Pennsylvania State University, USA
Thomas Martinetz	University of Lübeck, Germany
Risto Miikkulainen	University of Texas at Austin, USA

Klaus Obermayer	Technische Universität Berlin, Germany
Jaakko Peltonen	Aalto University, Finland
Marina Resta	University of Genova, Italy
Udo Seiffert	Otto von Guericke University of Magdeburg, Germany
Olli Simula	Aalto University, Finland
Kadim Taşdemir	European Commission Joint Research Centre, Italy
Heizo Tokutaka	SOM Japan Inc., Japan
Carme Torras	Universitat Politècnica de Catalunya, Spain
Alfred Ultsch	Philipps-Universität Marburg, Germany
Marc Van Hulle	Katholieke Universiteit Leuven, Belgium
Michel Verleysen	Université Catholique de Louvain, Belgium
Thomas Villmann	University of Applied Sciences Mittweida, Germany
Lei Xu	The Chinese University of Hong Kong, Hong Kong

Additional Referees

Jaakko Hollmén	Aalto University, Finland
Timo Honkela	Aalto University, Finland
Markus Koskela	Aalto University, Finland
Antonio Neme	Aalto University, Finland
Mats Sjöberg	Aalto University, Finland
Mika Sulkava	Aalto University, Finland
Sami Virpioja	Aalto University, Finland

Table of Contents

Plenaries

Financial and Societal Applications

Theory and Methodology

Applications of Data Mining and Analysis

Language Processing and Document Analysis

Visualization and Image Processing

Topographic Mapping of Dissimilarity Data

Barbara Hammer[1], Andrej Gisbrecht[1], Alexander Hasenfuss[2], Bassam Mokbel[1], Frank-Michael Schleif[1], and Xibin Zhu[1]

[1] CITEC centre of excellence, Bielefeld University, Germany
[2] Computing Centre, TU Clausthal, Germany

Abstract. Topographic mapping offers a very flexible tool to inspect large quantities of high-dimensional data in an intuitive way. Often, electronic data are inherently non-Euclidean and modern data formats are connected to dedicated non-Euclidean dissimilarity measures for which classical topographic mapping cannot be used. We give an overview about extensions of topographic mapping to general dissimilarities by means of median or relational extensions. Further, we discuss efficient approximations to avoid the usually squared time complexity.

1 Introduction

Electronic data sets are increasing rapidly with respect to size and dimensionality, such that Kohonen's ingenious self organizing map (SOM) has lost none of its attractiveness as an intuitive data inspection tool: it allows humans to rapidly access large volumes of high dimensional data [20]. Besides its very simple and intuitive training technique, the SOM offers a large flexibility by providing simultaneous visualization and clustering based on the topographic map formation. As a consequence, application scenarios range from robotics and telecommunication up to web- and music-mining; further, the self-organizing map is a widely used technique in the emerging field of visual analytics because of its efficient and robust way to deal with large, high-dimensional data sets [19].

The classical SOM and counterparts derived from similar mathematical objectives such as the generative topographic mapping or neural gas [23,3] have been proposed to process Euclidean vectors in a fixed feature vector space. Often, electronic data have a dedicated format which cannot easily be converted to standard Euclidean feature vectors: biomedical data bases, for example, store biological sequence data, biological networks, scientific texts, textual experiment descriptions, functional data such as spectra, data incorporating temporal dependencies such as EEG, etc. It is not possible to represent such entries by means of conventional feature vectors without loss of information, many data being inherently discrete or compositional. Rather, experts access such data by means of dedicated comparison measures such as BLAST or FASTA for biological sequences, alignment techniques for biological networks, dynamic time warping for time series, etc. From an abstract point of view, dissimilarity measures or kernels which are suited for the pairwise comparison of abstract data types such as strings, trees, graphs, or functions are used.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 1–15, 2011.

Already almost 10 years ago, Kohonen proposed a very intuitive way to extend SOMs to discrete data characterized by dissimilarities only [21]: instead of mean prototype positions in a Euclidean vector space, neuron locations are restricted to data positions. The generalized median serves as a computational vehicle to adapt such restricted neurons according to given dissimilarity data. This principle can be extended to alternatives such as neural gas, and it can be substantiated by a mathematical derivative from a cost function such that convergence of the technique can be proved [8]. Depending on the characteristics of the data set, however, the positional restrictions can lead to a much worse representation of the data as compared to the capabilities of continuous updates which are possible in a Euclidean vector space.

As an alternative, specific dissimilarity measures can be linked to a nonlinear kernel mapping. Kernel versions of SOM have been proposed for example in the contribution [30] for online updates and [4] for batch adaptation; in both cases, the standard SOM adaptation which takes place in the high-dimensional feature space is done implicitly based on the kernel. Kernelization of SOM allows a smooth prototype adaptation in the feature space, but it has the drawback that it is often not applicable since many classical dissimilarity measures cannot be linked to a kernel. For such cases, so-called relational approaches offer an alternative [15]: prototypes are represented implicitly by means of a weighting scheme, and adaptation takes place based on pairwise dissimilarities of the data only. This principle has already been used in the context of fuzzy clustering [17]; in the past years, it has been successfully integrated into topographic maps such as SOM, neural gas, or the generative topographic mapping [15,14].

Both principles, median extensions of SOM or relational versions, have the drawback of squared time complexity due to their dependency on the full dissimilarity matrix. Since the computational costs of specialized dissimilarities such as alignment for strings or trees can be quite time consuming, the main computational bottleneck of the techniques is often given by the computation of the full dissimilarity matrix. For this reason, different approximation techniques have recently been proposed which rely on only a linear subset of the full dissimilarity matrix and which reduce the computational effort to an only linear one. Two particularly promising techniques are offered by the Nyström approximation, on the one hand, which can be transferred to dissimilarities as shown in [13]. On the other hand, if a computation of the dissimilarities can be done online, patch processing offers a very intuitive and easily parallelisable scheme which can even deal with non i.i.d. data distributions [1]. This way, efficient linear time processing schemes for topographic mapping of dissimilarity data arises.

In this contribution, we define topographic mapping based on cost functions first. Afterwards, we introduce two different principles to extend the techniques to dissimilarity data: median and relational clustering. Both methods can be substantiated by mathematical counterparts linking it to cost functions and pseudo-Euclidean space, respectively. We conclude with technologies which allow to speed the topographic mapping up to linear time complexity.

2 Topographic Mapping

Prototype based approaches represent data vectors $\boldsymbol{x} \in \mathbb{R}^n$ by means of prototypes $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N \in \mathbb{R}^n$ based on the standard squared Euclidean distance

$$d(\boldsymbol{x}, \boldsymbol{w}_i) = \|\boldsymbol{x} - \boldsymbol{w}_i\|^2 \tag{1}$$

The receptive field of prototype $\boldsymbol{w}_i$ is determined by the characteristic function

$$\chi_i(\boldsymbol{x}) = \begin{cases} 1 & \text{if } d(\boldsymbol{x}, \boldsymbol{w}_i) \leq d(\boldsymbol{x}, \boldsymbol{w}_j) \text{ for all } j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Given a finite set of data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$, the quantization error

$$E_{\mathrm{qe}} = \frac{1}{2} \sum_{i,j} \chi_i(\boldsymbol{x}_j) d(\boldsymbol{x}_j, \boldsymbol{w}_i) \tag{3}$$

offers one quality measure for a prototype-based representation of data. Popular learning schemes such as k-means clustering or vector quantization are directly based on this cost term, which is optimized by means of an online gradient technique (vector quantization), or a batch approach (k-means), respectively [9]. The cost function can be interpreted as a limit case of statistical data modeling by means of a mixture of Gaussians where the centers are located at prototype positions. Batch learning results as a limit case of an EM optimization scheme of the data log likelihood in this setting [2]. Although the quantization error constitutes one of the most popular measures to evaluate unsupervised clustering, it is often not sufficient in practical applications due to several aspects: it suffers from numerical problems due to the multi-modality of the cost function and its sensitivity to noise and outliers. In addition, further functionalities are often required in application scenarios such as the possibility to visualize the prototypes and to inspect relations between prototypes. Both problems are addressed by topographic mapping.

The Batch SOM. Topographic mapping integrates a neighborhood structure of the prototypes into the model. This way it achieves both, a better robustness with respect to local optima, outliers, and noise in the data as well as enhanced functionality due to the explicit neighborhood relations of the prototypes. In essence, topographic mapping takes place by matching a neighborhood topology of the prototypes and the topology which is inherent in the data distribution; as a consequence, the prototypes together with their neighborhood structure can be interpreted as compressed representation of the data set and its topological structure. Concrete topographic mapping technologies differ in the way how the neighborhood structure is defined.

Neural gas (NG) as proposed by Martinetz relies on a data optimum topology which is inferred directly from the data [23]. The popular SOM imposes a fixed predefined neighborhood defined by a regular lattice topology, typically a two dimensional lattice in Euclidean or hyperbolic space [20,26]. This way, not only

a neighborhood structure is inferred but it can also directly be visualized on the computer screen. Since data and lattice topology need not coincide, topological mismatches can occur unlike in NG. The original SOM does not possess a cost function in the continuous case and its mathematical investigation is quite demanding, see e.g. [18,20,7]. A slight variation of the definition of the receptive fields as compared to (2), however, enables the derivation from a cost function very similar to (3)

$$E_{\mathrm{SOM}} = \frac{1}{2} \sum_{i,j} \chi_i^*(\boldsymbol{x}_j) \sum_k \exp(-\mathrm{nd}(i,k)/\sigma^2) d(\boldsymbol{x}_j, \boldsymbol{w}_k) \tag{4}$$

where $\mathrm{nd}(i,j)$ refers to a priorly fixed neighborhood structure of the prototypes, e.g. their distance in a predefined two dimensional lattice, and the characteristic function of the receptive fields $\chi_i^*(\boldsymbol{x}_j)$, unlike (2), is measured via the averaged distances $\sum_k \exp(-\mathrm{nd}(i,k)/\sigma^2) d(\boldsymbol{x}_j, \boldsymbol{w}_k)$. Online adaptation iteratively adapts the winning prototype and its neighborhood towards a given data point, while batch adaptation iterates the following two computations

$$\text{compute } \chi_i^*(\boldsymbol{x}_j) \text{ for all } i, \tag{5}$$

$$\text{adapt } \boldsymbol{w}_k := \frac{\sum_{i,j} \chi_i^*(\boldsymbol{x}_j) \cdot \exp(-\mathrm{nd}(i,k)) \cdot \boldsymbol{x}_j}{\sum_{i,j} \chi_i^*(\boldsymbol{x}_j) \cdot \exp(-\mathrm{nd}(i,k))} \tag{6}$$

It has been shown in [6] that this procedure converges in a finite number of steps towards a local optimum of the cost function. The convergence is very fast such that a good initialization is necessary to avoid topological mismatches as pointed out in [10]. For this reason, typically, an initialization by means of the two main principal components takes place, and the neighborhood σ is annealed carefully during training.

The GTM. The generative topographic mapping (GTM) can be seen as a statistical counterpart of SOM which models data by a constraint mixture of Gaussians [3]. The centers are induced by lattice positions in a low dimensional latent space and mapped to the feature space by means of a smooth function, usually a generalized linear regression model. That means, prototypes are obtained as images of lattice points $\boldsymbol{v}_i$ in a two dimensional space $\boldsymbol{w}_i = f(\boldsymbol{v}_i) = \Phi(\boldsymbol{v}_i) \cdot \boldsymbol{W}$ with a matrix of fixed base functions Φ such as equally spaced Gaussians in two dimensions and a parameter matrix $\boldsymbol{W}$. Every prototype induces an isotropic Gaussian probability with variance β^{-1} which are combined in a mixture model using uniform prior over the modes. For training, the data log likelihood $\sum_j \ln \frac{1}{N} \cdot \sum_i \left(\frac{\beta}{2\pi}\right)^{n/2} \exp\left(-\frac{\beta}{2} d(\boldsymbol{x}_j, \boldsymbol{w}_i)\right)$ is optimized by means of an EM approach which yields to linear algebraic equations to determine the parameters $\boldsymbol{W}$ and β. As batch SOM, GTM requires a good initialization which is typically done by aligning the principal components of the data with the initial images of the lattice points. The smoothness of the mapping f, i.e. the number of base functions in Φ, determines the stiffness of the resulting topological mapping. Unlike SOM which focuses on the quantization error in the limit of

small neighborhood size, this stiffness accounts for a usually better visualization behavior of GTM, see e.g. Fig. 1. It can clearly be seen that GTM respects the overall shape of the data manifold while SOM pushes prototypes towards data centers, leading to local distortions. A better preservation of the manifold shape can also be obtained using ViSOM instead of SOM [29], albeit this technique is not substantiated by a global cost function such as GTM.

3 Median Clustering

Often, data are not given as vectors, rather pairwise dissimilarities $d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ of data points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are available. Thereby, the dissimilarity need not correspond to the Euclidean metric, and it is not clear whether data $\boldsymbol{x}_i$ can be represented as finite dimensional vectors at all. In the following, we refer to the dissimilarity matrix with entries d_{ij} as D. We assume that D has zero diagonal and that D is symmetric.

Median SOM. This situation causes problems for classical topographic mapping since a continuous adaptation of prototypes is no longer possible like in the Euclidean case. One solution has been proposed in [21]: prototype locations are restricted to the positions offered by data points, i.e. we enforce $\boldsymbol{w}_i \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$. In [21] a very intuitive heuristic how to determine prototype positions in this setting has been proposed based on the generalized median. As pointed out in [8], it is possible to derive a similar learning rule from the cost function of SOM (4): Like in batch SOM, optimization takes place iteratively with respect to the assignments of data to prototypes (5) and with respect to the prototype positions. The latter step does not allow an explicit algebraic formulation such as (6) because of the restriction of prototype positions; rather, prototypes are found by exhaustive search optimizing their contribution to the cost function:

$$\boldsymbol{w}_k = \mathrm{argmin}_{\boldsymbol{x}_l} \left\{ \sum_{i,j} \chi_i^*(\boldsymbol{x}_j) \exp(-\mathrm{nd}(i,k)/\sigma^2) d(\boldsymbol{x}_j, \boldsymbol{x}_l) \right\} \tag{7}$$

In the original proposal [21], the summation is restricted to the neighborhood, and possible candidates $\boldsymbol{x}_l$ are restricted to data points mapped to the vicinity of prototype $\boldsymbol{w}_k$. This can be seen as an efficient approximation of the above optimization in particular for small neighborhood range. The choice of (7) has the advantage that convergence of the technique in a finite number of steps can be guaranteed since the algorithm optimizes the cost function of SOM (4) for restricted prototype locations [8].

In complete analogy, batch neural gas can be extended to dissimilarity data by means of the generalized median and the respective cost function. For GTM, a transfer is not possible in general because it is not possible to define a smooth mapping from a continuous latent space to the discrete space of known data points characterized by pairwise dissimilarities.

Evaluation. One important drawback of median approaches is given by the computational complexity: compared to linear time complexity for standard Euclidean topographic mapping, the effort increases to squared complexity due to the necessity of an exhaustive search for every optimization step of the prototypes (7). This can be partially accelerated by means of different techniques such as block summing and branch and bound techniques (see e.g. [16]); due to the dependency of the cost function on all pairwise dissimilarities, every exact technique must inherently be quadratic, however.

Another problematic issue concerns the initialization of median SOM, and its limited capability of smooth updates as compared to standard Euclidean versions. Unlike the Euclidean SOM, an initialization of the map in the direction of the main principal components is hardly possible since only a discrete data space is at our disposal. Due to the rapid convergence of batch techniques, this causes the severe risk that the topographic mapping gets stuck in local optima. Further, as compared to Euclidean settings, less flexibility of the prototypes is available which can cause worse solutions as compared to continuous settings. Tab. 1 shows the results of the techniques for the chromosome data set, a benchmark from cytogenetics [22]. It consists of 4200 images of chromosomes from 22 classes. Since the overall shape is the relevant feature to discriminate different chromosomes, the images cannot easily be compared based on Euclidean distance. Alternatively, images are described by strings which characterize the thickness of the chromosomes which are oriented according to their lengths. Images are compared by aligning these strings, i.e. a non-Euclidean dissimilarity results. Since a labeling is available, the number of the chromosomes, an evaluation of the results can be done by posterior labeling of the prototypes according to their receptive field. The test set accuracy which results from a repeated cross-validation is reported.

Obviously, median SOM yields worse results as compared to continuous variants such as relational SOM, which we will explain in the next section. Further, it can be observed that the topological constraint of SOM by the priorly fixed lattice leads to worse results as compared to NG. Interestingly, the accuracy of median techniques is not caused by the restricted representation ability of median clustering, rather numerical problems occur due to the restricted flexibility while optimizing the cost function. This observation is substantiated by the result of affinity propagation (AP) as shown in Tab. 1. AP constitutes an exemplar based clustering scheme which is derived from the quantization error by means of a representation of this cost function as factor graph, and an approximate optimization by means of the max-sum algorithm [12]. Unlike median SOM or median NG, an inherently smooth adaptation process which adapts the likelihood of the data points of becoming an exemplar takes place for AP, resulting in an increased classification accuracy albeit the final solution is represented in terms of data exemplars just as median clustering. AP, however, does not involve any topology such that no topographic mapping is obtained.

Table 1. Classification accuracy on the test set obtained by repeated cross-validation and different clustering techniques on the chromosome data, the numbers refer to (number of patches / k for k-approximation) for patch processing and (fraction of landmarks) for the Nyström approximation

median SOM	median NG	AP	relational SOM	relational NG	relational GTM
0.72	0.82	0.9	0.92	0.93	0.92
patch AP	patch RNG (40/10)	Ny RNG (0.01)	patch RGTM (10/5)	Ny RGTM (0.01)	Ny RGTM (0.1)
0.76	0.88	0.93	0.87	0.88	0.55

4 Relational Clustering

As discussed above, the discrete nature of median clustering causes a severe risk to get trapped in local optima of the cost function. Hence the question arises whether a continuous adaptation of prototypes is possible also for general dissimilarity data. A general approach to extend prototype-based clustering schemes to general dissimilarities has been proposed in [17] in the context of fuzzy clustering, and it has recently been extended in [15,14] to batch SOM, batch NG, and GTM.

Assume that the dissimilarities d_{ij} stem from unknown data in an unknown high dimensional feature vector space, i.e. $d_{ij} = \|\Phi(\boldsymbol{x}_i) - \Phi(\boldsymbol{x}_j)\|^2$ for some feature map Φ. Assume that prototypes can be expressed as linear combinations $\boldsymbol{w}_i = \sum_j \alpha_{ij}\Phi(\boldsymbol{x}_j)$ with $\sum_j \alpha_{ij} = 1$. Then, distances can be computed implicitly

$$d(\boldsymbol{w}_i, \boldsymbol{x}_j) = [D\boldsymbol{\alpha}_i]_j - \frac{1}{2} \cdot \boldsymbol{\alpha}_i^t D \boldsymbol{\alpha}_i \tag{8}$$

It has been shown in [15] that this equation also holds if an arbitrary symmetric bilinear form induces dissimilarities in the feature space rather than the squared Euclidean distance.

Relational SOM. This observation offers a way to directly transfer batch SOM and batch NG to a general symmetric dissimilarity matrix D. As explained e.g. in [15], there always exists a vector space together with a symmetric bilinear form which gives rise to the given dissimilarity matrix. This vector space need not be Euclidean since some eigenvalues associated to the bilinear form might be negative or zero. Commonly, this is referred to as pseudo-Euclidean space where the eigenvectors associated to negative eigenvalues serve as a correction to the otherwise Euclidean space. For this vector space, batch NG or SOM can be applied directly in the vector space, and using (8), it can be applied implicitly without knowing the embedding, because of two key ingredients:

1. an implicit representation of prototype $\boldsymbol{w}_i$ in terms of coefficient vectors $\boldsymbol{\alpha}_i$,
2. Equation (8) to compute the distance in between a data point and a prototype.

Since prototypes as computed by batch NG or SOM can be written as convex combination of data points, and since the update of a prototype depends on the distance only and it decomposes into updates of the coefficients, NG and SOM can be immediately transferred to this setting. So-called relational SOM (RSOM) is given by the iteration of the following steps:

$$\text{compute } d(\boldsymbol{w}_i, \boldsymbol{x}_j) \text{ based on Equation (8)} \tag{9}$$

$$\text{compute } \chi_i^*(\boldsymbol{x}_j) \text{ based on these values} \tag{10}$$

$$\text{adapt } [\boldsymbol{\alpha}_k]_j := \frac{\sum_i \chi_i^*(\boldsymbol{x}_j) \cdot \exp(-\mathrm{nd}(i,k))}{\sum_{i,j} \chi_i^*(\boldsymbol{x}_j) \cdot \exp(-\mathrm{nd}(i,k))} \tag{11}$$

Note that this procedure is equivalent to an implicit application of SOM in the pseudo-Euclidean embedding space. It is independent of the concrete embedding and gives the same results if an alternative embedding is used. Further, it is equivalent to standard SOM if a Euclidean embedding of data exists. For general dissimilarities, it constitutes a reasonable extension of SOM to the general case with continuous updates of prototypes.

This procedure, however, has one drawback: although it constitutes an exact implementation of SOM in pseudo-Euclidean space, it is no longer clear that the procedure offers an optimization of the corresponding SOM cost function in the embedding space. This is due to the fact that batch SOM itself does not necessarily optimize the cost function in non-Euclidean space; rather, the mean value might constitute a saddle point of the quantization error if data are non-Euclidean. In fact, the quantization error of one receptive field is no longer a convex cost function in the general setting and its optimization is NP hard [27]. Regarding this complexity, the choice (11) can be seen as a reasonable efficient compromise which optimizes the data representation within a receptive field with respect to the positive eigendirections of the underlying bilinear form. See the work [15] for more discussions and experiments concerning this issue. It turns out that the choice (11) hardly deteriorates the value of the cost function in practical applications.

Relational GTM. In a similar way, NG can be directly extended to arbitrary dissimilarity data, yielding relational NG (RNG). Similarly, GTM can be extended based on the key observation (8) since also for GTM, prototypes can be chosen as linear combinations of data with coefficients summing up to one. Using Lagrangian functions, it can be proved that this is automatically fulfilled for standard GTM [14]. To realize the approach efficiently, the low dimensional latent space is directly mapped to the space of coefficients, see [14]. For relational GTM (RGTM), however, an interpretation by means of a stochastic model is not always clear due to the fact that distances can become negative in pseudo-Euclidean space. For such settings, an interpretation as density values is not obvious; in addition, numerical problems can occur. In the publication [14], this setting is investigated and the feasibility of the approach is demonstrated in several real life examples.

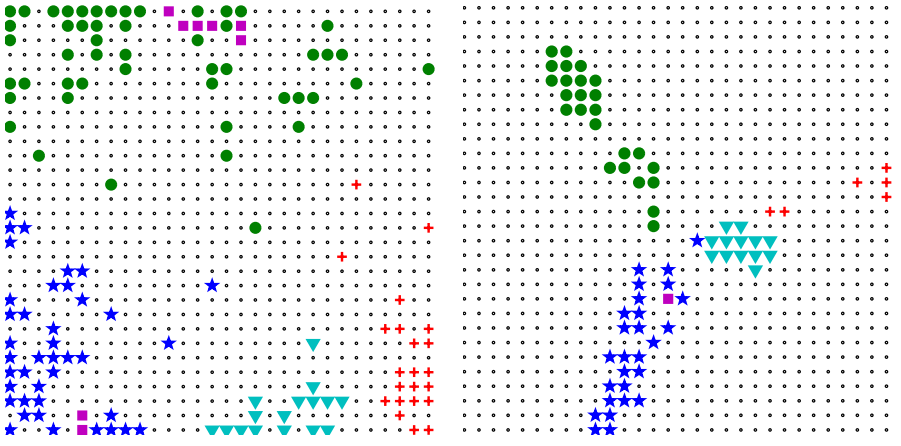


Fig. 1. Visualization of the protein data set incorporating 226 proteins in 5 classes using RSOM (left) and RGTM (right); labels are determined by posterior labeling according to majority vote of the receptive fields

Evaluation. As for the standard Euclidean counterpart, relational GTM tends to display data in a way more suitable for direct data visualization, since less distortions take place if a reasonable number of base functions is used. One example is shown in Fig. 1. Here protein sequences from different families are compared using an evolutionary distance [24]. In total, 226 globin proteins with 5 different classes are depicted. In both visualizations, the clusters separate according to the a priori known classes. For the RSOM, the prototypes cover the data space with many data being located at the map boundaries, while RGTM widely keeps the internal arrangement due to its stiffness.

In Tab. 1, relational topographic mapping is compared to median approaches, evaluating the techniques in a repeated cross-validation considering the classification accuracy on the test set for the chromosomes benchmark data. As can be seen from the results, the larger flexibility offered by continuous prototype adaptation in relational topographic mapping leads to an improvement of almost 20% for SOM and almost 10% for NG, arriving at a slightly better value than AP. This fact can be explained by the much simpler numerical optimization of the techniques if a more flexible continuous prototype adaptation is possible instead of only discrete steps. Albeit convergence of relational topographic mapping is not strictly guaranteed (since saddle points might be chosen instead of local optima in case of negative eigenvalues of the corresponding pseudo-Euclidean embedding), divergence never occurred in practical problems.

5 Efficient Approximations

Both median and relational clustering suffer from a quadratic time complexity as compared to linear complexity for their vectorial counterparts. In addition,

relational clustering requires linear space complexity since it stores prototypes in terms of coefficient vectors representing the relevance of every data point for the respective prototype. This fact makes the interpretability of the resulting map difficult since it is no longer easily possible to inspect prototypes in the same way as data points. Further, the quadratic time complexity makes the methods infeasible already for medium sized data sets. Different heuristics have recently been proposed in this context to speed up median and relational clustering.

Patch approximation. Patch processing constitutes a very simple approach to derive a finite space linear time method based on a prototype based technique. It has been proposed in [1] in the context of the application of NG for streaming data, and, interestingly, it even gives good results if data are not i.i.d. The main idea is to process data consecutively in patches of fixed size. The prototypes counted with multiplicities according to their receptive fields represent all already seen data, and they are included as regular points counted with multiplicities in the next patch. This way, all information is taken into account either directly or in compressed form in the succeeding clustering steps.

If transferred to dissimilarity data, this approach refers to a linear subset of the full dissimilarity matrix only: only those dissimilarities are necessary which correspond to a pair of data in the same patch, further, distances of prototypes representing the previous points and data points in a patch are used. In consequence, an only linear subpart of the full dissimilarity matrix is used this way. Since it is not known a prior which prototypes are used for the topographic mapping, however, the method requires that dissimilarities can be computed instantaneously during the processing. For real life applications this assumption is quite reasonable; e.g. biological sequences can be directly stored and accessed in a data base; their pairwise comparisons can be done on demand using sequence alignment.

Median clustering can directly be extended in a similar way. Unfortunately, such as median topographic mapping itself, it suffers from local optima due to the limited prototype flexibility. In [31], a corresponding extension of affinity propagation is proposed. Due to problems of AP to deal with multiple points, however, the result is worse as compared to AP for the full data set, see Tab. 1.

Patch approximation for relational approaches. For relational clustering a direct extension of the patch approach is not possible because prototypes are presented indirectly by referring to the data points. This way, eventually, every prototype refers to all data, i.e. all pairwise dissimilarities have to be known to compute distances in between prototypes and data. In the approach [15], a simple though efficient heuristic is proposed. A prototype is approximated by a fixed number of data points k which are closest to the prototype. These data points are taken to represent the already seen information in compressed form for a new patch. Depending on the value k and the number of patches, a different approximation quality is obtained. Tab. 1 displays the result of relational NG and relational SOM when using patch clustering. As can be seen from the results, a mild degradation of the accuracy (less than 5%) can be observed due to the

information loss. The method turns out to be rather robust with respect to the choice of the approximation quality k and the patch size. Further, it can deal with data which are not accessible in an i.i.d. fashion.

Nyström approximation. As an alternative, the Nyström approximation has been introduced as a standard method to approximate a kernel matrix in [28]. It can be transferred to dissimilarities as presented in [13]. The basic principle is to pick M representative landmarks in the data set which give rise to the rectangular sub-matrix $D_{M,m}$ of dissimilarities of data points and landmarks. This matrix is of linear size, assuming M is fixed. It can be shown (see e.g. [13]) that the full matrix can be approximated in an optimum way in the form

$$D \approx D_{M,m}^t D_{M,M}^{-1} D_{M,m} \tag{12}$$

where $D_{M,M}$ is induced by an $M \times M$ eigenproblem depending on the rectangular sub-matrix of D. Its computation is $\mathcal{O}(M^3)$ instead of $\mathcal{O}(m^2)$ for the full matrix D. The approximation (12) is exact if M corresponds to the rank of D. It is possible to integrate the approximation (12) in such a way into the distance computation (8) such that the overall effort is only linear with respect to m. This way, a linear approximation technique for relational clustering results. See [13] for detailed formulas.

Evaluation. The quality of the result depends very much on the approximation quality of (12), i.e. landmarks should induce a representative dissimilarity matrix. In consequence, the technique is not suited for data which are not i.i.d. For representative landmarks, however, the result can be quite good, as can be seen in Tab. 1: an approximation of the full dissimilarity matrix using only 1% of the data as landmarks deteriorates the result not at all for RNG, and by only 4% for RGTM. Interestingly, the result can severely be influenced by the choice of the landmarks: for RGTM, if we pick 10% of the data as landmarks, the classification accuracy decreases by nearly 40%. This can be associated to the fact that a highly skewed representation of the dissimilarity matrix is obtained in this case due to the characteristic of the eigenvalue profile of the corresponding dissimilarity matrix. Unlike patch processing, it is fixed a priori which parts of the dissimilarity matrix are relevant for the Nyström method. In consequence, this technique is suited if the dissimilarity matrix D is available a priori, but access to entries of D and the topographic mapping are costly.

Computational effort. As a final demonstration of the feasibility of the approach to deal with realistic data sets, we show the result of a visualization experiment in line with the early work of Kohonen for median clustering [21]: RGTM is used to visualize a portion of the Swiss-Prot data base containing protein sequences [5]. Swiss-Prot constitutes a high quality data base of known protein structures which are manually curated. Together with automatically generated further data sets of sequences, it forms a part of one of the most popular publicly available data bases of known protein sequences. We select a subset of

roughly 11000 sequences according to 32 different functional classes characterized by functional labels as stored in the Prosite data base (see the reference [11] for the ExPASy proteomics server where protein data bases as well as functions e.g. for pairwise alignment are provided). Typically, protein sequences are compared using alignment techniques. We rely on the FASTA algorithm (with default parameters) which offers a linear time approximation of exact sequence alignment by linear programming (the latter has squared complexity) to compute sequence similarity.

This example consists a medium sized application of the technique to a real life data set – we are working on an extension to a larger portion of the Swiss-Prot data set. Since data base retrieval and data inspection is very important in this context, it would be interesting to obtain a tool to visualize and inspect large portions of the data bases, and to obtain an efficient retrieval method e.g. by means of a compressed representation by prototypes. Using the RGTM technique, a reasonable visualization and arrangement of the data can be obtained. RGTM with Nyström approximation with 100 landmarks yields the visualization as shown in Fig. 2. Prototypes are posteriorly labeled by majority vote based on the Prosite labeling of the data in the receptive field. 19 out of 32 classes are visible on the map this way; one can clearly see that the largest classes arrange in a topology preserving fashion on the map, i.e. an inspection of the relative arrangement of these classes according to their similarity becomes possible.

This data set is of medium size, such that the speedup in time and space efficiency of the Nyström approximation becomes apparent. In the given example, assuming double precision, about 500 Megabyte are necessary to store the full dissimilarity matrix as compared to about 4.5 Megabyte for the dissimilarities referred to by the Nyström approximation. Since the number of necessary landmarks does not depend on the number of data but the underlying rank of the dissimilarity matrix, it can be expected that the same number of landmarks gives reasonable results also for larger data sets. In consequence, using 100 landmarks and assuming a standard RAM of 12 Gigabyte, this would allow to store the required dissimilarities of almost 30 million data points as compared to only 30 thousand if the full dissimilarity matrix is required. For the considered data set, the computational speedup is determined by two factors: the speedup due to the smaller number of necessary dissimilarities which are computed, and the speedup due to the smaller effort to compute the responsibilities in RGTM. Both computations are reduced from squared time complexity to a linear one.

For the considered data set, the speedup to compute the required part of the dissimilarity matrix accounts for a factor 55, while training RGTM is about a factor 25 faster for the Nyström technique as compared to the exact version. In numbers, referring to a standard desktop PC (Intel Xeon QuadCore 2.5 GHz), this reduces the computation time for the dissimilarity matrix from more than eight days to less than two hours, and the training time for RGTM from about one day to less than an hour. Extrapolating this behavior and assuming the same number of landmarks, training RGTM on a standard desktop PC would be possible for at most 30 thousand data points leading to a training time of

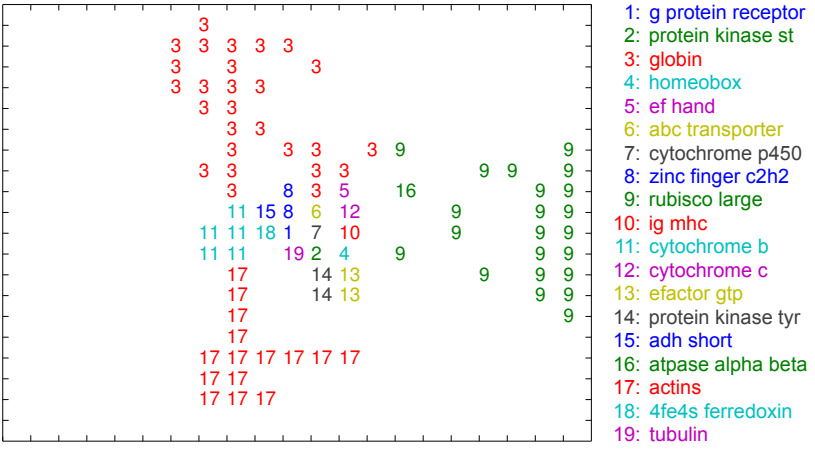


Fig. 2. Around 10000 protein sequences compared by pairwise alignments are depicted on a RGTM trained with the Nyström approximation and 100 landmarks. Posterior labeling displays 19 out of the 32 classes defined by Prosite for this data set in a topology preserving manner.

more than a week (disregarding the computation of the dissimilarity matrix which would need more than two month on a single PC) and it would work at the limit of the available RAM, while the Nyström approximation could deal with two million data points in the same time and it would use only a fraction of the available RAM.

6 Conclusions

We have presented an overview of topographic mapping of dissimilarity data by means of median and relational clustering. Interestingly, popular techniques such as SOM, NG, or GTM can be extended this way, opening the way towards modern data analysis tools for general data formats described in terms of pairwise dissimilarities only. For large data sets, the squared complexity caused by the size of the dissimilarity matrix makes the techniques infeasible already for medium sized data sets. We have presented two techniques to arrive at efficient linear time approximations which offer state of the art linear techniques to deal with large data sets.

Acknowledgment

This work was supported by the "German Science Foundation (DFG)" under grant number HA-2719/4-1. Further, financial support from the Cluster of Excellence 277 Cognitive Interaction Technology funded in the framework of the German Excellence Initiative is gratefully acknowledged.

References

1. Alex, N., Hasenfuss, A., Hammer, B.: Patch clustering for massive data sets. Neurocomputing 72(7-9), 1455–1469 (2009)
2. Bishop, C.: Pattern Recognition and Machine Learning. Springer, Heidelberg (2007)
3. Bishop, C.M., Williams, C.K.I.: GTM: The generative topographic mapping. Neural Computation 10, 215–234 (1998)
4. Boulet, R., Jouve, B., Rossi, F., Villa-Vialaneix, N.: Batch kernel SOM and related Laplacian methods for social network analysis. Neurocomputing 71(7-9), 1257–1273 (2008)
5. Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.-C., Estreicher, A., Gasteiger, E., Martin, M.J., Michoud, K., O'Donovan, C., Phan, I., Pilbout, S., Schneider, M.: The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. Nucleic Acids Research 31, 365–370 (2003)
6. Bottou, L., Bengio, Y.: Convergence properties of the k-means algorithm. In: Tesauro, G., Touretzky, D.S., Leen, T.K. (eds.) NIPS 1994, pp. 585–592. MIT, Cambridge (1995)
7. Cottrell, M., Fort, J.C., Pagès, G.: Theoretical aspects of the SOM algorithm. Neurocomputing 21, 119–138 (1999)
8. Cottrell, M., Hammer, B., Hasenfuss, A., Villmann, T.: Batch and median neural gas. Neural Networks 19, 762–771 (2006)
9. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. John Wiley & Sons, New York (2001)
10. Fort, J.-C., Letrémy, P., Cottrell, M.: Advantages and drawbacks of the Batch Kohonen algorithm. In: Verleysen, M. (ed.) ESANN 2002, D Facto, pp. 223–230 (2002)
11. Gasteiger, E., Gattiker, A., Hoogland, C., Ivanyi, I., Appel, R.D., Bairoch, A.: ExPASy: the proteomics server for in-depth protein knowledge and analysis. Nucleic Acids Res. 31, 3784–3788 (2003)
12. Frey, B.J., Dueck, D.: Clustering by passing messages between data points. Science 315, 972–976 (2007)
13. Gisbrecht, A., Mokbel, B., Hammer, B.: The Nystrom approximation for relational generative topographic mappings. In: NIPS Workshop on Challenges of Data Visualization (2010)
14. Gisbrecht, A., Mokbel, B., Hammer, B.: Relational generative topographic map. Neurocomputing 74, 1359–1371 (2011)
15. Hammer, B., Hasenfuss, A.: Topographic mapping of large dissimilarity datasets. Neural Computation 22(9), 2229–2284 (2010)
16. Hammer, B., Hasenfuss, A., Rossi, F.: Median topographic maps for biological data sets. In: Biehl, M., Hammer, B., Verleysen, M., Villmann, T. (eds.) Similarity-Based Clustering. LNCS, vol. 5400, pp. 92–117. Springer, Heidelberg (2009)
17. Hathaway, R.J., Bezdek, J.C.: Nerf c-means: Non-Euclidean relational fuzzy clustering. Pattern Recognition 27(3), 429–437 (1994)
18. Heskes, T.: Self-organizing maps, vector quantization, and mixture modeling. IEEE Transactions on Neural Networks 12, 1299–1305 (2001)
19. Keim, D.A., Mansmann, F., Schneidewind, J., Thomas, J., Ziegler, H.: Visual analytics: Scope and challenges. In: Simoff, S., Boehlen, M.H., Mazeika, A. (eds.) Visual Data Mining: Theory, Techniques and Tools for Visual Analytics. LNCS, Springer, Heidelberg (2008)

20. Kohonen, T. (ed.): Self-Organizing Maps, 3rd edn. Springer, New York (2001)
21. Kohonen, T., Somervuo, P.: How to make large self-organizing maps for nonvectorial data. Neural Networks 15, 945–952 (2002)
22. Lundsteen, C., J-Phillip, Granum, E.: Quantitative analysis of 6985 digitized trypsin g-banded human metaphase chromosomes. Clinical Genetics 18, 355–370 (1980)
23. 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), 558–569 (1993)
24. Mevissen, H., Vingron, M.: Quantifying the local reliability of a sequence alignment. Protein Engineering 9, 127–132 (1996)
25. Neuhaus, M., Bunke, H.: Edit distance based kernel functions for structural pattern classification. Pattern Recognition 39(10), 1852–1863 (2006)
26. Ontrup, J., Ritter, H.: Hyperbolic self-organizing maps for semantic navigation. In: Dietterich, T., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems, vol. 14, pp. 1417–1424. MIT Press, Cambridge (2001)
27. Pardalos, P.M., Vavasis, S.A.: Quadratic programming with one negative eigenvalue is NP hard. Journal of Global Optimization 1, 15–22 (1991)
28. Williams, C., Seeger, M.: Using the Nyström method to speed up kernel machines. In: Advances in Neural Information Processing Systems, vol. 13, pp. 682–688. MIT Press, Cambridge (2001)
29. Yin, H.: ViSOM - A novel method for multivariate data projection and structure visualisation. IEEE Trans. on Neural Networks 13(1), 237–243 (2002)
30. Yin, H.: On the equivalence between kernel self-organising maps and self-organising mixture density networks. Neural Networks 19(6-7), 780–784 (2006)
31. Zhu, X., Hammer, B.: Patch affinity propagation. In: European Symposium on Artificial Neural Networks (to appear, 2011)

Contextually Self-Organized Maps of Chinese Words

Teuvo Kohonen[1] and Hongbing Xing[2]

[1] Aalto University, Espoo, Finland
teuvo.kohonen@tkk.fi
[2] Beijing Language and Culture University, Beijing, China

Abstract. Contextual SOMs of Chinese words have been constructed in this work. Differing from previous approaches, in which individual words were mapped onto the SOM, in this work histograms of various word classes or otherwise defined subsets of words were formed on the SOM array. It was found that the words are not only clustered according to the word classes, but joint or overlapping clusters of words from different classes can also be formed according to the role of the words as sentence constituents. A further new effect was found. When the histograms were formed using test words restricted to certain intervals of word frequencies, the histograms were found to depend on the frequency, and the corresponding partial clusters were often very compact.

1 The Method

The *contextual SOMs* [1], [2], [3] are used to represent relationships between *local contexts* (groups of contiguous words) in text corpora, believed to reflect semantic properties of the words. A local context relates to and is labeled by its central word, called the *target word*. It has been found earlier that the SOM can be used to map *words* linguistically in such a way that the target words of different word classes are mapped into separate areas on the SOM on the basis of the local contexts in which they occur.

The local context around a particular target word can be defined in different ways. In early works it was made to consist of the target word itself, indexed by its position i in the corpus, and of the preceding and the subsequent word to it, respectively. In this work, in order to take more contextual information into account, the contexts were defined to consist of *five successive words*. In computation they were represented by the coding vectors $\mathbf{r}_{i-2}$, $\mathbf{r}_{i-1}$, $\mathbf{r}_i$, $\mathbf{r}_{i+1}$, and $\mathbf{r}_{i+2}$, respectively.

In order to minimize the effect of the word forms on the context structures, and to concentrate on the pure word patterns, i.e., combinations of the words, without paying attention to the writing forms, one ought to select *representations* for the words that are mutually as uncorrelated as possible. To that end, the coding vectors can be defined, e.g., as high-dimensional Euclidean vectors with normally distributed random elements. A typical dimensionality of these vectors is on the order of a few hundred. In this way, the representation vectors of

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 16–29, 2011.

any two words are approximately orthogonal and can be regarded as almost uncorrelated.

For a *statistical* analysis, the so-called *averaged contextual feature* $\mathbf{x}(w)$ for each unique word w in the text corpus can be defined as the vector

$$\mathbf{x}(w) = \mathrm{avg}_{i(w)}([\mathbf{r}_{i-2} \;\; \mathbf{r}_{i-1} \;\; \mathbf{r}_i \;\; \mathbf{r}_{i+1} \;\; \mathbf{r}_{i+2}]) \;\; , \tag{1}$$

where $\mathrm{avg}_{i(w)}$ means the average over all positions i in the text corpus, on the condition that the contextual feature relating to position i belongs to word w (i.e., on the condition that $\mathbf{r}_i$ is the random-vector representation of word w).

When constructing a contextual SOM, the averaged contextual feature vectors are used as the training data.

Differing from previous approaches, in which individual *test words* were mapped onto the SOM, in this work *histograms of various test word classes* or otherwise defined subsets of words will be formed over the SOM array. Thus, the testing of the SOM, i.e., the mapping of selected subsets of target words onto the SOM is carried out using similarly defined averaged feature vectors as input vectors, but averaging the input vectors only over the words w of a particular subset (such as general adjectives) or words that occur in the corpus only a specified number of times.

One particular problem encountered in this simple context analysis is that the words in most languages are inflected, and in languages such as Latin, Japanese, Hungarian, Finnish, etc,. the linguistic roles of the words are also indicated by many kinds of endings. One simple method is to treat every word form as a different word. Another method would be to reduce each word to its base form or word stem, whereby, however, some semantic information is lost.

Nonetheless there also exist languages such as Chinese, where the words are not inflected at all, and which would then be ideal for the context experiments. Since nowadays there are available large Chinese text corpora that are provided with linguistic analysis of the words used in them, it was possible to construct the contextual SOMs automatically on the basis of this information only [4], [5].

The text corpus used in this work, called the MCRC (Modern Chinese Research Corpus) [6] is an electronic collection of text material from newspapers, novels, magazines, TV shows, folktales, and other text material from modern Chinese media. In our experiment it contained 1,524,121 words provided with classification of the words into 114 classes (of which 88 were real linguistic classes, while the rest consisted of punctuation marks and nonlinguistic symbols). This corpus was prepared by one of the authors (Hongbing Xing).

A further notice is due. In order to utilize the information of the contexts maximally, only *pure contexts* (which did not contain any punctuation marks or nonlinguistic symbols) were accepted to these experiments. In this way, however, a substantial portion of the text corpus was left out of the experiments. Notice that if the target word has a distance of less than five words from these specific symbols, the five-word contexts could not be formed. Nonetheless the original corpus was so large that the remaining amount of text (488,878 words) was still believed to produce statistically significant results.

The size of the original lexicon used in this work was 48,191. The number of words actually used, restricting to pure contexts only, was 27,090.

The patches of the SOM array were selected as hexagonal, and the size of the array was relatively small, 40 by 50, in order to save memory but still to be able to discern the cluster structures on it.

In earlier works the dimensionalities of all of the random-code vectors of the words were always taken as equal. A new idea in this work was to select the dimensionality as a function of the relative position of the word within the local context. The dimensionality of the target vector $\mathbf{r}_i$ was selected as 50. The dimensionalities of $\mathbf{r}_{i-1}$ and $\mathbf{r}_{i+1}$ were taken equal to 200, and those of $\mathbf{r}_{i-2}$ and $\mathbf{r}_{i+2}$ equal to 100, respectively. In this way, the different words within the context have different statistical weights in the matching of the input vector with the SOM weight vectors. The above dimensionalities, based on many experiments, were chosen experimentally to roughly maximize the clustering accuracy, under the restriction that the total dimensionality of the feature vectors $\mathbf{x}(w)$ could still fit to the Matlab programs, especially to the SOM Toolbox used in the computations.

In order to write a great number of variable scripts for this rather large experiment, the Matlab SOM Toolbox [7] was used. However, the calibration of the numerous SOMs was based on dot-product matching, for which both the source data and the SOM vectors (prototypes) were normalized.

The *batch training* procedure of the Matlab SOM Toolbox was applied. The neighborhood function used in it was Gaussian. First, a coarse training phase, consisting of 20 training cycles was used. During it, the effective radius parameter of the neighborhood function decreased linearly from 6 to 0.5. The topological ordering occurred during this phase. After it, a fine tuning phase was applied. During it, the effective neighborhood radius stayed constant, equal to 0.5. It has been shown recently [8] that if this kind of learning with constant neighborhood is used, the batch training algorithm will usually converge in a finite number of cycles. In the present application this kind of fine tuning was continued until no changes in the map vectors took place in further training. This occurred in about 70 cycles.

2 Preliminary Clustering Results

In Fig. 1 we have the histograms of four main linguistic classes of the words in the MCRC corpus (restricting to local contexts that do not contain any punctuation marks or other nonlinguistic symbols).

Some clusters may look rather broad, but one has to realize three facts: 1. Due to competitive learning, the SOM is always trying to utilize the whole array optimally, so any cluster that contains a large number of elements will look wide. 2. As will be seen later, the diffuse zones usually consist of much narrower partial clusters. 3. It will further be shown in this work that the clusters of some word classes also depend on the *frequencies of the words* they contain. If one would select to the vocabulary only words that, e.g., exceed a certain frequency limit, one would obtain much sharper clusters.

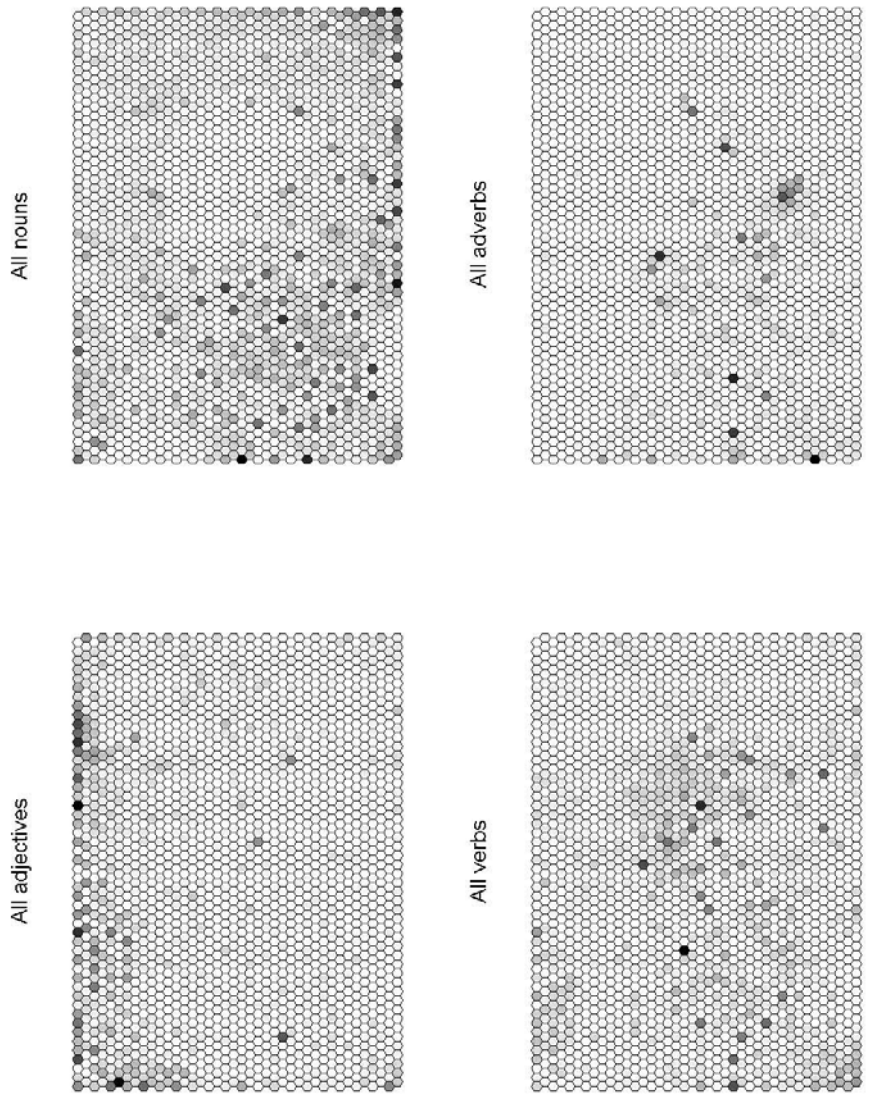


Fig. 1. Histograms of all adjectives, nouns, verbs, and adverbs in the MCRC

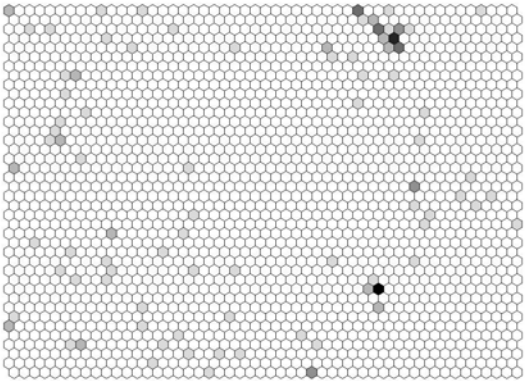


Fig. 2. Clustering of all attributive pronouns

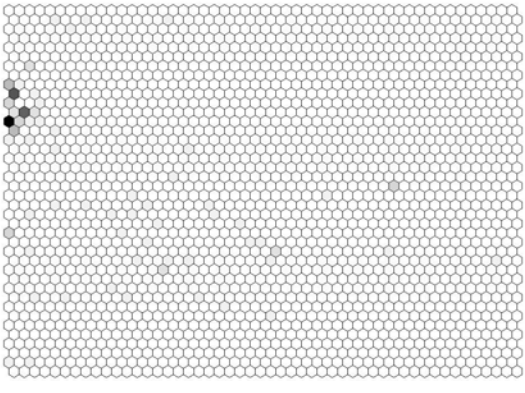


Fig. 3. Clustering of all adverbial idioms

In Fig. 2 and Fig. 3 we show two smaller clusters of specific word classes that are located close to the area of the adjectives. Consider first the cluster of attributive pronouns. If the words, as believed, are clustered in the contextual SOM according to their *role as sentence constituents*, then the adjectives that coincide with the attributive pronouns obviously represent attributive adjectives. On the other hand, the adjectives that coincide with the cluster of the adverbial idioms apparently have an adverbial nature, respectively.

3 Effect of Word Frequency on Clustering

A new effect found in this study is that when the histograms are formed using words restricted to certain *intervals of word frequencies*, they will depend on the frequency and be more compact.

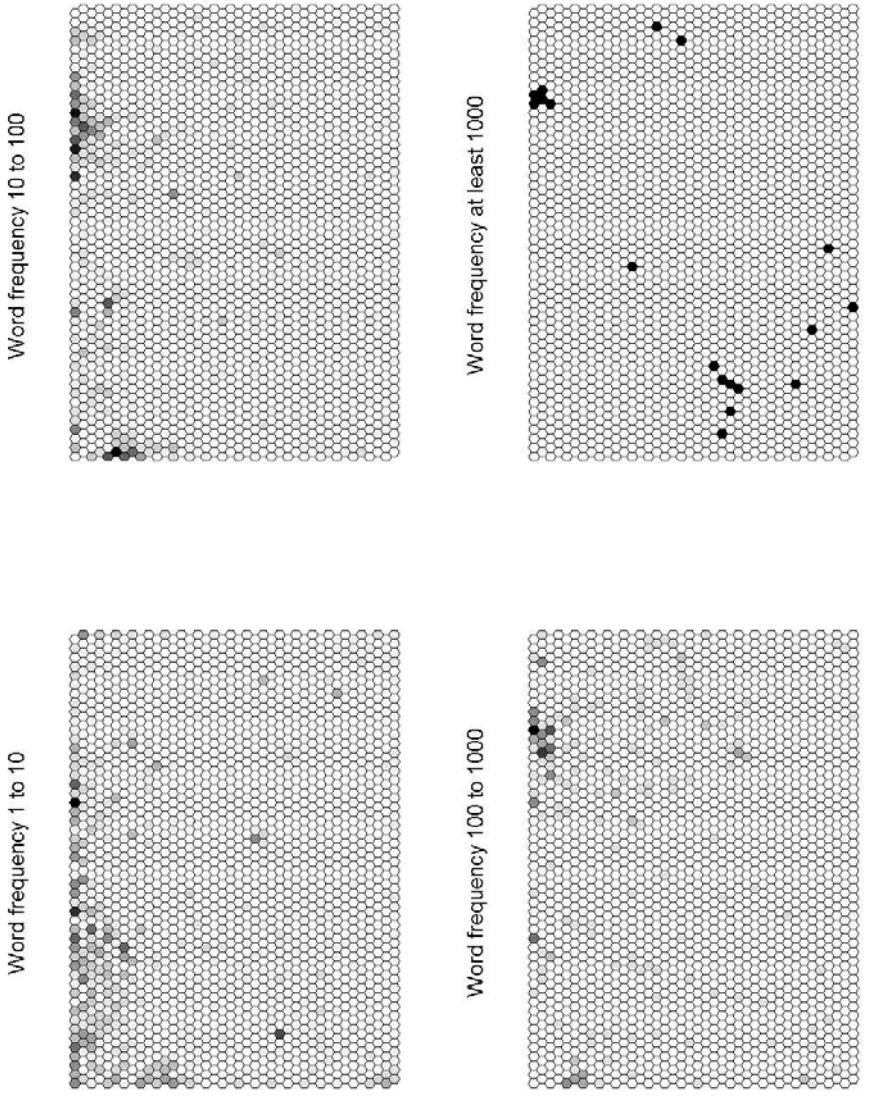


Fig. 4. Histograms of general adjectives, as a function of word frequency

3.1 General Adjectives

First we show the frequency effect by four histograms of general adjectives. In Fig. 1, all of the adjectives were mapped onto the top of the SOM into a relatively narrow zone, while one could discern certain substructures in their distribution. In Fig. 4 we now have four subplots (with word frequencies of 1 to 10, 10 to 100, 100 to 1000, and 1000 or more in the text corpus, respectively). One can clearly see that the linguistic nature of the subsets of words selected to each histogram changes gradually. Some of the adjectives in all frequency ranges are clustered very tightly at the left side of the map and thus seem to have an *adverbial nature*, whereas the more frequently used adjectives at the top-right obviously act in the role of an *attribute*. The rest of the adjectives located along the top of the map have other, variable linguistic roles.

One particular caution may be necessary, when looking at the graphics of the histograms. In order to be able to compare histograms that contain very different total numbers of hits, one usually normalizes the intensities of the images. The Matlab graphics does this automatically, unless other options are specified. Then, however, the clusters in low-intensity images may be overemphasized and should not be compared directly with clusters in the other images.

3.2 General Nouns

The class of the *general nouns*, differing from the class of all nouns shown in Fig. 1, does not contain any names of persons or places, or nouns of time.

The effect of word frequency on the general nouns is even more surprising than that on the general adjectives. From Fig. 5 we see that the general nouns that occur with the lowest frequencies (1 to 10), and whose number in the corpus is also the highest, have a very broad distribution. Compared with Fig. 1, however, the differences are not very large in this range. On the other hand, in the range of 10 to 100 of word frequencies, the centroid of the histogram has already moved to the right and upwards. In the range of 100 to 1000 of word frequencies, most of the nouns are clustered into three very compact subsets close to the upper right corner, and one compact cluster at the bottom. In this range of frequencies the nouns may have only fewer definite semantic roles, whereas the roles of the more rare nouns are more vague. The fourth histogram in Fig. 5 as well as in Fig. 4 contain so few words that it is difficult to draw any conclusions from them.

In all of the above partial diagrams of general nouns. there is a salient empty oval region in the middle, where the verbs, according to Fig. 1, are located.

3.3 Verbs

Verbs without objects. In many languages, this category of verbs is called the *intransitive verbs*, while in some other languages (like Chinese and French) the term *verbs without objects* is used. In Fig. 6, the least frequently (1 to 10 times) used verbs have a fuzzy cluster on the top-left. This cluster coincides with that of the predicative idioms (not shown in this publication but in [4]), and so this cluster of verbs is believed to represent verbs used as the so-called *center of predicates*.

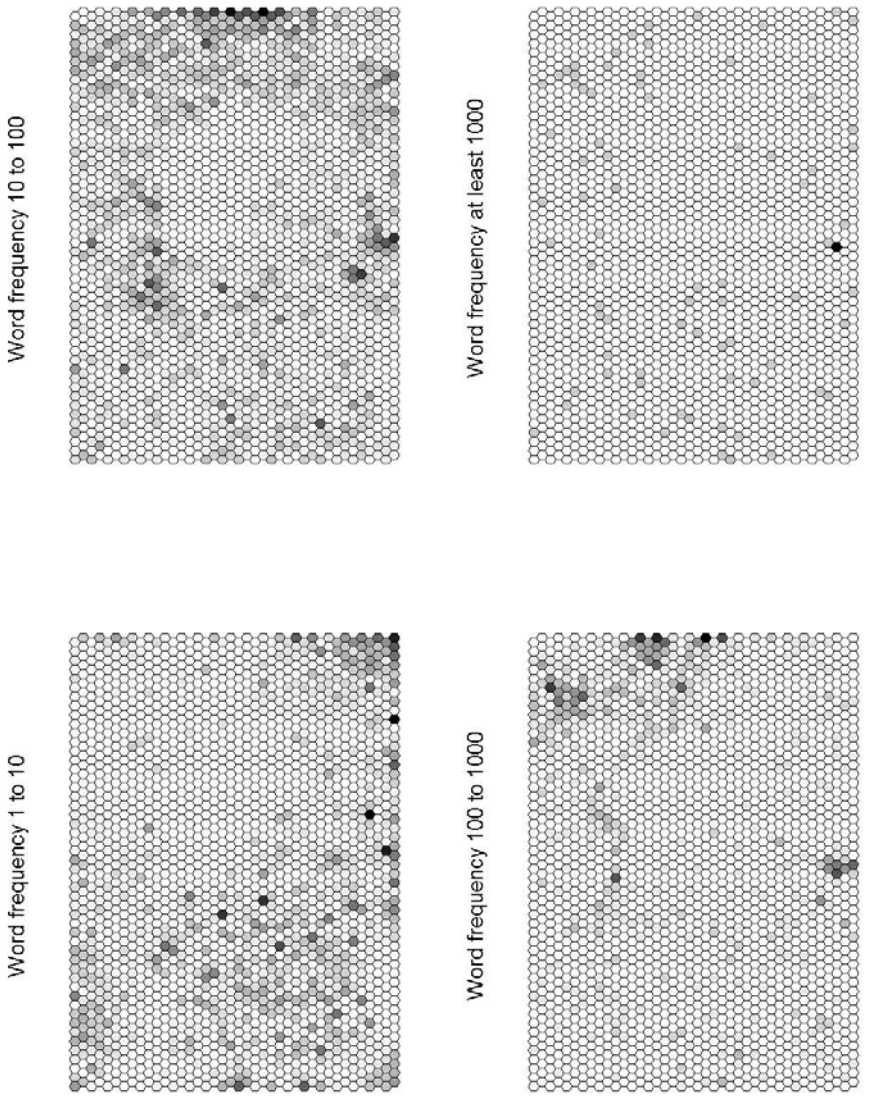


Fig. 5. Histograms of general nouns, as a function of word frequency

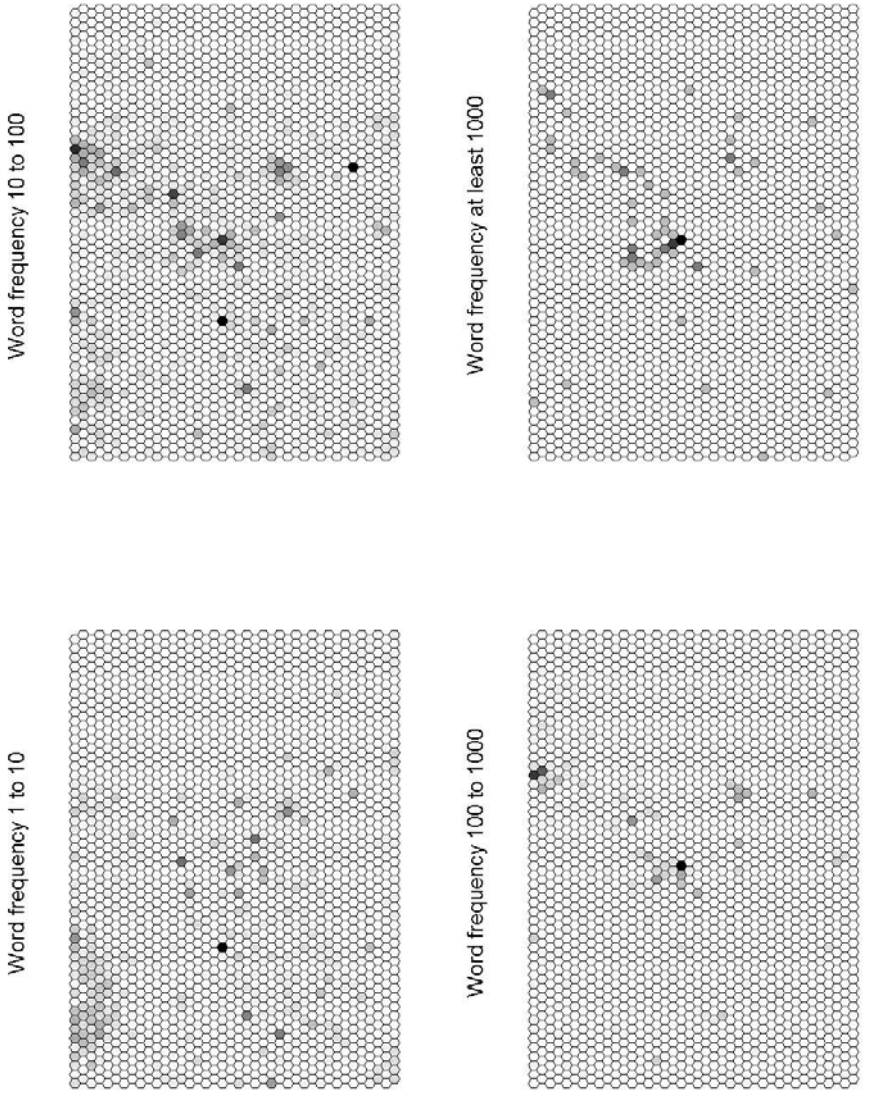


Fig. 6. Histograms of verbs without objects, as a function of word frequency

Fig. 7. Histograms of verbs followed by nouns, as a function of word frequency

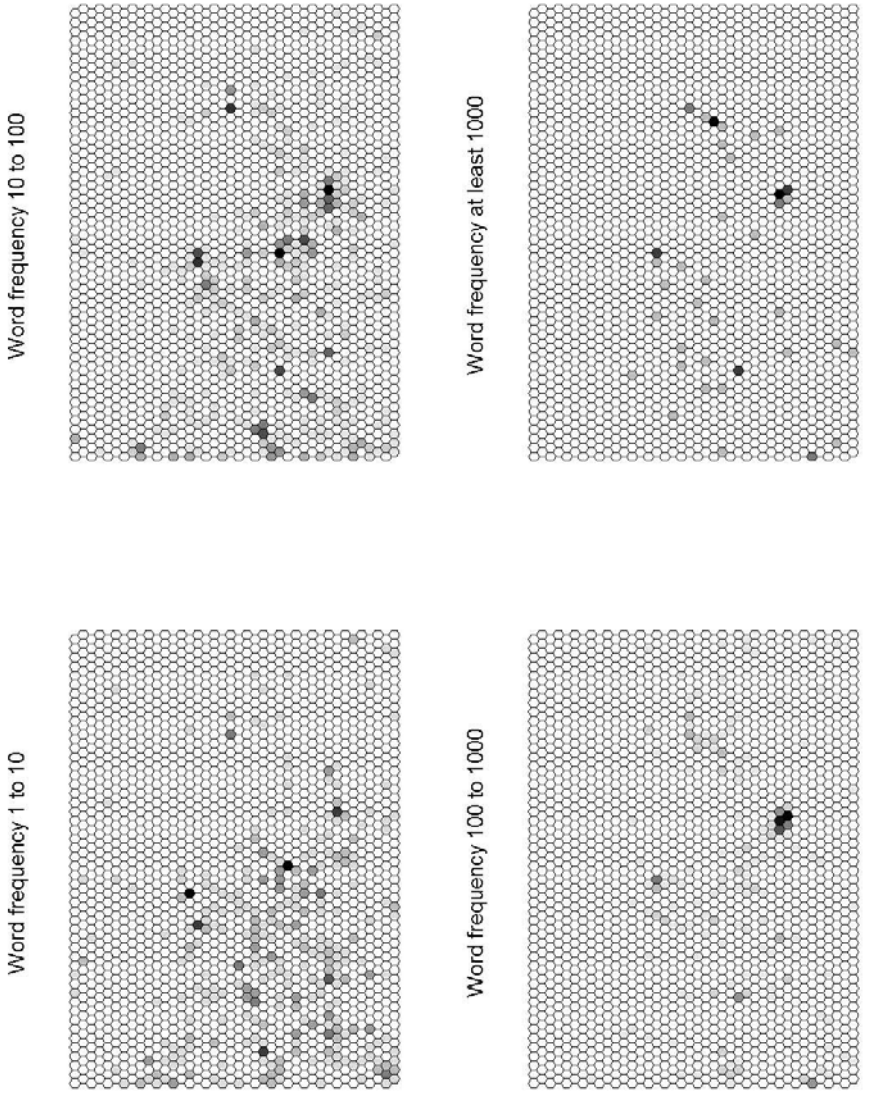


Fig. 8. Histograms of general adverbs, as a function of word frequency

The other verbs without objects are clustered mainly in the middle, where the nouns have an empty space.

Verbs followed by nouns. This category would be called the *transitive verbs* in some other languages. The noun subsequent to the verb forms a very close context with the former, and so one might expect that this correlation should also be reflected in the contextual SOM. Indeed, almost independent of the word frequency, the histograms in Fig. 7 are clustered into the middle of the empty space in the distribution of the nouns. It seems that the locations of these clusters have been automatically fitted to the locations of the surrounding nouns.

The correlation coefficient of the histogram of the verbs without objects and that of the verbs followed by nouns is 0.2514, indicating that their linguistic roles are different.

3.4 General Adverbs

The general adverbs are mainly located in the histograms along the border between the nouns and the verbs (Fig. 8). They have only relatively few clusters in fixed places, showing that there are only few main types of adverbs, and their contexts depend very little on word frequency.

3.5 Other Classes

As mentioned earlier, there were 88 linguistic classes into which the words of the MCRC were divided. Histograms of some of them can be found in [4] and [5].

The *numerals* are clustered very compactly around the lower left corner of the SOM, and this cluster does not depend on word frequency.

The *conjunctions*, on the other hand, have histograms scattered randomly over the area of the SOM, showing that they do not correlate with the text.

The *pronouns* are projected into areas occupied by the other word classes. In Fig. 2 we saw the mapping of the attributive pronouns. *The pronouns used as subjects or objects* have histograms similar to those of the nouns, confirming that it is the role of the words as sentence constituents rather than their linguistic class that is reflected in the contextual SOMs.

The verbs, in general, are clustered into a round area in the middle of the SOM. An exceptional subset of verbs is formed by those 1035 verbs used *as the core of a noun phrase*. Independent of their frequency, they are mapped compactly into the lower right corner of the SOM, indicating that these verbs occur so tightly with the other words in the noun phrases that the latter words determine the location of the cluster on the SOM.

4 Discussion

Differing from previous approaches to the contextual SOMs, in which individual words were mapped onto the SOM array, in this work histograms of various word classes or otherwise defined subsets of words were formed over the array.

In addition to being among the first works in which contextual SOMs have been constructed for the *Chinese language*, this study contains two new results. First, it has been found that the target words, on the basis of their local contexts, are not only clustered according to the main linguistic classes. It seems that the role of the words as *sentence constituents* defines their location more closely in the contextual SOM. Second, the histograms have also been found to depend on the *frequencies of the words* selected for testing. In some cases, e.g., for nouns and adjectives (the histograms of which were the most diffuse ones) this dependence is strong, whereas for some other word classes it is much weaker.

One simple explanation of the frequency dependence that comes into mind is that the MCRC corpus used in this work is very heterogeneous. It contains texts from very different areas written by different people. The vocabularies of the different parts, especially the sets of nouns and adjectives used in them have probably very different word frequencies. Conversely, when the word frequencies during testing are restricted to certain intervals, these words correlate closest with certain parts of the corpus, and thus with the specific topic and writer. It would be very interesting to compare the present results with those produced by one author only and dealing with a well-defined topic area, preferably written in a traditional style.

On the other hand, it is also thinkable that the contexts in which especially the nouns and the adjectives are used have *transformed* with time, and frequent usage accelerates this transformation. One fact that supports this assumption is that a histogram as a function of word frequency often changes *gradually* in the same direction (cf. Figs. 4 and 5).

In the contextual SOM, the selection of the *random-vector representations* for the words may have an effect on the exact form of the SOM, due to statistical variations in the matching of the random vectors. These statistical variations could be eliminated for the most part if one were able to use representation vectors with extremely high dimensionalities, for which supercomputers would be needed.

The main message of the work in presentation is that the word frequencies probably have an important role in all of the contextual-SOM experiments and should be taken into account when picking up words from the lexica for testing.

The two main conclusions derivable from the work in presentation are thus: 1. If one wants to produce contextual maps in which the word classes are well segregated, one may select a vocabulary that contains only the most regular words, i.e., only words that have their frequency above a certain limit, and discard the very rare words, as usually also has been done in previous experiments. 2. If all of the occurring words are taken into account, however, one is able to see intriguing transformations of the word classes as a function of the frequency of usage of the words, as demonstrated in this work.

Acknowledgements. This work has been carried out under the auspices of the Academy of Finland, the Aalto University of Finland, and the Beijing Language and Culture University of China. Special thanks are due to Drs. Zhirong Yang and Timo Honkela of Aalto University for their help in the decoding of the

Unicode files used to transfer the Chinese text corpus. Dr. Xiaowei Zhao of the Golgate University, U.S.A., has provided an excellent translation of the linguistic analysis of the word list of the MCRC corpus. Dr. Ping Li of the Pennsylvania State University, U.S.A., has created the contacts between Aalto University and Beijing Language and Culture University, and followed this work with keen interest. Dr. Jorma Laaksonen has been very helpful at various editing phases of this article.

References

1. Ritter, H., Kohonen, T.: Self-organizing semantic maps. Biol. Cyb. 61, 241–254 (1989)
2. Honkela, T., Pulkki, V., Kohonen, T.: Contextual relations of words in Grimm tales, analyzed by self-organizing maps. In: Fogelman-Soulié, F., Gallinari, P. (eds.) Proc. ICANN 1995, Int. Conf. on Artificial Neural Networks, vol. II, pp. 3–7. EC2, Nanterre, France (1995)
3. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, Heidelberg (2001)
4. Kohonen, T.: Contextually Self-Organized Maps of Chinese Words. TKK Reports in Information and Computer Science, TKK-ICS-R30. Aalto University School of Science and Technology, Espoo, Finland (2010) (This report is downloadable from ics.tkk.fi/en/research/publications)
5. Kohonen, T.: Contextually Self-Organized Maps of Chinese Words. Part II, TKK Reports in Information and Computer Science, TKK-ICS-R35. Aalto University School of Science and Technology, Espoo, Finland (2010) (This report is downloadable from ics.tkk.fi/en/research/publications)
6. Sun, H.L., Sun, D.J., Huang, J.P., Li, D.J., Xing, H.B.: Corpus for modern Chinese research. In: Luo, Z.S., Yuan, Y.L. (eds.) Studies in the Chinese language and characters in the era of computers, pp. 283–294. Tsinghua University Press, Beijing, China (1996)
7. Vesanto, J., Alhoniemi, E., Himberg, J., Kiviluoto, K., Parviainen, J.: Self-organizing map for data mining in Matlab: the SOM Toolbox. Simulation News Europe 25, 54 (1999) (The SOM Toolbox software package is downloadable from ics.tkk.fi/en/research/software)
8. Kohonen, T., Nieminen, I.T., Honkela, T.: On the quantization error in SOM vs. VQ: A critical and systematic study. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 133–144. Springer, Heidelberg (2009)

Assessing the Efficiency of Health Care Providers: A SOM Perspective

Marina Resta

University of Genova,
DIEM sezione Matematica Finanziaria, via Vivaldi 5, 16126 Genova, Italy
55246@unige.it
http://www.diem.unige.it/Marina_RESTA.html

Abstract. We explored the use of Self Organizing Map (SOM) to assess the problem of efficiency measurement in the case of health care providers. To do this, we used as input the data from the balance sheets of 300 health care providers, as resulting from the Italian Statistics Institute (ISTAT) database, and we examined their representation obtained both by running classical SOM algorithm, and by modifying it through the replacement of standard Euclidean distance with the generalized Minkowski metrics. Finally, we have shown how the results may be employed to perform graph mining on data. In this way, we were able to discover intrinsic relationships among health care providers that, in our opinion, can be of help to stakeholders to improve the quality of health care service. Our results seem to contribute to the existing literature in at least two ways: (a) using SOM to analyze data of health care providers is completely new; (b) SOM graph mining shows, in turn, elements of innovations for the way the adjacency matrix is formed, with the connections among SOM winner nodes used as starting point to the process.

Keywords: SOM, Network Representation, Efficiency, Health Care Providers.

1 Introduction

In an ideal world the health system should be effective, and it should be efficient, i.e. it should be able to achieve the specified outcomes in a way to maximise access, outputs and outcomes within the available resources. In the real world, however, this does not happen. Just to make an example, looking at the situation of Italy, health care expenditure plays a crucial impact into the financial resources of the country; nevertheless our health care system is lesser efficient than others, and it is not easy to explain why. In particular, the basic difficulty is to find a common platform to compare efficiency of health systems, because of their intrinsic complexity, and of certain ambiguity in what does efficiency itself consist and how to measure it.

For what it concerns the complexity of health systems, a quite recent study of the Australian National Health and Hospitals Reform Commission [10] found

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 30–39, 2011.

low relation between efficiency and the level of health spendings, thus suggesting that rather than increasing expenditures, regulatory efforts should be addressed on different allocation of the existing resources.

With respect to the ambiguity of the definition of efficiency and to the way to measure it, there are at least two issues we can point on. The first one relates to the method used to generate efficiency scores. Most commonly used techniques include Data Envelopment Analysis (DEA) and Stochastic Frontier Analysis (SFA) [9]; there is a huge literature devoted to compare them, and to evaluate their statistical properties [5], [13]. The second major area of research uses either DFA or SFA to examine efficiency in a single area of health care production: [4] focused on hospitals; [11] on pharmaceutical industry, and [3] on long term care, just to cite some.

Within such framework, the main contributions of this work may be briefly summarized as follows:

- we focused on the case of Italy, and, being aware of the need for the country to control health care costs, we examined the balance sheets data of 300 health care providers that receive total or partial public fundings;
- we run our analysis using Self Organizing Maps (SOMs): since, to the best of our knowledge, this is a first time application in the health care sector, we mainly addressed our efforts to the application of *classic* SOM algorithm. As unique concession to more sophisticated analysis, due to the high dimensionality of the dataset, we explored the convenience to train SOM with similarity measures other than the Euclidean one, such metrics being chosen among Minkowski norms [17]:

$$||X||_p = \left(\sum_i |X_i|^p \right)^{\frac{1}{p}}, \text{ for } p \in \mathbb{R}_+. \tag{1}$$

  We have then analized how the clustering capabilities of SOM are modified when both prenorms ($0 < p < 1$), and ultrametrics ($p >> 1$) are considered.
- As final step, we selected the best performing SOM, and we analyzed the connections among the winner nodes, thus obtaining an adjacency matrix that has been the starting point for a network representation of health care providers. We have used it to retrieve more information about their efficiency, and to suggest some economic interpretations of the results.

This paper is therefore organized as follows: Section 2 briefly describes the problem we focused on, and the data we have considered in the study; Section 3 provides a glimpse on the literature that deals on the alternatives to standard Euclidean metric, and then illustrates the changes carried on SOM algorithm to use it with norms derived from (1). Section 4 illustrates the details of our simulations, and discusses the results we have obtained. Section 5 concludes.

2 Problem Statement and Data Description

The Italian health system assumes that health services can be provided both by public and private structures, the former essentially totally funded.

Here the term public identifies two kind of structures: *Aziende Sanitarie Locali* (ASL) and *Aziende Ospedalierie* (AO). The main difference between the two enterprises stands in the fact that while AO are generally single structures (namely: hospitals), ASL, on the other hand, are more composite, since, by definition of law, they can include more than one local (regional, district, municipal) units that provide health care to citizens.

According to the more recent reform statements of the public health sector, ASL and AO are required to act like autonomous units to control their financial flows. This means that:

(i) Each unit of the system should exhibit capabilities concerning the management of economic and financial flows.
(ii) The efficiency of each unit does not only depend on factors of technical type (such as quality of the provided health service, innovation, satisfaction of the final consumer), but also by more strictly financial factors.
(iii) The capability of the whole system to maintain satisfying levels of solvency and efficiency depends, in turn, on those of every component of the system (ASL and AO), and on their capability to interact one to each other.

The efficiency of the system becomes therefore something that include in a broad sense the bare management of financial variables: for this reason we have analyzed the balance sheets of 300 public enterprises (ASL and AO), as resulting from the more recent Italian Statistics Institute (ISTAT[1]) database. The goal was to retain information that might help to monitor the actual level of efficiency of the National Health System, and, eventually, to find some suggestions to improve it.

The data under examination were arranged into two different aggregation levels: regional, and by single unit. Since Italy is organized into twenty regional districts (as resulting from Table 1), we managed twenty files, and within each of them, a variable number of financial statements of public health care providers operating into the region itself.

Every unit is identified by a string code whose first part is the region ID, and the second part is a number varying from 101 to 999. For instance, PIEM101 identifies the first ASL of Turin in Piedmont, while VEN112 is associated to the ASL of Venice, and so on. The records in the balance sheet, on the other hand, are organized according to the principles of the International Accounting Standards (IAS[2]), so that they capture the financial flows of each single unit. Examples of such flows are given by fundings (from public institutions or from private organizations), inflows deriving from the provision of health services, or costs and liabilities, for an overall number of 164 variables.

If we examine the data in the traditional accounting way, we should move to set apart from the balance sheet those variables that are generally employed to calculate financial ratios, but we decided to behave differently, for at least two reasons. The first one is that although financial ratios should accomplish to

[1] www.istat.it
[2] http://www.ifrs.org/Home.htm

Table 1. Name of Italian Regional Districts, and the ID associated to them throughout the paper

Name	ID	Name	ID
Abruzzo	ABR	Aosta Valley	VDA
Apulia	PGL	Basilicata	BAS
Calabria	CAL	Campania	CAM
Emilia–Romagna	EMROM	Friuli–Venezia Giulia	FRI
Lazio	LAZ	Liguria	LIG
Lombardy	LOM	Marche	MAR
Molise	MOL	Piedmont	PIEM
Sardinia	SAR	Sicily	SIC
Trentino–Alto Adige	TNT–BZ	Tuscany	TOSC
Umbria	UMB	Veneto	VEN

simplification purposes, the number of ratios that can be built from the balance sheet does not sensitively differ from the number of records in the balance sheet itself. A more technical explanation of our choice comes by looking to the pecularity of data we are considering. Both ASL and AO, in fact, are enterprises almost uniquely devoted to provide health care services, so that the greater part of the records we can read in their balance sheet pertains costs and inflows related to such specific activity; on the other hand, the accounting literature does not provide proper financial ratios that can be able to capture such specificity.

As a result, we decided to consider all the available data from the financial statements of ASL and AO, thus obtaining an input matrix of dimensions 300×164, where each row represents either ASL or AO with their 164 normalized determinants.

3 The SOM Algorithm and the Curse of Dimensionality

In recent years a number of contributions questioned (mostly from the theoretical point of view) on the relevance of the Euclidean norm, when it is used to deal with data embedded into high–dimensional spaces [7]. The problem is that it might make pairwise distances more similar than effectively they are; this, in turn, might lead to regrettable inconvenients, especially in cases where the distance among various patterns is the fundament for more complex content retrieval tasks [1]. Being aware of such curse of dimensionality, [2] focused on the analysis of the concentration in the alternatives to the standard Euclidean norm, and stressed the attention on the family of generalised Minkowsky norms:

$$||X||_p = \left(\sum_i |X_i|^p \right)^{\frac{1}{p}} . \tag{2}$$

Here p is a strictly positive real value (fractional norms). Using the family defined by (2), [2] observed that nearest neighbour search is meaningless in high-dimensional spaces for integer p values equal or greater that two (the so called

ultrametrics). Those results are general, in the sense that they hold also when p takes positive real values. In addition, [8] outlined that the optimal distance could depend on the type of noise on the data: fractional norms should be preferable in the case of colored noise, while in the case of Gaussian noise, the Euclidean metrics should be more robust than fractional ones. More recently, [17] gave also proof that, in contrast to what expected, prenorms ($0 < p < 1$) are not always less concentrated than higher order norms. Finally, [14] and [15] provided evidence that the use of both prenorms and ultrametrics can be noteworthy, when dealing with financial data.

We considered such debate of particular interest for our study, since we need to manage input patterns embedded into a very high–dimensional space: we are primarily concerned to test if the performances of SOM may take advantage from changes in the adopted similarity measures.

To do this, we needed to modify the SOM procedure in a proper way. In practice, the plain SOM uses a set of q neurons, (arranged either on a strip or into a 2D rectangular or hexagonal grid) to form a discrete topological mapping of an input space embedded into a n–dimensions space ($n >> 2$). At the start of the learning, all the weights are initialised at random. Then the algorithm repeats the following steps: we will refer to the case of a mono–dimensional SOM, but the layout presented can be easily generalized to higher dimensional grids.

If $\mathbf{x}(t) = \{x_j(t)\}_{j=1,...,n} \in \mathbb{R}^n$ is the input item presented at time t to a map M having q nodes with weights $\mathbf{m}_i(t) = \{m_{i,j}(t)\}_{j=1,...,n} \in \mathbb{R}^n$, $(i = 1, ..., q)$, i_t^* will be claimed the winner neuron at step t iff:

$$i_t^* = argmin_{i \in M} \left(\sum_{i \in M} \sum_{j=1}^{n} |x_j(t) - m_{i,j}(t)|^p \right)^{1/p}, \; p \in \mathbb{N} \; . \tag{3}$$

Where p is the distance parameter. More common choices for p include $p = 1$ (Manhattan or city block distance), and $p = 2$ (Euclidean distance).

Once the leader has been identified according to (3), the correction of nodes in the map takes place; if $N_{i^*}(t)$ is the set of neurons in the map belonging to the *neighbourhood* of i^* (in a topological sense), then:

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + h_{i*,i}(t)[\mathbf{x}(t) - \mathbf{m}_i(t)] \; . \tag{4}$$

Where $h_{i^*,i}(\cdot)$ is an interaction function, governing the way the nodes adjust in relation to the winning neuron on the grid: most common shapes for h include the constant function, and the Gaussian function [12]. After iterating such procedure over a number of epochs, the map should tend to a steady organized state, and neighbouring neurons should represent similar inputs. The degree of organization reached by the map can be checked by means of convergence indexes, such as the quantization error [18]: in this way, the learning procedure is stopped once a proper convergence threshold level is reached.

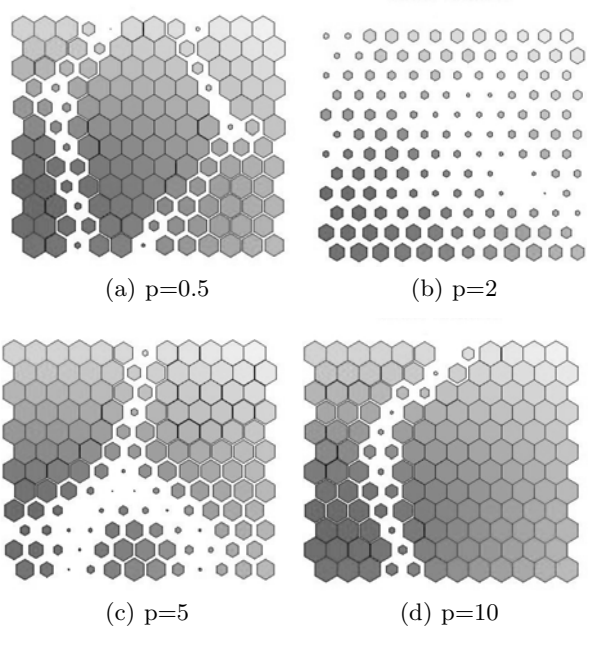
In accordance to the studies presented in [2] and [17], we have then examined various extensions of the standard Minkowski metrics appearing as argument in

(3), and we trained SOM accordingly. In particular, we have relaxed (3), allowing p to assume real positive values, to include both ultrametrics ($p >> 1$), and prenorms ($0 < p < 1$). Obviously changes affected all the procedures involving the use of the Euclidean metric as similarity measure, including, for instance, the search for best matching units and the evaluation of quantization error.

4 Simulations and Discussion of the Results

We run simulations considering values of p in the range $[0.5, 10]$ sampled at step 0.5 for an overall number of twenty alternative p values. For each of them we trained a bunch of 100 plain SOMs with rectangular grid topology, and dimensions varying from 5×5 to 21×21, isolating the SOM with best performances in terms of quantization error. For every value of p such ideal SOM tends to exhibit very closer topology grid dimensions (around 12×12.) Our next move was then to choose among the best performing SOMs the most representative one. In this task we considered both the level of the quantization error, and the organization of SOM nodes. Figure 1 provides a look at the four most significant results.

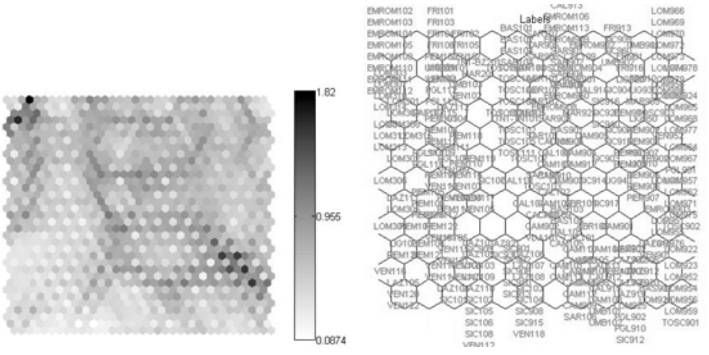
One can immediately note the concentration effect for $p > 2$ (Figure 1(c) and 1(d)): the blank parts of the maps are the only ones where winner nodes are placed. This is a common feature to all SOMs trained with $p >> 2$. Concentration was less evident for $p < 2$; in such case, however, the advantage of using p values other than two was not as higher (with respect to the quantization error) as to justify the replacement of $p = 2$. We then concluded that, at least in our



(a) p=0.5 (b) p=2

(c) p=5 (d) p=10

Fig. 1. Distance matrix with map units size for the four best performing SOMs



(a) (b)

Fig. 2. From left to right: U–matrix (2(a)) and Best Matching Units –BMUs– (2(b)) for the best SOM trained with $p = 2$. It may be noticed the sparsity of BMUs

case, despite of the size of the embedding dimension, plain SOM trained with the standard Euclidean norm still remains the best choice.

However, focusing on the case of $p = 2$ (see Figure 2), we noticed that, despite of the overall good performance of SOM in terms of quantization error, the winner nodes were too much sparse, to our purposes. We then decided to move one step further, and we analized the connections among winner nodes (Best Matching Units –BMUs) to build the related adjacency matrix. In practice, we used SOM to perform graph mining like in [6], but with the difference that we acted directly on the connections of SOM BMUs. The algorithm we used is similar to that introduced in [16] to build Planar Maximally Filtered Graph (PMFG), with changes involving the way distances among BMUs are evaluated: where [16] uses correlation, here we used (1), with the p value as selected in the previous stage of the procedure (in our case: $p = 2$.)

As a result, we obtained a representation of SOM nodes connections like the one shown in Figure 3. Although the representation need to be interpreted with certain care, the graph allowed us to extract some notable information. First of all, the twelve clusters that now clearly emerge from the SOM exhibit quite distinct features: we are going to discuss the more significant ones. Clusters 1 and 2 are characterized by lower overall positive revenues, and higher specific (i.e. related to the provision of health care services) costs, cluster 3 is the one with both the highest revenues from medical activity, and the lowest taxation costs; clusters 4 and 9 are those which invest more on employees training. On the other hand, cluster 5 groups enteprises which have received lower public fundings: it is not very surprising to discover that this cluster is associated to the lowest level in the value of production. In the balance sheet this variable generally monitors the enterprise overall inflows: the (quite trivial) lesson we can learn from this cluster is then that its members seem not able to manage financial inflows others than public fundings. Cluster 8 is in the opposite situation of cluster 5, receiving the highest level of public fundings, but despite of it, its members were not able

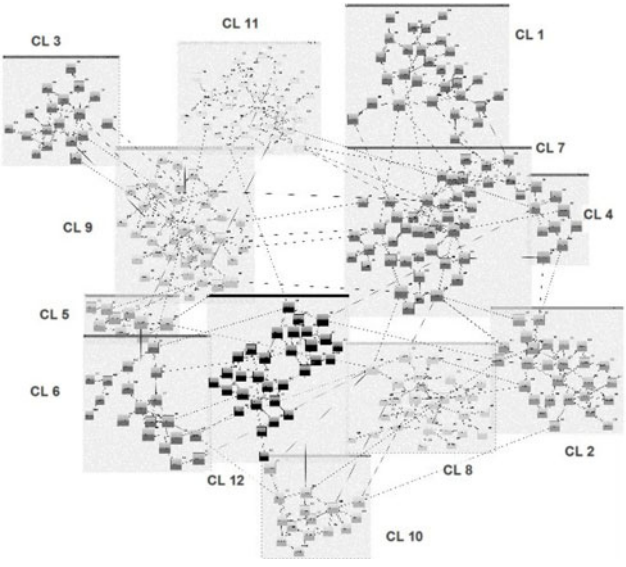


Fig. 3. Graph mining on SOM

to reach the best financial results. Finally, cluster 11 exhibits the best financial flows among those not specifically related to the health care provision.

Another interesting remark relates to the composition of clusters: clusters are not territorial homogeneous, i.e. they generally group ASL and AO from different regions; a partial exception to this rule is provided by cluster 2 that includes 53% of units from the region Emilia Romagna (EMROM). This could be of particular importance, because it points on the existence of financial differences among public health care providers belonging to the same region. This, in turn, suggests that greater efficiency could be reached by operating on the allocation of public fundings at regional level.

Finally, the organization of clusters provides information at technical level too, suggesting that wraparound grid topologies (either toroidal or cylindric) could reach more satisfying results.

5 Conclusion

In this paper we discussed an application of Self Organizing Map (SOM) to assess the efficiency of health care providers. To do this, we examined by means of SOM the data of the balance sheet of 300 italian health care providers that receive public fundings. Since, to the best of our knowledge, this is a first–time application in the health care sector, we mainly addressed our efforts to the application of *classic* SOM algorithm. As unique concession to more sophisticated analysis, due to the high dimensionality of the dataset we explored the convenience to train SOM with similarity measures other than the Euclidean one,

the metrics being chosen among Minkowski norms, as defined in [17]. We then trained 20 blocks of SOM each of 100 maps, characterized by various grid topology size, and by different distance metrics. The SOM performances were checked focusing on the quantization error, variously evaluated according to the metric in use. We obtained the best results with both SOM trained with the standard Euclidean metric and with those trained through prenorms. In this latter case, however, the gains in terms of quantization error were not as significant as to justify the leaving of the Euclidean metric. In addition, we found that the information provided by SOM were too much sparse to be significant to our purposes, and we then moved one step further, using the map best matching units to build an adjacency matrix that has been then starting point to a graph mining process. This task was particularly proficient, since it allowed us to retain a number of information about the efficiency condition of the health system in Italy. In particular, we observed that more than increasing health care expenditures, a succesfull move could be that to potentiate the integration among regions, and the allocation of existing funds inside the regions themselves. Moreover, from the technical point of view, the clusters organization we obtained suggests the direction for further experiments: we could probably get better and more refined results using a different grid topology, like the cylindric or the toroidal one.

References

1. Aggarwal, C.C., Yu, P.S.: The IGrid Index: Reversing the Dimensionality Curse For Similarity Indexing in High Dimensional Space. In: Proc. of KDD, pp. 119–129 (2000)
2. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the Surprising Behavior of Distance Metrics in High Dimensional Space. In: Van den Bussche, J., Vianu, V. (eds.) ICDT 2001. LNCS, vol. 1973, pp. 420–434. Springer, Heidelberg (2000)
3. Bjorkgren, M., Hakkinen, U., Linna, M.: Measuring Efficiency of Long Term Care Units in Finland. Health Care Management Science 4(3), 193–201 (2001)
4. Braithwaite, J., Westbrook, M., Hindle, D., Ledema, R., Black, D.: Does Restructuring Hospitals Result in Greater Efficiency?-an Empirical Test Using Diachronic Data. Health Services Management Research 19(1), 1–13 (2006)
5. Banker, R.: Maximum Likelihood, Consistency and Data Envelopment Analysis: A Statistical Foundation. Management Science 39(10), 1265–1273 (1993)
6. 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)
7. Demartines, P.: Analyse de Données par Réseaux de Neurones Auto-Organisés. PhD dissertation, Institut Nat'l Polytechnique de Grenoble, Grenoble, France (1994)
8. Francois, D., Wertz, V., Verleysen, M.: Non-euclidean metrics for similarity search in noisy datasets. In: Proc. of ESANN 2005, European Symposium on Artificial Neural Networks (2005)
9. Hollingsworth, B.: Non-Parametric and Parametric Applications Measuring Efficiency in Health Care. Health Care Management Science 6(4), 203–218 (2003)
10. Hurley, E., McRae, I., Bigg, I., Stackhouse, L., Boxall, A.M., Broadhead, P.: The Australian health care system: the potential for efficiency gains. In: Working paper, Australian Government National Health and Hospitals Reform Commission (2009)

11. Key, B., Reed, R., Sclar, D.: First-order Economizing: Organizational Adaptation and the Elimination of Waste in the U.S. Pharmaceutical Industry. Journal of Managerial Issues 17(4), 511–528 (2005)
12. Kohonen, T.: Self-Organizing Maps. Springer, Berlin (2002)
13. Murillo Zamorano, L.: Economic Efficiency and Frontier Techniques. Journal of Economic Surveys 18(1), 33–77 (2004)
14. Resta, M.: Seize the (intra)day: Features selection and rules extraction for tradings on high-frequency data. Neurocomputing 72(16-18), 3413–3427 (2009)
15. Resta, M.: On the Impact of the Metrics Choice in SOM Learning: Some Empirical Results from Financial Data. In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010. LNCS, vol. 6278, pp. 583–591. Springer, Heidelberg (2010)
16. Tumminello, M., Aste, T., Di Matteo, T., Mantegna, R.N.: A tool for filtering information in complex systems. PNAS 102(30), 10421–10426 (2005)
17. Verleysen, M., Francois, D.: The Concentration of Fractional Distances. IEEE Trans. on Knowledge and Data Engineering 19(7), 873–886 (2007)
18. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: SOM Toolbox for Matlab 5. Helsinki University of Technology Technical Report (2000)

Fuzzy Clustering of the Self-Organizing Map: Some Applications on Financial Time Series

Peter Sarlin[1,2] and Tomas Eklund[1]

[1] Turku Centre for Computer Science – TUCS, Department of Information Technologies, Åbo Akademi University, Joukahaisenkatu 3-5, 20520 Turku, Finland
[2] European Central Bank, Kaiserstrasse 29, D-60311 Frankfurt am Main, Germany
{Peter.Sarlin,Tomas.Eklund}@abo.fi

Abstract. The Self-organizing map (SOM) has been widely used in financial applications, not least for time-series analysis. The SOM has not only been utilized as a stand-alone clustering technique, its output has also been used as input for second-stage clustering. However, one ambiguity with the SOM clustering is that the degree of membership in a particular cluster is not always easy to judge. To this end, we propose a fuzzy C-means clustering of the units of two previously presented SOM models for financial time-series analysis: financial benchmarking of companies and monitoring indicators of currency crises. It allows each time-series point to have a partial membership in all identified, but overlapping, clusters, where the cluster centers express the representative financial states for the companies and countries, while the fluctuations of the membership degrees represent their variations over time.

Keywords: Self-organizing maps, fuzzy C-means, financial time series.

1 Introduction

The Self-organizing map (SOM), proposed by Kohonen [1], has been widely used in industrial applications. It is an unsupervised and nonparametric neural network approach that pursues a simultaneous clustering and projection of high-dimensional data. While clustering algorithms, in general, attempt to partition data into natural groups by maximizing inter-cluster distance and minimizing intra-cluster distance, the SOM performs a clustering of a slightly different nature. The SOM can be thought of as a spatially constrained form of *k*-means clustering or as a projection maintaining the neighborhood relations in the data. In the early days of the SOM, information extraction was mainly facilitated by visual analysis of a U-matrix, where a color code between all neighboring nodes indicates their average distance [2]. The SOM has, however, not only been utilized as a stand-alone clustering technique, its output has also been used as input for a second stage of two-level clustering. Lampinen and Oja [3] proposed a two-level clustering by feeding the outputs of the first SOM into a second SOM. Further, Vesanto and Alhoniemi [4] outperformed stand-alone techniques using a two-level approach with both hierarchical agglomerative and partitional *k*-means clustering algorithms. Minimum distance and variance criteria have also been proposed for SOM clustering [5–7].

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 40–50, 2011.

However, one ambiguity with the SOM clustering is that the degree of membership in a particular cluster is not always easy to judge. In some cases, it might be beneficial to judge the degree to which a particular area of a cluster differs from the rest of the cluster, and what its closest match among the other clusters is. To this end, we apply fuzzy C-means (FCM) [8] clustering on the units of the SOM grid. The FCM algorithm allows each unit to have a partial membership in all identified, but overlapping, clusters. This enables sensible representation of the real world filled with uncertainty and imprecision. The model is not only expected to provide an adequate clustering, but also to enable easily interpretable visualizations of the evolution of cluster memberships over time. As the crispness of the data cannot be known *a priori*, the FCM clustering presents information on the overlapping of the clusters, be they crisp or fuzzy, while still always enabling comparisons between data points. We apply FCM clustering to two previously presented SOM models for financial time-series analysis: financial benchmarking of companies [9] and monitoring indicators of currency crises [10]. In this paper, such as in Liu and Lindholm's [11] stand-alone FCM clustering, the cluster centers express the representative financial states for the companies and countries, while the varying membership degrees represent their fluctuations over time. The results indicate that fuzzy clustering of the SOM units is a useful addition to visual monitoring of financial time-series data.

The paper is structured as follows. Section 2 discusses fuzzy clustering of the SOM. In Section 3, the two-level clustering is applied on financial time series. Section 4 concludes by presenting our key findings and future research directions.

2 The Two-Level SOM-FCM Model

The SOM is a non-parametric artificial neural network utilizing a competitive learning method first developed in [1]. The network of neurons consists of an input layer and an output layer. The number of neurons in the input layer equals the dimensions of the data, while the output layer is a topological grid. The SOM algorithm used in this paper is described here briefly – for further reference, see [12].

The training process starts with an ordered (e.g., principal component analysis) or random initialization of the reference vectors. The training algorithm has two steps: (1) finding the best-matching units (BMUs) and (2) adjusting the reference vectors. The first step compares, using the Euclidean distance, each input data vector x_j (where j=1,2,…,N) with the network's reference vectors m_i (where i=1,2,…,M) to find the best match m_b,

$$\left\| x_j - m_b \right\| = \min_i \left\| x_j - m_i \right\|, \tag{1}$$

such that the distance between the data vector x_j and the BMU m_b is less than or equal the distance between x_j and any other reference vector m_i. Then the second step adjusts each reference vector m_i with the sequential updating algorithm [12, p. 111]:

$$m_i(t+1) = m_i(t) + h_{ib(j)}(t)\left[x(t) - m_i(t)\right], \tag{2}$$

where t is a discrete time coordinate and $h_{ib(j)}$ a neighborhood function. The reference vectors can also be updated using the batch algorithm, which projects all x_j to their m_b before each m_i is updated [12, p. 138].

The units of the map can further be divided into clusters of similar units. Instead of dividing the units into crisp clusters, we employ the FCM algorithm, developed by [13] and improved by [8], for assigning a degree of membership of each unit in each of the clusters. The FCM algorithm implements an objective function-based fuzzy clustering method. The objective function J_μ is defined as the weighted sum of the Euclidean distances between each unit and each cluster center, where the weights are the degree of memberships of each unit in each cluster, and constrained by the requirement that the sum of memberships of each point equals 1:

$$J_\mu = \sum_{i=1}^{M}\sum_{k=1}^{C} {u_{ik}}^{\mu} \left\| m_i - c_k \right\|^2, \qquad \sum_{k=1}^{C} u_{ik} = 1, \tag{3}$$

where $\mu \in (1, \infty)$ is the fuzzy exponent, u_{ik} is the degree of membership of reference vector m_i (where i=1,2,…,M) in the cluster center c_k (where k=1,2,…,C, and 1<C<M), and $\left\| m_i - c_k \right\|^2$ is the squared Euclidean distance between m_i and c_k. It operates through an iterative optimization of J_μ by updating the membership degree u_{ik}:

$$u_{ik} = 1 \Big/ \left(\sum_{s=1}^{C} \left[\frac{\left\| m_i - c_k \right\|}{\left\| m_i - c_s \right\|} \right]^{\frac{2}{\mu - 1}} \right), \tag{4}$$

where s are the iteration steps, and by updating the cluster centers c_k:

$$c_k = \left[\sum_{i=1}^{M} {u_{ik}}^{\mu} \cdot m_i \right] \Big/ \left[\sum_{i=1}^{M} {u_{ik}}^{\mu} \right]., \tag{5}$$

The algorithm proceeds as follows. First, the cluster centers are initialized randomly. Thereafter, each reference vector is assigned a membership grade in each cluster. Then the so-called Picard iteration through Eq. (4) and Eq. (5) is run to adjust the cluster centers and the membership values. The iterations will stop when the minimum amount of improvement between two consecutive iterations is less than a small positive number ε or after a specified number of iterations.

We use $\varepsilon = 0.0001$ and a maximum of 100 iterations. The improvement criterion ε is small enough to ensure no possible significant improvements of the possibly local optima, while we never reached 100 iterations. The extent of overlapping between the clusters is set by the fuzzy exponent μ. When $\mu \to 1$, the fuzzy clustering converges to a crisp k-means clustering, while when $\mu \to \infty$ the cluster centers tend towards the center of the data set. Several experiments were performed to set the μ - and c-values.

3 Applications of SOM-FCM for Financial Times Series Analysis

3.1 The Financial Benchmarking Model

The financial benchmarking model was created to perform longitudinal financial performance benchmarking of international pulp and paper companies. The model

consisted of seven financial ratios for the years 1995–2003, for a total of 78 pulp and paper companies. The ratios included were operating margin, return on equity, and return on total assets (profitability ratios), equity to capital and interest coverage (solvency ratios), quick ratio (liquidity ratio), and receivables turnover (efficiency ratio). The model is presented in detail in Eklund et al. [9] and validated in Eklund et al. [14].

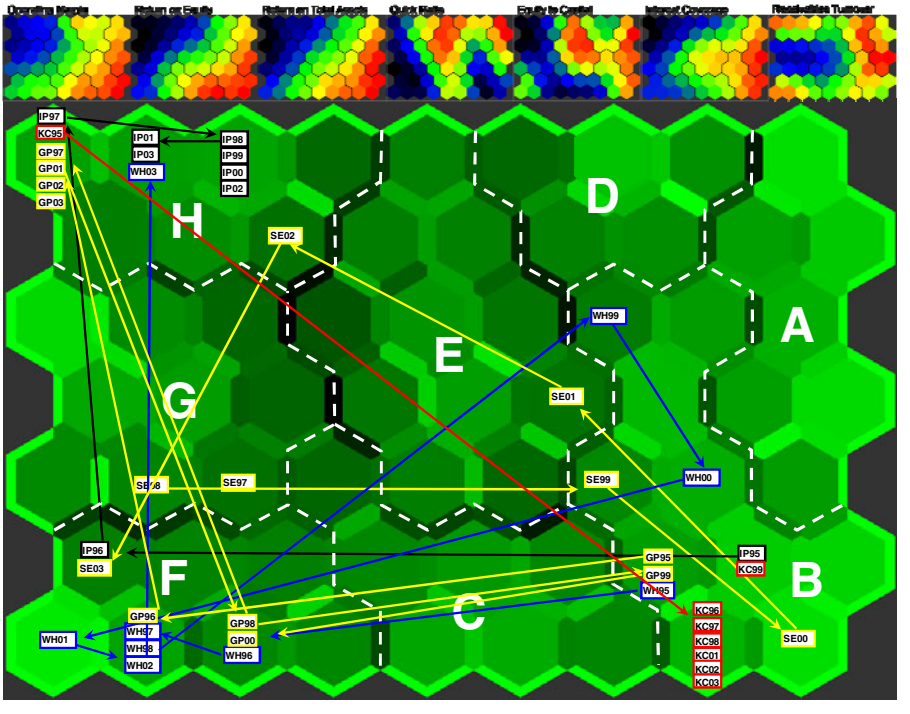


Fig. 1. The financial benchmarking model

The model was created in SOM_PAK 3.1, using randomly initialized reference vectors and sequential training, and visualized in Nenet 1.1. Histogram equalization [15] was used to preprocess the outlier-rich and heavily non-normally distributed financial ratios. The map consists of a 9 x 7 lattice, divided into eight clusters representing different aspects of financial performance, and can be found in Fig. 1. In the figure, the five largest pulp and paper companies according to net sales in 2003 are displayed. The notations are as follows: International Paper = IP, Gerogia Pacific = GP, Stora Enso = SE, Kimberly-Clark = KC, and Weyerhaeuser = WH. The feature planes of the map are displayed at the top of the figure. The map is roughly ordered into high profitability on the right hand side of the map, high solvency and liquidity in the middle and upper right hand side of the map, and high efficiency in upper right hand side, as well as lower and upper left hand sides of the map. Generally speaking, the best in class companies are in clusters A and B, and poorest in clusters G and H.

3.2 Fuzzy Clustering of the Financial Benchmarking Model

The reference vectors from the financial benchmarking model were used as input for a second–level clustering. Several experiments were performed, varying the μ-value (between 1.0 and 3.0) and the *c*-value (between 3 and 9). Based upon these experiments, an μ-value of 2.0 provided the best visual interpretability of the map, introducing a fuzziness degree large enough to show relationships between clusters, but not large enough to completely eliminate cluster borders. The *c*-value was set as 8, in accordance with the originally identified number of clusters on the map. However, different *c*-values were tested, including a three cluster model that roughly divided the companies into good, average, and poor performers. Eight clusters were in this case used in order to be able to assess this clustering in terms of the original model.

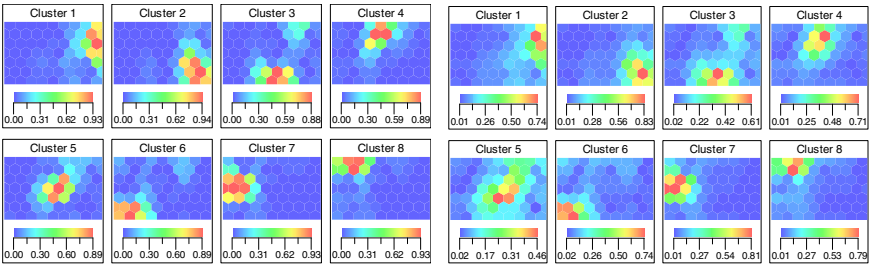


Fig. 2. FCM clustering with (a) μ-value = 1.8, and (b) μ-value = 2.2

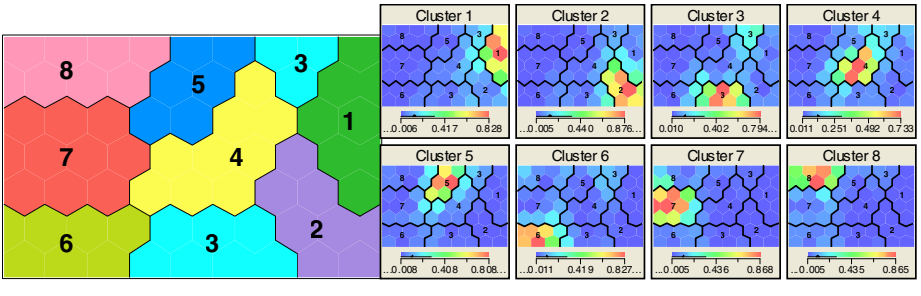


Fig. 3. FCM clustering of the benchmarking model with an μ-value of 2.0 and c-value of 8

Fig. 2 shows a SOMine 5.1 visualization of the nodes' cluster membership degrees for (a) an μ-value of 1.8 and (b) 2.2. It clearly shows the higher crispness of the clusters in (a) vis-à-vis the higher fuzziness of (b). The right map in Fig. 3. shows the nodes' cluster membership degrees for the chosen μ-value of 2.0, while the left shows a defuzzification using maximum memberships. It shows that most of the clusters of the FCM model coincide with the clustering of the original map. For example, the best in class clusters (A and B in Fig. 1) largely coincide with clusters 1 and 2 in Fig. 3.,

only partially overlapping each other and not really any other clusters. The poorest clusters (F, G, and H) are also quite clearly identifiable as clusters 6, 7, and 8 in Fig. 3. The only cluster not identifiable is cluster D, which forms a part of cluster 3 in Fig. 3. The nodes in cluster D thus seem to display similarity to nodes in clusters C, E, and to a degree, cluster F. When using Ward's [16] clustering on the map in Fig. 1, cluster D does indeed merge with cluster C, indicating a slightly twisted map. This is a complement to other methods, such as Sammon's mapping, for testing map twistedness. Further, the FCM clustering shows that cluster E is split into two groups largely based upon liquidity (quick ratio), clusters 4 and 5.

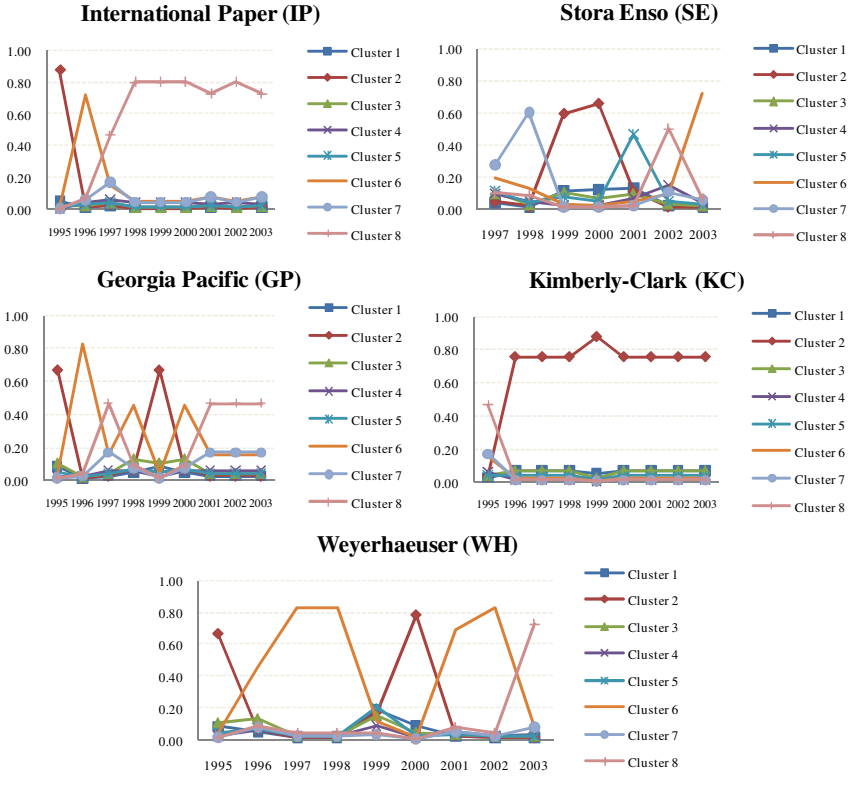


Fig. 4. Membership degrees for the five largest P&P companies in 2003

Fig. 4 shows the cluster membership degrees of the top five pulp and paper companies. The figures depict that the cluster memberships of the most stable companies (KC, best) and (IP, poorest) are high (membership of ca 0.8), while the companies that shift between clusters show low membership values (ca 0.6 or less). Further, Fig. 4 shows that the clusters on the left and the right border of the map overlap to a lesser degree than the clusters in the middle, such as the data point for Weyerhaeuser in 1999. In this particular case, the membership degree does not exceed 0.2 for any of the clusters, indicating no predominant cluster over others. This is

indeed informative when judging the certainty to which the financial performance of a company is categorized to a cluster. To incorporate this type of uncertainty, defuzzification using a threshold on the above utilized maximum-membership method might be advantageous.

3.3 The Currency Crisis Model

The currency crisis model was created for visual monitoring of currency crisis indicators, as is done in [17] on general economic and financial variables. The model consisted of four monthly indicators of currency crises for 23 emerging market economies from 1971:1–1997:12. The indicators were chosen and transformed based on a seminal early warning system created by IMF staff [18]. The indicators included were foreign exchange reserve loss, export loss, real exchange-rate overvaluation relative to trend and current account deficit to GDP. This model is, however, conceptually different from the benchmarking model. Each data point has a class dummy indicating the occurrence of a crisis, pre-crisis or tranquil period. A crisis period is defined to occur when exchange-rate and reserve volatility exceeds a specified threshold, while the pre-crisis periods are defined as 24 months preceding a crisis. The class labels were associated with the model by only affecting the updating of the reference vectors (batch version of Eq. 2), not the choice of the BMU (Eq. 1). Thus, the main purpose of the model is to visualize the evolution of financial indicators to assist the detection of vulnerabilities or threats to financial stability. The model is presented in detail in Sarlin [10] and a model on the same data set is evaluated in terms of out-of-sample accuracy in Sarlin and Marghescu [19]. Moreover, a stand-alone FCM clustering has been applied on a close to similar data set in [20].

The model was created and visualized with Viscovery SOMine 5.1, using the two principle components for initializing the reference vectors and the batch updating algorithm. The contribution of each input is standardized using columnwise normalization by range. However, the effects of extremities and outliers are not eliminated, since a crisis episode is *per se* an extreme event. The map consists of 137 output neurons ordered on a 13 x 11 lattice, divided into four crisp clusters representing different time periods of the currency crisis cycle. The units were clustered using Ward's [16] hierarchical clustering on the associated variables. The map, with a projection of indicators for Argentina from 1971–1997, and its feature planes are shown in Fig. 5. The map is roughly divided into a tranquil cluster on the

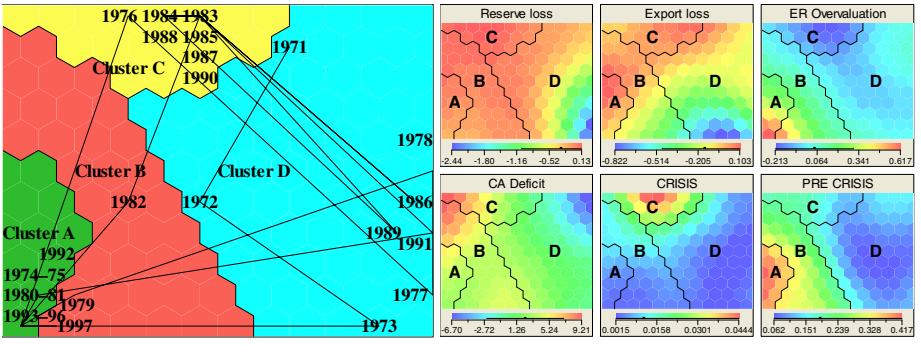


Fig. 5. The currency crisis model with indicators for Argentina from 1971–97

right side of the map (cluster D), a crisis cluster in the upper-left part (cluster C), and a slight early-warning and a pre-crisis cluster in the lower-left part (cluster B and A).

3.4 Fuzzy Clustering of the Currency Crisis Model

Similarly as for the benchmarking model, the reference vectors from the crisis model were used as input for a second–level clustering, whereafter the membership degrees and the defuzzification is visualized in SOMine. The same experiments were performed, varying the μ-value (between 1.0 and 3.0) and the c-value (between 3 and 9). For those models, $\mu = \{1.8, 2.0, 2.2\}$ give the best results; higher fuzzy exponents give non-smooth memberships, while lower give roughly crisp memberships. However, as for the benchmarking model, an μ-value of 2.0 provided the best visual interpretability. The c-value was first set as 4 (Fig. 6), in accordance with the originally identified number of clusters on the map, but later adjusted to 3 (Fig. 7). The concern with the 4-cluster model is that the cluster termed Early warning does not directly contribute to the currency crisis cycle. Although it would, of course, be informative to have an Early warning cluster, the cluster is quite small and borders the pre-crisis cluster both between the tranquil cluster (as desired) and the crisis cluster (as not desired). In Fig. 8, where the vertical dotted lines represent crisis episodes, the fluctuations of indicators for Argentina are shown using both models. This exercise confirms that the Early warning cluster is a less influential cluster that does not add real value to the analysis of the currency crisis cycle. Thus, the 3-cluster model is utilized for assessing the fluctuations in the data.

Argentina experienced three crisis episodes during the analyzed period. As shown in Fig. 8, the first crisis in 1975 was preceded by high membership values in the pre-crisis cluster, whereafter the memberships in the crisis and subsequently the tranquil cluster dominated. The membership values before, during, and after the crisis episode in 1982

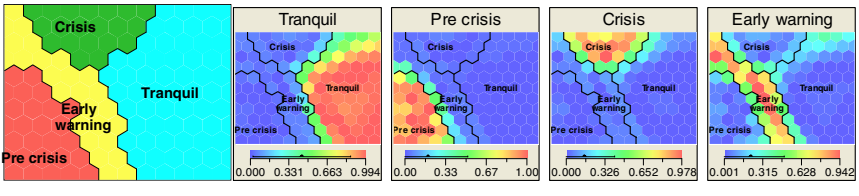


Fig. 6. FCM clustering of the crisis model with an μ-value of 2.0 and c-value of 4

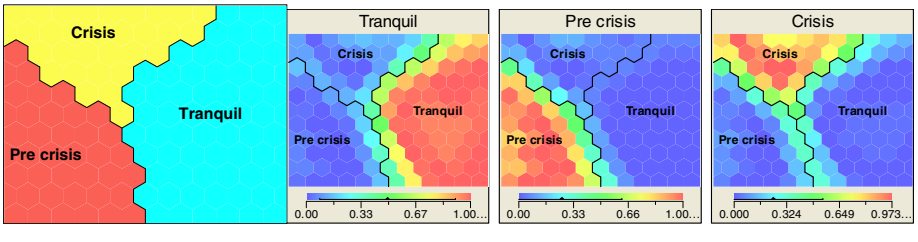


Fig. 7. FCM clustering of the crisis model with an μ-value of 2.0 and c-value of 3

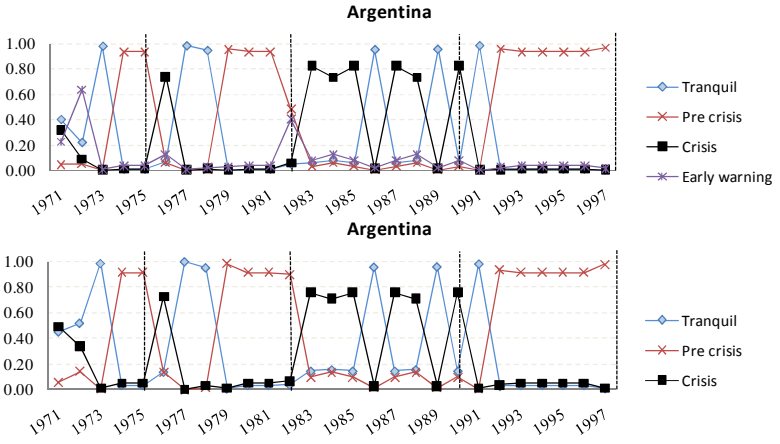


Fig. 8. Membership degrees for Argentina in the 4 and 3 cluster models

similarly characterized a currency crisis cycle. The pre-crisis period for the crisis episode in 1990 is, on the other hand, characterized by abnormal memberships that vary between the tranquil and the crisis cluster, and does thus not resemble the generalization of this model. Further, indications of the out-of-sample crisis episode in 1999 are given already from 1992 onwards.

As evaluating the SOM model's accuracy is not the concern of this paper, and has been done previously, the focus is on the added value of the membership values. The fuzzy clustering in this application is rather crisp, as a comparison of the data points for 1972 and 1982, for example, indicates. The conditions in 1972 and 1982, respectively, are projected into different sides of the border between the pre-crisis and the tranquil cluster, while still having high memberships in their respective cluster centers. The crispness is, however, something that cannot be known *a priori*. Although the clustering is to some extent non-overlapping, the differences within each cluster and between each data point still indicate fluctuations in the conditions.

4 Conclusions

This paper addresses an ambiguity of the SOM clustering; the degree of membership in a particular cluster. To this end, FCM clustering is applied on the units of the SOM grid, allowing each data point to have a partial membership in all identified, but overlapping, clusters. The FCM clustering is applied to two previously presented SOM models for financial time-series analysis. Using FCM clustering, the cluster centers express the representative financial states for the companies and countries, respectively, while the varying membership degrees represent fluctuations of their states over time. The results indicate that fuzzy clustering of the SOM units is a useful addition to visual monitoring and representation of financial time series. However, the clustering still needs to be objectively validated. For this task, there exist cluster validity measures, such as [21–22]; however, this is left for future work.

Acknowledgments

We acknowledge Academy of Finland (grant no. 127656) and *Lars och Ernst Krogius forskningsfond* for financial support. The views in this paper are those of the authors and do not necessarily reflect those of the European Central Bank.

References

1. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics 66, 59–69 (1982)
2. Ultsch, A., Siemon, H.P.: Kohonen's self organizing feature maps for exploratory data analysis. In: Proceedings of the International Conference on Neural Networks, pp. 305–308. Kluwer, Dordrecht (1990)
3. Lampinen, J., Oja, E.: Clustering properties of hierarchical self-organizing maps. Journal of Mathematical Imaging and Vision 2(2–3), 261–272 (1992)
4. Murtagh, F.: Interpreting the Kohonen self-organizing feature map using contiguity-constrained clustering. Pattern Recognition Letters 16(4), 399–408 (1995)
5. Kiang, M.Y.: Extending the Kohonen self-organizing map networks for clustering analysis. Computational Statistics and Data Analysis 38, 161–180 (2001)
6. 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)
7. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11(3), 586–600 (2000)
8. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
9. Eklund, T., Back, B., Vanharanta, H., Visa, A.: Using the Self-Organizing Map as a Visualization Tool in Financial Benchmarking. Information Visualization 2, 171–181 (2003)
10. Sarlin, P.: Visual monitoring of financial stability with a self-organizing neural network. In: Proceedings of the 10th IEEE International Conference on Intelligent Systems Design and Applications, pp. 248–253. IEEE Press, Los Alamitos (2010)
11. Liu, S., Lindholm, C.: Assessing the Early Warning Signals of Financial Crises: A Fuzzy Clustering Approach. Intelligent Systems in Accounting, Finance & Management 14, 179–202 (2006)
12. Kohonen, T.: Self-Organizing Maps. Springer, Berlin (2001)
13. Dunn, J.C.: A Fuzzy Relative of the ISODATA Process and its Use in Detecting Compact, Well-Separated Clusters. Cybernetics and Systems 3, 32–57 (1973)
14. Eklund, T., Back, B., Vanharanta, H., Visa, A.: Evaluating a SOM-Based Financial Benchmarking Tool. Journal of Emerging Technologies in Accounting 5, 109–127 (2008)
15. Guiver, J.P., Klimasauskas, C.C.: Applying Neural Networks, Part IV: Improving Performance. PC AI Magazine 5, 34–41 (1991)
16. Ward, J.: Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association 58, 236–244 (1963)
17. Resta, M.: Early Warning Systems: an approach via Self Organizing Maps with applications to emergent markets. In: Proceedings of the 18th Italian Workshop on Neural Networks, pp. 176–184. IOS Press, Amsterdam (2009)

18. Berg, A., Pattillo, C.: What caused the Asian crises: An early warning system approach. Economic Notes 28, 285–334 (1999)
19. Sarlin, P., Marghescu, D.: Visual Predictions of Currency Crises using Self-Organizing Maps. Intelligent Systems in Accounting, Finance and Management (forthcoming, 2011)
20. Marghescu, D., Sarlin, P., Liu, S.: Early Warning Analysis for Currency Crises in Emerging Markets: A Revisit with Fuzzy Clustering. Intelligent Systems in Accounting, Finance and Management 17(2–3), 143–165 (2010)
21. Bezdek, J.C.: Cluster validity with fuzzy sets. Cybernetics 3, 58–73 (1974)
22. Xie, X.L., Beni, G.: A validity measure for fuzzy clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 13(8), 841–847 (1991)

Self Organizing Maps as Models of Social Processes: The Case of Electoral Preferences

Antonio Neme[1,2,3], Sergio Hernández[2,3], and Omar Neme[4]

[1] Adaptive Informatics Research Centre, Aalto University,
Konemiehentie 2, Espoo, FIN
aneme@cis.hut.fi
[2] Complex Systems Group, Autonomous University of Mexico City
San Lorenzo 290, Mexico City, MEX
[3] Centre for Complex Sciences, National Autonomous University of Mexico, MEX
sergiohz@c3.unam.mx
[4] School of Economics National Polytechnic Institute, Plan de agua Prieta 66
Mexico City, MEX
oneme@ipn.mx

Abstract. We propose the use of self-organizing maps as models of social processes, in particular, of electoral preferences. In some voting districts patterns of electoral preferences emerge, such that in nearby areas citizens tend to vote for the same candidate whereas in geographically distant areas the most voted candidate is that whose political position is distant to the latter. Those patterns are similar to the spatial structure achieved by self-organizing maps. This model is able to achieve spatial order from disorder by forming a topographic map of the external field, identified with advertising from the media. Here individuals are represented in two spaces: a static geographical location, and a dynamic political position. The modification of the later leads to a pattern in which both spaces are correlated.

Keywords: Self-organizing maps; electoral preferences; social sciences and computational models.

1 Introduction

Self-organizing maps (SOM) have been widely applied in several fields, covering visualization [1], time series processing [2], and many others. Here, we propose its use not as a data analysis tool, but as a model of social processes. SOM is, at the end, an algorithm, and in that sense, is not different from other models for social sciences, that are also algorithms. The field of statistical physics has been very productive in proposing models for a wide variety of social phenomena [3]. In that sense, we propose the use of SOM as a model of a specific phenomenon, that of electoral preferences.

Electoral preferences of individuals are dynamic. They may be influenced by the opinion of other voters, the perception they have from candidates, as well as from factors like propaganda from the media, and several other issues. Although

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 51–60, 2011.

citizens tend to vote for those candidate or political parties that reflects more sharply their own ideas, these perceptions may be modified. Electoral preferences have been extensively studied from different angles [3,4,5]. Here, we present a model of a special case of electoral preferences based on SOM.

In some voting districts, there is a correlation between geographical space and the perceived political position of the candidate they voted for. In some cases, whole adjacent regions of cities or countries tend to vote for the same political party or candidate while other, possibly distant regions, tend to vote for other candidates, with a perceived different political position (see fig. 1-a).

One issue that seems to be a fundamental factor in the dynamics of electoral preferences is the impact of the media [6]. Specifically, we refer to the propaganda from the parties and candidates aiming to influence the decision of voters. One way to model this influence is as follows. A party or candidate is described by a point in a high dimensional space, the political position space. This space is defined by several political issues, such as public health, education, foreign affairs, labor issues, environmental policies, etc. Each party is defined by a vector with the relevant political issues. Each voter has an opinion over the same issues, defined by a vector that summarizes his/her political position. Opinion vectors from voters are susceptible of being modified.

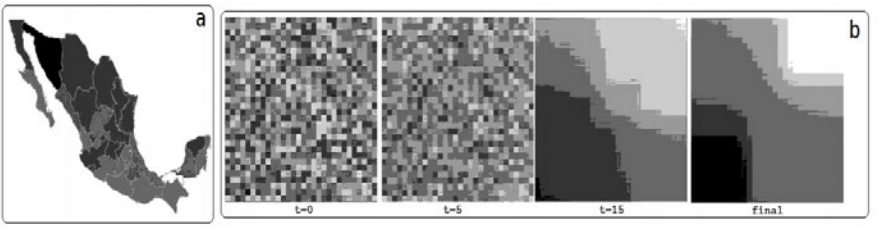


Fig. 1. a) Voting in Mexico. Gray level codes voting percentages for two parties, represented by white and black. North Mexico presents darker levels than those in south Mexico. b) SOM formation for four input vectors and three dimensions. It is observed that in the final map the most distant vectors are located at opposite locations.

When a political party presents an advertising in the media voters may react to it. The main assumption here is that the voters whose political position is closer to the political position that describes the party, will react and modify their opinion in order to get even closer to the position of the party. At the same time, these voters will act as active voters or promoters, affectig other voters within their neighborhood and modify as well their opinion to get closer also to that of the political party. The area of influence of these active voters tend to decrease with time as effect of habituation to active voters [3]. At the end of the process, it is observed that certain regions tend to vote for a certain candidate while distant regions tend to vote for a very different, politically speaking, candidate (see fig. 1-a). The model explains the spatial patterns in which electoral preferences form clusters, known as topographic maps. Although such patterns may be explained

by demographic factors, may also be explained at the light of both, the exposure to an external field (media), and a self-organizing process. The described process is similar to the map formation in SOM.

Voting is a major feature of democratic regimes. For that, it has attracted the attention of the scientific community to study deeply its dynamics. It has been studied from several perspectives, including democratic debates and opinion forming models [3], neighborhood influence from similar voters in the Sznajd model, voting through opinion shift [4], and many others. Electoral preferences are an example of opinion dynamics, widely studied from the social sciences but also from the mathematical-based sciences. Several ideas are common in all models. First, political position of voters is susceptible of being influenced. Second, the aspects that voters take into account for voting are measurable. Third, in some models, the dynamics are internal, that is, opinions are driven only by the actual opinion of some voters. However, in other models, such as in [7,8], the internal opinion dynamics are subject to external influences. The idea behind the external influence is that it is possible to affect some of the voters in order to shift their electoral preference toward some desired option.

SOM has been used as a tool to elucidate patterns in data. A less studied side of SOMs are its capabilities of modeling dynamical systems. By this, we mean it may model certain processes. We intend to use the SOM as a model of a social process. We do not intend to use it as a data analysis tool. That is, we propose the use of SOM as a dynamical system model to study a social phenomena, the spatial pattern formation in some electoral processes. These patterns are associated to patterns observed in SOM.

We are interested in the topographic map formation, a particular case of spatial patterns. A topographic map (TM) is a global structure in a low-dimensional physical media, which is an approximation of the distribution shown by the input stimulus from the multidimensional input space or structure of the external field. In a TM, high-dimensional input vectors that are similar are mapped to close regions in the map, while other, distant vectors are mapped to farther areas. A TM is that in which topology of the input vectors are preserved in the lattice [9]. In a TM, there is a correlation between geographical space and an abstract space, that in this contribution corresponds to the political position space. Voting distribution over a city or country may be the approximation of the distribution of perceived political positions from candidates or parties.

2 The Model

It is important for the social sciences scholars to discover why some specific patterns in the electoral preferences appear. In particular, studying the mechanisms and dynamics that allow the appearance of TM-related maps over voting disticts is a case that has attracted the attention. In this contribution, we study the relevance of external stimulus (media) and the influence voters receive from their peers in order for those patterns to appear. We study those influences with the self-organizing map as a model of electoral preferences.

The SOM is a model of neural connections with the capability of producing organization from disorder [1]. One of the main properties of the SOM is the ability to preserve in the output map those topographical relations present in the input data [1]. This attribute is achieved through the transformation of an incoming analogical signal of arbitrary dimension into a discrete low-dimensional map (usually one or two-dimensional), and by adaptively transforming data in a topologically ordered fashion [1]. Each input data is mapped to a unit or neuron in the lattice, to the one with the closest weight vector to the input vector, or best matching unit (BMU). The SOM preserves neighborhood relationships during training through the learning equation (1), which establishes the effect that each BMU has over any other neuron.

The SOM structure consists, generally, of a two-dimensional lattice of units. Each unit n maintains a dynamic weight vector w_n which is the basic structure for the algorithm to lead to map formation. The dimension of the input space is considered in the SOM by allowing weight vectors to have as many components as features in the input space. Variables defining the input space, and thus, the weight space, are continuous. Weight vectors are adapted accordingly to:

$$w_n(t+1) = w_n(t) + \alpha_n(t) h_n(g,t)(x_i - w_n(t)) \tag{1}$$

where $\alpha(t)$ is the learning rate at epoch t, $h_n(g,t)$ is the neighborhood function from BMU g to unit n at epoch t and x_i is the input vector. The neighborhood decreases monotonically as a function of distance and time [10,11]. Neighborhood is equivalent to a dynamic coupling parameter. In this work, we applied the so-called bubble neighborhood in which units farther than a given distance do not update their weight vector. The SOM preserves relationships in the input data by starting with a large neighborhood and reducing it during the course of training [1].

The SOM algorithm is divided in three stages. 1. Competition: The best matching unit (BMU) g is the one whose weight vector is the closest to the input vector x: $BMU = \arg\min_g ||x - w_g||$ 2. Cooperation: The adaptation is diffused from the BMU g to the rest of the units in the lattice through the learning equation (1). 3. Annealing: The learning parameter and neighborhood are updated.

The map formed by the algorithm is a topographic one. The output map is an approximation of the vectors distribution in the input space. We are not interested in mapping multidimensional signals to a low dimensional space and study the topographic relations, as it is done in several other applications, for example the case of study of parliamentary elections in [13]. We are mainly interested in the spatial pattern of the units when exposed to the input signals or stimulus.

In this model, each unit corresponds to a voter or group of voters within a geographic static area. Units are susceptible of being affected and also influence other units. Although real individuals move around over the city, the discussions about political issues are mainly present within their neighbors, which makes it equivalent to static location. The weight vector associated to each unit is the

position of voters with respect to the relevant political issues, that is, the weight vector defines the position of voters in the political position space. The variables that define this space are continuous and voters may occupy any region on this multidimensional space.

When a BMU affects neighbors, all issues are equally modified accordingly to eq. 1, that is, the position of the affected units is shifted in all dimensions. Also, as the neighborhood function is discrete (bubble), all units within its neighborhood are equally affected. The competition stage is interpreted as the assignment of resources from parties or candidates to those possible voters which may act as promoters. The cooperation stage summarizes the electoral campaign as the influence from promoters or active voters in order to modify the political position of their neighbors. The annealing stage reflects the habituation or refractoriness from voters to the influence of promoters.

When an input stimulus (advertising) is presented, it only affects a single unit. The affected unit corresponds to the BMU in the SOM and this unit will affect its neighbors in order to attract them in the feature space. The feature space corresponds to the political position of both, individuals and parties. So the advertising affects a whole area through the most influenced individual (BMU), regardless of the previous political position of affected units.

The process of self-organizing is iterative and as the neighborhood that BMUs affect tend to zero, the weight vectors reach a steady state. This convergence may represent a TM, if some conditions are satisfied (see fig. 2). First, the initial neighborhood area should be sufficiently large. Second, the neighborhood function should decrease in both, time and space [11,12]. Third, enough epochs should occur [1].

Each unit i has its own weight vector w_i, that defines the opinion of a group of neighbor voters to the relevant issues considered in voting. The position of each party k is defined by the vector p_k. Citizens will vote for that candidate or party to whom they are closer in the space of the considered issues. That is, an unit i is said to vote for party j that:

$$j = \arg\min_k |w_i - p_k| \tag{2}$$

The position a unit has over each item is continuous and defined in the range $[0, 1]$. The items that voters may consider as relevant are the position of the political parties with regard to a vector of political issues. At the same time, parties are defined in the political space by their own position to those aspects. Parties that have a similar position will be represented as closer points in the feature space, while parties with opposite positions will be defined by distant points. Political parties tend to attract as many voters as possible by modifying voters position's towards their own position. Parties are coded as input vectors: the political position of each one is defined as a vector. A party from the right-most wing could be coded as $[0, ..., 0]$ and a party from the left-most wing is $[1, ..., 1]$. In general, parties do not change their position.

In the seminal work of Schelling [14], agents move towards an available location in which the perceived comfort is better than that in the present location. Agents

do not change their opinions, but their location, which leads to a segregation pattern through displacements in geographical space. In the cultural model of Axelrod [15], agents change their opinions (culture) by means of interaction dictated by homophilia and in a stochastic fashion. In this model, cultures or regions of similar individuals are formed, and segregation is observed. In this model culture is defined by discrete variables, and there is a comparison between an individual's opinion and that of their neighbors. In the model we present, there is not a comparison between an individual and her neighbors in order to interact.

Table 1 shows the interpretation of SOM's parameters and attributes in the context of electoral preferences models. The area BMUs affect is decreasing as a function of space and time. Voters have limited presence and resources, which constraint them to promote candidates in distant regions from their own geographical position for a long time. These corresponds to the neighborhood of a BMU. Also, voters decrease their presence as time goes by, which is equivalent to the decreasing neighborhood as a function of time: voters become refractory to active voters as a function of time. Neighborhood function summarizes the mobility of voting promoters, as they may be visiting other areas, but they will stop visiting distant ones as time elapses.

Table 1. Control parameters and variables in the electoral preferences model and its equivalence in SOM

θ	Equivalence in SOM	Description
w	Weight vector	Political position of voters
P	Number of input vectors	Number of political parties or candidates
H	Initial neighborhood area	Area of influence of voters
E	Number of epochs	Political campaigns duration
d	Input space dimension	Number of political issues considered
ρ	Avg. distance between input vectors	Average difference of opinion among parties
δ	Minimum distance between input vectors	Minimum of political differences
ψ	Maximum distance between input vectors	Maximum of political differences
γ	Avg. initial distance between neighbor units	Avg initial diffences among neighbors
P_i	Input vector i	Political position of party i
f_i	No. of copies of each input vector i	Number of advertising of party i per day
α	Learning parameter	Permeability of voters
V_i	Initial number of units closer to P_i	No. of supporters of P_i at the beginning
B	Number of BMUs for each input vector	No. of active voters for each party

The ρ, δ, and ψ parameters refer to the distribution of input vectors. ρ is defined as the average Euclidean distance between all pairs of input vectors: $\rho = 1/P \sum_{i \neq j} d(w_i, w_j)$. These three parameters summarize the political options and are a measure of how radical those positions are.

The control parameter f_i is associated to the amount of advertisement per party per day. A political party is defined by its position to d political issues. The position of party i is P_i, a point in the $d-$dimensional space. Not all the f_i

copies are exactly the same: each of the f_i issues is modified by a random small value σ. The interpretation is that each advertising emphasizes some issues while others are put aside, at the time that some vote voters may get confused about the message or misinterpret some aspects. The set of all f_i defines the input or stimulus space SOM will form a low-dimensional topographic map of that space.

Although there are several observable parameters, we are interested only in one of them: the quality of the topographic map, i.e, how well the maps are formed. The topographic error (TE) is a measure of the quality of the map and is defined as the average number of input vectors whose BMU and second-BMU are not contiguous [9].

In the SOM only one unit is selected as BMU. However, there are a number of variations in which the BMU is not unique [16], and all BMU act simultaneously to modify their neighbors. Here, we included this variation to give the model more plausibility, as a given party may have more than a single active promoter.

The order parameter TE should be featured by the control parameters. It has been stated that good maps (low TE) are achieved if initial neighborhood is large enough and decreasing with time and space, otherwise, local order may be achieved but global order does not emerge, when the initial configuration is random. There are not analytical results about the conditions to achieve good maps [17,1], even though some results are known for very specific cases[17,12]. Thus, we ran a set of experiments in order to characterize the topographic map formation as a function of the control parameters.

In the experiments initial weight vectors are random, which corresponds to a situation in which supporters are randomly distributed in the city. A number of P parties tends to attract as many voters as possible in order to win the elections. Voters are exposed to advertising from the media for a period of E epochs and each party i presents to the voters f_i advertisings per day.

3 Results

To study the electoral preferences dynamics and its organization, a hypothetical city is defined as a lattice of units. It is also analyzed the topological organization achieved as a function of external stimulus (input vectors), but also by means of internal constraints, such as the size of the neighborhood units affect. The external stimulus are defined as the advertising the media presents to voters and those voters that are the closest to the advertising will react to them. Several hundred thousands of maps were formed varying the parameters in table 1. Parameters were randomnly sampled following flat distributions. Figures 2 show the TE as a function of some of the control parameters.

We applied mutual information to determine the relevance of each parameter and TE. It has been identified as a good correlation measure as it is able to find non-linear relations between data [18]. It is defined as the amount of information between possible states of two possible correlated systems X and Y. Mutual information between the parameters in table 1 and TE are shown in fig. 2.

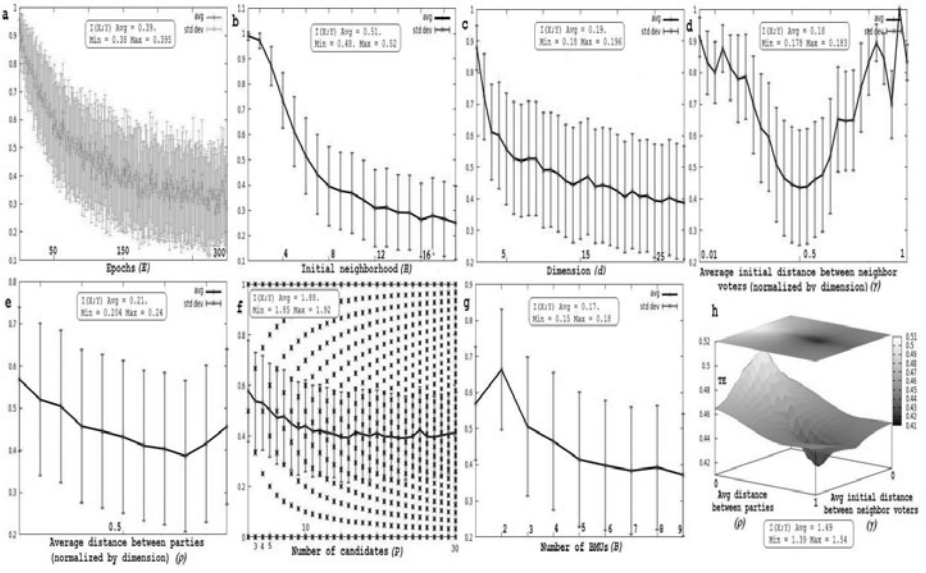


Fig. 2. TE as a function of some control parameters in table 1 for lattice size 20×20. Mutual information between the parameter and TE is shown. Ranges were discretized in 100 intervals. It is shown the average $I(X;Y)$ for the lattice sizes considered, as well as the maximum and minimum $I(X;Y)$. X corresponds to TE whereas Y corresponds to the studied parameter.

From fig. 2-a), it is observed that, as established by analytical results and numerical explorations [17,9], TE decreases as the duration of the process increases. Also (fig. 2-b), as established by theory [1], the tendency is that the larger the initial neighborhood area, the lower the TE. TE is also a decreasing function of the number of political issues considered by voters (fig. 2-c). This is explained by the fact that the more dimensions defining the input space, the easier is to modify the unit's weight vector to unfold and approximate the high-dimensional input vectors [1]. These three parameters have been widely studied, so our simulations in these three parameters adjusted to the established theory and numerical findings.

Besides the results that are predicted by the theory, such as the decreasing TE with time, neighborhood and dimension of input space, other results not previously identified were obtained. We proceed to explain such findings.

In its original version, SOM works with one BMU for input vector. In our model, $B \geq 1$ simultaneous BMUs are present for each input vector. A novel finding is that TE decreases as the number of BMUs increases, with exception of $B = 2$. The more the number of promoters per input vector, the more likely for a TM is to form (fig. 2-g).

As a function of initial average political differences among neighbor voters (γ), TE presents an interesting curve. γ is defined as $1/N^2 \sum_{\forall i \neq j} d(w_i, w_j)$, where $d(a, b)$ is the Euclidean distance between weight vectors a and b. When these

differences are small, or very large, TE tends to be large. This is interpreted as if individuals are very similar, then the map is unable to unfold and represent the input vectors over the lattice. If political positions are too radical, then the TM is not likely to appear (fig. 2-d).

There is a tendency towards low TE as the average distance between parties increases, except for very distant vectors (fig. 2-e). In electoral terms, the more distant the position of political parties, the easier it gets to the voters to polarize and thus, form a TM. The number of candidates or parties P is the parameter with the highest influence over TE ($I(X;Y) = 1.88$). TE is a measure of order as it establishes for each input vector if its two most similar units are adjacent in the lattice. It may happen that for r input vectors ($r \leq P$) the second most BMU is not adjacent to the first, which increases TE. If $r = 0$ then $TE = 0$, while if $r = P$, then TE = 1. If P is even, then it may happen that exactly half of the input vectors are not properly mapped, that is $TE = 0.5$, otherwise, r is not even ($TE \neq 0.5$). TE may take values of $1/P \times i$, $0 \leq i \leq P$ (fig. 2-f).

TE is characterized by the parameters (γ, ρ). In fig. 2-h it is observed that low TE are achieved if ρ is high but not maximum and at the same time γ is approximately half the maximum distance (0.5, as it is normalized). Mutual information between the compound system $X = (\gamma, \rho)$ and $Y = TE$ is 1.49.

4 Conclusions

We propose here the use of self-organizing map (SOM) as a model to study voting processes under the constraints of permeability of voters, influence from parties through an external field, and a decreasing influence from vote promoters. We related parameters and variables in SOM with electoral issues and processes. We have given evidence that the same mechanism that leads to self-organization in SOM may, help to explain the patterns in electoral processes.

In previous models, electoral results over voting districts are not viewed as topographic maps, but at most, as segregation states. Here, we have proposed that electoral results may resemble topographic maps if some constratints are observed. As in every model, the explanation power is limited by the assumptions supporting it. Thus, we present our model as a possible explanation of the observed spatial patterns in some electoral districts under the circumstances here detailed.

In the model, the final results of electoral processes is dictated by the influence between voters, and the external influence of the media. We propose that the election results are guided by the political position of the contenders and the number of their advertising, subject to initial distribution of political preferences of voters. The media exerts a non-linear influence in the spatial pattern formation of voting. For a topographic map to appear, the duration of political campaigns should be large enough. The higher the number of aspects that are considered by voters, the more likely a topographic map is to emerge. If at the beginning voters have more or less the same opinion, or population is radicalized, then it is unlikely that topographic maps will emerge.

References

1. Kohonen, T.: Self-Organizing maps, 3rd edn. Springer, Heidelberg (2000)
2. Barreto, G., Araujo, A.: Identification and control of dynamical using the self-organizing map. IEEE Transactions on Neural Networks 15(5), 1244–1259 (2004)
3. Galam, S.: The dynamics of minority opinions in democratic debates. Physica A 336, 46–62 (2004), doi:10.1016/j.physa.2004.01.010
4. Pabjan, B., Pekalski, A.: Model opinion forming and voting. Physica A 387, 6183–6189 (2008), doi:10.1016/j.physa.2008.07.003
5. Costa, R., Almeida, M., Andrade, S., Moreira, M.: Scaling behavior in a proportional voting process. Phys. Rev. E. 60, 1067–1068 (1999)
6. Tuncay, C.: Opinion Dynamics Driven by Leaders, Media, Viruses and Worms. Int. J. of Modern Physics C 18(5), 849–859 (2007)
7. González-Avella, J., Cosenza, M., Tucci, K.: Nonequilibrium transition induced by mass media in a model for social influence. Phys. Rev. E 72 (2005)
8. Mazzitello, K., Candia, J., Dossetti, V.: Effects of Mass Media and Cultural Drift in a Model for Social Influence. Int. J. Mod. Phys. C 18, 1475 (2007)
9. Villmann, T., Der, R., Herrmann, M., Martinetz, T.: Topology preservation in self-organizing feature maps. IEEE Tr. on NN. 8(2), 256–266 (1997)
10. Flanagan, J.: Sufficiente conditions for self-organization in the SOM with a decreasing neighborhood function of any width. C. of Art. NN. Conf. pub. 470 (1999)
11. Erwin, E., Obermayer, K., Schulten, K.: Self-organizing maps: Ordering, convergence properties and energy functions. Biol. Cyb. 67, 47–55 (1992)
12. Erwin, E., Obermayer, K., Schulten, K.: self-organizing maps: stationary states, metastability and convergence rate. Biol. Cyb. 67, 35–45 (1992b)
13. 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)
14. Schelling, T.: Micromotives and Macrobehavior. W. W. Norton (1978)
15. Axelrod, R.: The dissemination of culture. J. of Confl. Res. 41, 203–226 (1997)
16. Schulz, R., Reggia, J.: Temporally Asymmetric Learning Supports Sequence Processing in Multi-Winner Self-Organizing Maps. Neural Comp. 16(3), 535–561 (2004)
17. Flanagan, J.: Self-organization in the one-dimensional SOM with a decreasing neighborhood. Neural Networks 14(10), 1405–1417 (2001)
18. Celluci, C., Albano, A., Rap, P.: Statistical validation of mutual information calculations. Phys. Rev. E 71, 66208 (2005)

EnvSOM: A SOM Algorithm Conditioned on the Environment for Clustering and Visualization

Serafín Alonso[1], Mika Sulkava[2], Miguel Angel Prada[2], Manuel Domínguez[1], and Jaakko Hollmén[2]

[1] Grupo de Investigación SUPPRESS, Universidad de León, León, Spain
saloc@unileon.es, manuel.dominguez@unileon.es
[2] Department of Information and Computer Science, Aalto University School of Science, Espoo, Finland
mika.sulkava@tkk.fi, miguel.prada@tkk.fi, Jaakko.Hollmen@hut.fi

Abstract. In this paper, we present a new approach suitable for analysis of large data sets, conditioned on the environment. Mainly, the envSOM algorithm consists of two consecutive trainings of the self-organizing map. In the first phase, a SOM is trained using every available variable, but only those which characterize the environment are used to compute the winner unit. Therefore, this phase produces an accurate model of the environment. In the second phase, a new SOM is initialized appropriately with information from the codebooks of the first SOM. The new SOM uses all the variables for winner selection. However, in this case the environmental variables are kept fixed and only the remaining ones are involved in the update process. A model of the whole data set influenced by the environmental conditions is obtained in this second phase. The result of this algorithm represents a probability function of a data set, given the environment information. Therefore, it could be very useful in the analysis of processes which have close dependencies on environmental conditions.

Keywords: Self-organizing maps, variants of SOM, environmental conditions, envSOM, data mining, pattern recognition.

1 Introduction

Many variants of SOM appeared in the literature [1]. The aims of these approaches comprise improvements in clustering, visualization, accuracy of the model, computation time, etc. For instance, it is possible to define different neighborhood functions, change the winner searching process, and introduce some a priori information about classes or states. An overview of the main ideas which can be used to modify the standard SOM is presented in [2].

These variants have brought great advantages for data analysis, but, so far, none of them has been focused on the data analysis conditioned on the environment. It is well known that environmental conditions influence strongly most of the real processes and systems. Furthermore, it is generally desirable to compare

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 61–70, 2011.

data from different processes whose environmental conditions are the same. For these reasons, a new algorithm, the envSOM, is proposed in this paper. It still captures the behavior of the processes, but takes into account the model of the environment.

This paper is structured as follows: In Section 2, several approaches related to the envSOM are reviewed briefly. In Section 3, the envSOM algorithm and its two phases are explained in detail. Two examples used to test the algorithm are described in Section 4. Also, the results obtained using the envSOM are shown there. Finally, the conclusions are drawn in Section 5.

2 Similar Approaches

Several approaches, already presented in the literature, are described below. Although they have some similarities with the proposed algorithm, they also have essential differences.

In the *Supervised SOM* [3], the main idea is to modify the traditional unsupervised SOM into a supervised algorithm by adding information about class-identity in the learning process. For that purpose, the input vectors, $x = [x_s, x_u]$, consist of two different parts. The first one, x_s, corresponds to the input data and the second one, x_u, is related to the class of the sample [4]. In order to visualize the map, the second part is pruned out. Classification is enhanced using this method since the second part is the same for input vectors of the same class. It could be necessary to weight the values of the second part to achieve a better accuracy in the classification. The proposed envSOM algorithm does not depend on a class-identity variable, but the input vectors comprise two rather different parts, the environmental variables and the others.

The *Tree-Structured SOM* (TS-SOM) consists of several traditional SOMs organized hierarchically in several layers, i.e., a pyramid-like structure is obtained where the lower SOMs are larger [5,6]. Firstly, training will take place at the higher levels. Codebooks from these SOMs are kept fixed and then, the training continues at the subsequent layers according to the hierarchy. The differences appear in both search and update steps. The winner searching process is performed on the units at the same layer and the neighbors on the higher level according to the hierarchy. In the update step, only the units at the same layer are updated, keeping fixed the units at the higher level. Therefore, this variant is computationally quite inexpensive whereas the envSOM comprises two consecutive SOM trainings.

The *PicSOM* algorithm is based on several TS-SOMs [7]. It was proposed for retrieving images similar to a given reference image from a database. A separate TS-SOM is used for each kind of feature vectors extracted from the images (color, texture and shape). The responses from individual TS-SOMs are combined automatically according to user's preferences. The PicSOM approach provides a robust method for using a set of image maps in parallel. The envSOM algorithm also uses a set of special variables (characterizing the environment) to cluster data.

In the *Layering SOM*, a SOM is trained for each individual layer to achieve better results in the field of exploratory analysis. In this sense, a growing hierarchical SOM has been presented in [8]. That work explains a dynamic model which adapts its architecture in the training process and uses more units where more input data are projected. The major benefits of this approach are the reduction of training time due to the concept of layers, the possibility to discover a hierarchical structure of the data, the improvement of cluster visualization by displaying small maps at each layer and the preservation of topological similarities between neighbors. This algorithm allows us to visualize data in detail, but the models obtained are not conditioned on the environment.

The self-organizing map has also been used for time series processing in the form of the *Temporal SOM*. In order to exploit the temporal information, SOM needs to be enabled with a short-term memory, which can be implemented, e.g., through external tapped delay lines or different types of recurrence. Several of these extensions are reviewed in [9,10]. In our approach, no short-term memory is explicitly implemented, but it is usually advisable to introduce time information in the model to analyze the temporal evolution, together with the environment.

3 The envSOM Algorithm

The purpose of this work is to develop an algorithm suitable for extracting and analyzing information from large data sets, but considering the environmental information such as weather variables, atmospheric deposition, etc. The envSOM approach consists of two consecutive phases based on the traditional SOM [2]. Some slight variations have been introduced in each phase. The winner searching process in the first phase and the update process in the second one have been modified appropriately in order to achieve the desired result. In our experiments the learning rate decreases in time and the neighborhood function is implemented as Gaussian in both phases. However, other functions could be used as well.

The proposed envSOM algorithm has the advantageous features of the traditional SOM. Likewise, it reaches spatially-ordered and topology-preserving maps. It also provides a good approximation to the input space, similar to vector quantization, and divides the space in a finite collection of Voronoi regions. The main innovation of this algorithm is that it reflects the probability density function of data set, given the environmental conditions. Therefore, it can be useful from the point of view of environmental pattern recognition and data comparison, conditioned on these patterns. On the contrary, it should be noted that it will be more expensive computationally compared to the traditional SOM, since two learning phases are needed. Furthermore, it requires knowledge of the environmental variables which influence the behavior of the process, characterized by the remaining variables. The envSOM approach will be explained in detail below.

3.1 The First Phase

In the first phase of the envSOM algorithm, a traditional SOM is trained using all variables. The initialization can be either linear along the greatest eigenvectors or

random, depending on the user's preference. It is necessary to know in advance which are the environmental variables, since only these variables will be used for computing the winner neurons. For this reason, the other variables must be masked in the winner searching process. Similarly to the traditional SOM, the winner c is selected using equation 1.

$$c(t) = \arg\min_i \|\mathbf{x}(t) - \mathbf{m}_i(t)\|_\omega \ , i = 1, 2, \ldots, N \tag{1}$$

where $\mathbf{x}$ represents the current input and $\mathbf{m}$ denotes the codebook vectors. N and t are, respectively, the number of the map units and the time. The difference is that a binary mask is always used to indicate which variables are used for computing the winner. As usual, if the Euclidean norm, $\|^\circ\|$, is chosen, the winner will be computed using equation 2, where ω is the binary mask and k is a component or variable.

$$\|\mathbf{x}(t) - \mathbf{m}_i(t)\|_\omega^2 = \omega\|\mathbf{x}(t) - \mathbf{m}_i(t)\|^2 = \sum_k \omega_k \left[x_k(t) - m_{ik}(t)\right]^2 \tag{2}$$

The mask, ω, is a k-dimensional vector whose values ω_k are 1 or 0, depending on if the component corresponds to an environmental variable or not. The update rule has not been modified and therefore, it is similar to the traditional SOM.

The result obtained from this phase will be a map where only the components related to the environment are organized. The remaining components do not affect the organization. The aim of this phase is to achieve a model which represents the environment in the best possible way. Moreover, the values of the remaining components will be used for initialization in the second phase. It should be remarked that although this initialization seems completely random, it has proven to be good and the values lie in the range of the variables.

3.2 The Second Phase

In the second phase of the envSOM algorithm, a new traditional SOM is trained using all variables. It will be initialized using the codebooks from the first phase SOM. Thanks to this appropriate initialization, a fast convergence of the algorithm is reached and an accurate model which defines the environment will be used in the second phase. It should be noted that environmental components have been already organized in the first phase of the envSOM. Therefore, values from the codebooks of first SOM are a good starting point for the second phase.

In this case, every component will take part equally in the winner computation and no mask will be applied. Unlike the first phase, the update process is now slightly modified. As environmental variables are already well organized, it is only required that the remaining variables are updated properly. For this reason, a new mask is introduced in the update rule and equation 3 will be used in this case. The mask, $\mathbf{\Omega}$, is a k-dimensional vector which takes binary values Ω_k, i.e., 0 if it corresponds to an environmental variable and 1 otherwise. k is the number of components or variables.

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \alpha(t)h_{ci}(t)\mathbf{\Omega}[\mathbf{x}(t) - \mathbf{m}_i(t)] \tag{3}$$

At the end of this phase, all variables will be organized properly. The learning rate, $\alpha(t)$, and the neighborhood function, $h_{ci}(t)$, are not modified so that a value decreasing in time and a Gaussian function could be used, respectively, like in the traditional SOM. The purpose of this phase is to reach a good model of the whole data set, given environmental information.

4 Experiments and Results

Two kinds of experiments have been planned in order to test the envSOM algorithm. First, an artificial data set based on binary patterns is created. It allows us to check the clustering property of the algorithm. Then, a simulated data set characterizing climate and carbon flux in several ecosystems is studied. It allows us to check the usefulness of the algorithm with more realistic data and compare the behavior of carbon in different ecosystems, given environmental conditions.

Matlab software has been used to make the experiments and the SOM Toolbox [11] has been modified to implement the necessary changes, such as a new mask in the update process.

4.1 A Toy Example

An artificial data set with structured data has been used to test the envSOM algorithm. The data set consists of 16000 samples and 4 variables (X1, X2, X3, X4). It contains all binary patterns from (0, 0, 0, 0) to (1, 1, 1, 1), i.e., the numbers from 0 to 15 in binary system. A low level of noise (10%) has been added to the variables. Each binary pattern is equally represented by a set of 1000 samples. There are 16 different patterns, so the envSOM algorithm should find 16 clusters in this data set. The choice of this data set is justified by the simple structure of the data, which facilitates the visualization and understanding of the results from the algorithm.

First, a traditional SOM was trained using this input data set. The number of epochs in the training should be high enough in order to guarantee a complete organization. A number over 500 epochs was chosen. The dimensions of SOM were 16×20 (320 units). A Gaussian function was selected as the neighborhood function and a value decreasing exponentially in time as the learning rate. The SOM should be able to divide the data into 16 clusters and allows us to visualize them, for instance, by means of the U-matrix representation. The results from the traditional SOM can be seen in Figure 1. After the training, the U-matrix yields a clear visualization of the binary patterns. Note that each component has been organized in a random way, as it is shown by the component planes. If a new traditional SOM is trained using another data set Y, also based on binary patterns, i.e., (Y1, Y2, Y3, Y4), the organization of the four components will probably be completely different. Thus, it will be very difficult to make a good comparison between the results from both data sets, X and Y.

When there are environmental conditions in the data set, it can be desirable that these components define the organization of the map. In this case, it is supposed that X1 and X2 are the environmental variables and X3 and X4 are

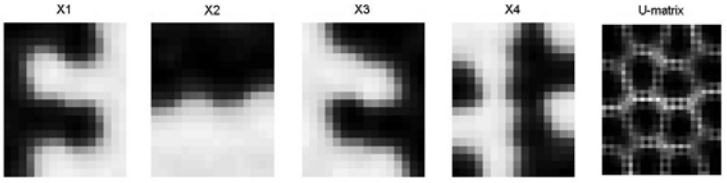


Fig. 1. Component planes and U-matrix of traditional SOM for binary patterns. Black color corresponds to values of 0 and white color to 1.

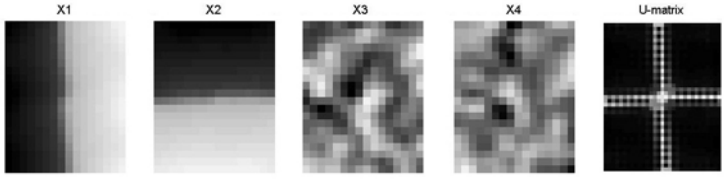


Fig. 2. Component planes and U-matrix of envSOM algorithm after the first phase of learning

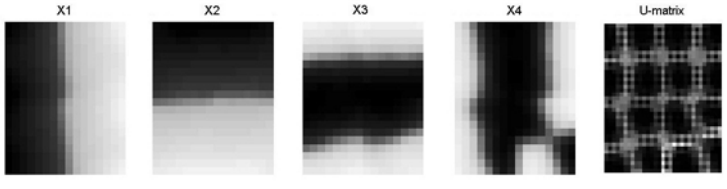


Fig. 3. Component planes and U-matrix of envSOM algorithm after the second phase of learning

features of the data set to be analyzed and compared. The envSOM algorithm consists of two consecutive SOMs as mentioned above. The parameters of both SOMs are the same as in the traditional SOM (500 epochs, 320 units, Gaussian neighborhood function and learning rate decreasing exponentially).

In the first phase, only X1 and X2 variables are used to compute the winner neurons and all variables are updated. The results of the first phase can be seen in Figure 2. As expected, the organization is only performed on variables X1 and X2 since X3 and X4 do not take part in the winner computation. Therefore, the U-matrix only represents four patterns corresponding to possible combinations of variables X1 and X2.

In the second phase, all four variables are used in the winner computation, but X1 and X2 are kept fixed whereas X3 and X4 are updated. At the end of this phase, the data set is organized as depicted in Figure 3. In this case, the 16 patterns can be clearly distinguished in the U-matrix in a similar way to the traditional SOM. Moreover, the organization of the map conditioned on X1 and X2, i.e., the environmental variables, is achieved. It can be said that the envSOM algorithm represents the probability function of data, given the

environmental variables. A comparison of quality of the traditional SOM and the proposed algorithm has been done. The mean quantization error is 0.1364 for the traditional SOM and 0.1717 for the envSOM.

If an envSOM is trained using another binary data set, e.g., Y (Y1, Y2, Y3, Y4), components Y3 and Y4 will be conditioned on the first ones, Y1 and Y2, as expected. The organization of the maps from data sets X and Y can be different and therefore the comparison is difficult. However, the result after the first phase of the envSOM with the data set X can be used to organize Y3 and Y4 in the same way that X3 and X4, respectively. Now, it will be very easy to compare the results from both data sets, X and Y. Furthermore, when components Y1 and Y2 are the same as X1 and X2 because they represent the common environmental conditions, the first phase of the envSOM can be trained jointly with the variables X1, X2, X3, X4, Y3, Y4. The second phase of the envSOM can be carried out with an individual SOM for each data set or one SOM containing all variables from both data sets. The first approach can be applied in any case but, whenever the number of variables and data sets is low enough, the second approach will provide similar results. In those cases, the second choice might be preferred, since it requires fewer computations.

4.2 O-CN Example

A more realistic scenario for presenting the performance of the envSOM approach was performed by analyzing data containing environmental characteristics and simulated gross primary production (GPP, the amount of carbon sequestrated in photosynthesis) of different ecosystems in Europe. The SOM has been previously used for analysis of carbon exchange of ecosystems in, e.g., [12,13]. The GPP estimates used in this study has been generated by the O-CN model [14,15]. The model is developed from the land surface scheme ORCHIDEE [16], and has been extended through representation of key nitrogen cycle processes. O-CN simulates the terrestrial energy, water, carbon, and nitrogen budgets for discrete tiles (i.e. fractions of the grid cell) occupied by up to 12 plant functional types (PFTs) from diurnal to decadal timescales. The model can be run on any regular grid, and is applied here at a spatial resolution of 0.5×0.5. Values of the model input variables: air temperature, precipitation, shortwave downward flux, longwave downward flux, specific humidity, and N deposition and simulated GPP from 1996 to 2005 were used in this example. These values were analyzed for four PFTs: temperate needle-leaved evergreen forests (TeNE), temperate broadleaved seasonal forests (TeBS), temperate grasslands (TeH), and temperate croplands (TeH crop).

The envSOM algorithm was compared with the traditional SOM in this example. First, four traditional SOMs were trained for four PFTs using environmental data and GPP estimates. As example, the component planes and U-matrices of SOMs of two PFTs, temperate broadleaved seasonal forests and temperate grasslands, are shown in Figures 4 and 5. The organization of the two maps characterizing the two PFTs is very different from each other, so it is very laborious to compare them. If one tries to compare the magnitudes of GPP in

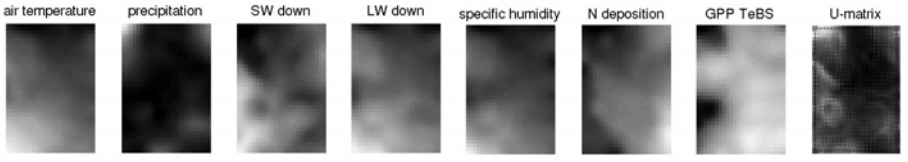


Fig. 4. Component planes and U-matrix of traditional SOM for temperate broadleaved seasonal forests

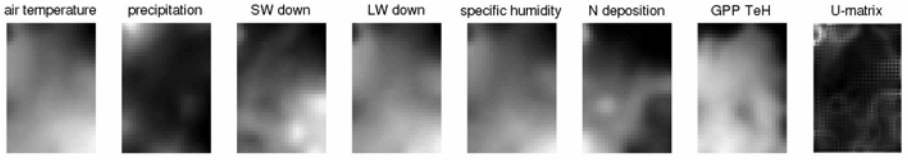


Fig. 5. Component planes and U-matrix of traditional SOM for temperate grasslands

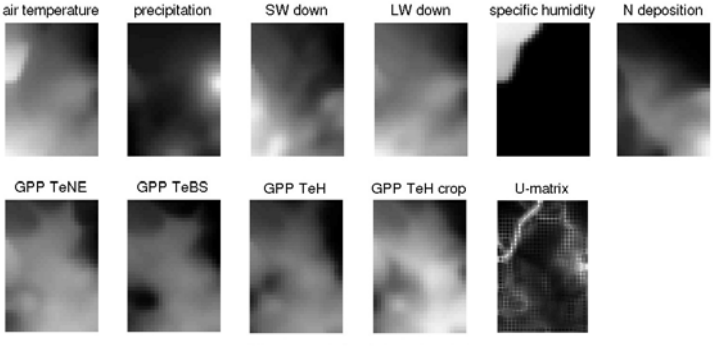


Fig. 6. Component planes and U-matrix of envSOM algorithm for four PFTs

different PFTs connected to a certain combination of environmental variables, spotting the corresponding locations on the maps is not straightforward. The organization of the SOMs of the two PFTs not shown was also different from the other PFTs. The four PFTs were used to train an envSOM with six environmental variables and four GPPs. In the first phase, the environmental variables (air temperature, precipitation, shortwave downward flux, longwave downward flux, specific humidity, and N deposition) were used for training. In the second phase, the variables affected by the environment, i.e., four GPPs were trained. Figure 6 shows the obtained component planes and the U-matrix.

When using envSOM for comparing the PFTs as shown in Figure 6, the commonalities and differences in the connections between environmental variables and GPP can be spotted with ease among the PFTs. The qualitative behavior of the PFTs seems to be rather similar, i.e., relatively high and low GPP values are

usually found in the same regions of the map and are thus, connected with similar environmental conditions. This similarity between the PFTs was expected. However, the absolute values of GPP are different. There are also some differences visible between the PFTs. E.g., the map units with the highest precipitation have very low GPP values for all PFTs except the temperate croplands. In addition, the area in the lower left part of the map associated with relatively high temperature, shortwave and longwave downward fluxes, low precipitation and high GPP in temperate needle-leaved evergreen forests, temperate grasslands, and temperate croplands contains very low values of GPP in temperate broadleaved seasonal forests. The reason for these differences may be different spatial distribution of the PFTs and that some spatially correlated confounding factors have an effect on GPP. More detailed investigation of the reasons behind the differences might be a topic of a future study.

5 Conclusions

In this paper the envSOM algorithm, which is conditioned on the environment was introduced. It consists of two phases based on the traditional SOM. The envSOM has similar features to the traditional SOM in clustering and visualization, although it adds an innovation very useful for finding patterns conditioned on the environment in large data sets. The main innovation of the envSOM is that it represents the data set given the environmental conditions. Therefore, the algorithm is suitable for data analysis of real processes strongly influenced by the environment. On the contrary, it is slightly more expensive computationally, since two consecutive SOMs are trained. The proposed algorithm has been satisfactorily tested using a binary data sets and environmental data and simulated carbon flux estimates of four plant functional types. This algorithm yields similar results in a round of different trainings with the same data set. The environmental variables are always organized in a similar way and the others are conditioned on the first ones. Incremental training is possible using the envSOM, i.e., new features or even data samples could be added to the training later, while keeping the environmental variables the same.

Acknowledgments. We thank Sönke Zaehle for providing us with the O-CN data for this study and insightful comments regarding the analysis of the data.

References

1. Kangas, J., Kohonen, T., Laaksonen, J.: Variants of self-organizing maps. IEEE Transactions on Neural Networks 1, 93–99 (1990)
2. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (1995)
3. Hagenbuchner, M., Tsoi, A.C.: A supervised training algorithm for self-organizing maps for structures. Pattern Recognition Letters 26, 1874–1884 (2005)
4. Melssen, W., Wehrens, R., Buydens, L.: Supervised Kohonen networks for classification problems. Chemometrics and Intelligent Laboratory Systems 83, 99–113 (2006)

5. Koikkalainen, P., Oja, E.: Self-organizing hierarchical feature maps. In: International Joint Conference on Neural Networks, vol. 2, pp. 279–284. IEEE, INNS (1990)
6. Koikkalainen, P.: Progress with the tree-structured self-organizing map. In: Cohn, A.G. (ed.) 11th European Conference on Artificial Intelligence, ECCAI (1994)
7. Laaksonen, J., Koskela, M., Laakso, S., Oja, E.: PicSOM - content-based image retrieval with self-organizing maps. Pattern Recognition Letters 21, 1199–1207 (2000)
8. Rauber, A., Merkl, D., Dittenbach, M.: The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. IEEE Transactions on Neural Networks 13(6), 1331–1341 (2002)
9. Hammer, B., Micheli, A., Sperduti, A., Strickert, M.: Recursive self-organizing network models. Neural Networks 17, 1061–1085 (2004)
10. Guimarães, G., Sousa-Lobo, V., Moura-Pires, F.: A taxonomy of self-organizing maps for temporal sequence processing. Intelligent Data Analysis (4), 269–290 (2003)
11. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: SOM toolbox for Matlab 5 (2000)
12. Abramowitz, G., Leuning, R., Clark, M., Pitman, A.: Evaluating the performance of land surface models. Journal of Climate 21(21), 5468–5481 (2008)
13. Luyssaert, S., Janssens, I.A., Sulkava, M., Papale, D., Dolman, A.J., Reichstein, M., Hollmén, J., Martin, J.G., Suni, T., Vesala, T., Loustau, D., Law, B.E., Moors, E.J.: Photosynthesis drives anomalies in net carbon-exchange of pine forests at different latitudes. Global Change Biology 13(10), 2110–2127 (2007)
14. Zaehle, S., Friend, A.D.: Carbon and nitrogen cycle dynamics in the o-cn land surface model: 1. model description, site-scale evaluation, and sensitivity to parameter estimates. Global Biogeochemical Cycles 24 (February 2010)
15. Zaehle, S., Friend, A.D., Friedlingstein, P., Dentener, F., Peylin, P., Schulz, M.: Carbon and nitrogen cycle dynamics in the o-cn land surface model: 2. role of the nitrogen cycle in the historical terrestrial carbon balance. Global Biogeochemical Cycles 24 (February 2010)
16. Krinner, G., Viovy, N., de Noblet-Ducoudre, N., Ogee, J., Polcher, J., Friedlingstein, P., Ciais, P., Sitch, S., Prentice, I.C.: A dynamic global vegetation model for studies of the coupled atmosphere-biosphere system. Global Biogeochemical Cycles 19 (February 2005)

Spectral Clustering as an Automated SOM Segmentation Tool

Kadim Taşdemir

European Commission Joint Research Centre,
Institute for Environment and Sustainability,
Via E. Fermi 2749, 21027, Ispra, Italy
kadim.tasdemir@jrc.ec.europa.eu

Abstract. A powerful method in knowledge discovery and cluster extraction is the use of self-organizing maps (SOMs), which provide adaptive quantization of the data together with its topologically ordered lower-dimensional representation on a rigid lattice. The knowledge extraction from SOMs is often performed interactively from informative visualizations. Even though interactive cluster extraction is successful, it is often time consuming and usually not straightforward for inexperienced users. In order to cope with the need of fast and accurate analysis of increasing amount of data, automated methods for SOM clustering have been popular. In this study, we use spectral clustering, a graph partitioning method based on eigenvector decomposition, for automated clustering of the SOM. Experimental results based on seven real data sets indicate that spectral clustering can successfully be used as an automated SOM segmentation tool, and it outperforms hierarchical clustering methods with distance based similarity measures.

1 Introduction

The self-organizing maps (SOMs) provide topology preserving mapping of high-dimensional data manifolds onto a lower-dimensional rigid lattice. This enables informative SOM visualization of the manifold, which can be used for interactive cluster extraction. Various SOM visualization schemes (see [1,2] and references therein) have been proposed; two of which stand out: U-matrix [3] (which shows Euclidean distances between SOM neighbors on the grid) and CONNvis [2] (which draws detailed local data distribution as a weighted Delaunay graph on the grid). However, the interactive process for cluster extraction from SOM visualization often requires practiced knowledge to evaluate visualized SOM information and hence is difficult for inexperienced users, and time consuming even for the experienced users. Therefore, automated SOM segmentation methods, which are generally hierarchical agglomerative clustering (HAC) schemes with different distance measures, have been proposed. Centroid linkage is considered in [4] with SOM lattice neighborhood, in [5] with a gap criterion; Ward's measure is used in [6]; a recent similarity measure based on distance and density (proposed in [7]) in [8]. These approaches accurately extract the clusters when

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 71–78, 2011.

they are well separated; however, they may be inefficient for extracting complex cluster structures. Another approach [9] uses a recursive flooding of a Gaussian surface (Clusot surface) constructed based on pairwise distances and receptive field sizes of SOM prototypes. However, the resulting partitionings are similar to that of k-means clustering. A recent approach [10] also uses HAC method but with a similarity measure (CONN linkage) based on weighted Delaunay graph of SOM units, which represents detailed local data distribution. It is shown in [10] that CONN linkage is very successful, when the SOM units are dense enough.

A clustering method, becoming more popular due to its high performance and easy implementation, is spectral clustering, which is a graph partitioning approach based on eigenvector decomposition (see [11] and references therein). Due to its advantageous properties, such as extraction of irregular-shaped clusters and obtaining globally optimal solutions [12,13], we propose to use spectral clustering as an automated SOM segmentation method. A preliminary use of spectral clustering for SOM segmentation was considered in [14], but specifically for data categorization to obtain semantically meaningful categories. Here, our experimental results on seven real data sets show that spectral clustering produces better partitionings compared to the ones obtained by other methods in this study. Section 2 briefly explains the spectral clustering, Section 3 discusses experimental results on the data sets and Section 4 concludes the paper.

2 Spectral Clustering

Spectral clustering methods [11,15,16] depend on relaxed optimization of graph-cut problems, using a graph *Laplacian* matrix, L. Let $G = (V, S)$ be a weighted, undirected graph with nodes V representing n points in $X = \{x_1, x_2, \ldots, x_n\}$ to be clustered and edges defined by $n \times n$ similarity matrix S, where s_{ij} is often described using (Euclidean) distance, $d(x_i, x_j)$, between x_i and x_j, as

$$s_{ij} = \exp(-\frac{d^2(x_i, x_j)}{2\sigma^2}) \tag{1}$$

with σ as a scale parameter determining the pairwise similarities. Let D be the diagonal matrix denoting the degree of n nodes where $d_i = \sum_j s_{ij}$. Then $L = D - S$. It is shown in [15] that the use of eigenvector decomposition of the normalized Laplacian matrix

$$L_{norm} = D^{-1/2} L D^{-1/2} = D^{-1/2}(D - S)D^{-1/2} = I - D^{-1/2} S D^{-1/2} \tag{2}$$

can achieve an approximate solution to the normalized cut. Ng et al. [16] extend the solution to extract k groups using the k eigenvectors of L_{norm}[16] $= D^{-1/2} S D^{-1/2}$ with the k highest eigenvalues, by the following algorithm:

1. Calculate similarity matrix S using (1), diagonal degree matrix D, and normalized Laplacian L_{norm}[16]
2. Find the k eigenvectors $\{e_1, e_2, \ldots, e_k\}$ associated with the k highest eigenvalues $\{\lambda_1, \lambda_3, \ldots, \lambda_k\}$

3. Construct the $n \times k$ matrix $E = [e_1 e_2 \dots e_k]$ and obtain $n \times k$ matrix U by normalizing the rows of E to have norm 1, i.e. $u_{ij} = \frac{e_{ij}}{\sqrt{\sum_k e_{ik}^2}}$
4. Cluster the n rows of U with the k-means algorithm into k clusters.

Selection of correct scale parameter σ is important for accurate partitioning. σ can be set manually or can be selected among many values which achieves least distorted partitioning of U [16]. Alternatively, a local scaling parameter σ_i for each node can be calculated based on the k nearest neighbor distance [17],

$$\sigma_i = d(s_i, s_k),\ s_k : \textit{the kth nearest neighbor of } s_i \tag{3}$$

to automatically set σ from intrinsic data details, and to reflect local statistics. Detailed information on spectral clustering can be found in [11,12].

3 Experimental Results

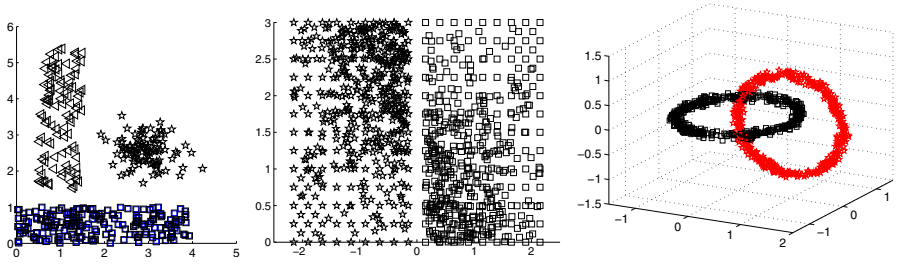
The spectral clustering method as an automated SOM segmentation method, were evaluated using three synthetic data sets from [18], six benchmark data sets from UCI machine learning repository [19], and a large data set, Boston remote sensing test bed, used in [20]. This remote sensing data set can be downloaded from http://techlab.bu.edu/resources/data_view/boston_remote_sensing_testbed/. We compared the performance of the spectral clustering to the performances of the hierarchical agglomerative clusterings with different linkages based on distances (average linkage, centroid linkage, Ward's measure) and with a linkage based on local data distribution (CONN linkage), and to the performance of the k-means clustering. In our experiments, the number of clusters for each data set is assumed to be known. For calculation of local σ_i in (3), $k = 7$ is used.

We obtained the SOMs by Matlab SOMtoolbox, using a rectangular lattice, Gaussian neighborhood, sequential learning and long training. We used one grid structure (10x10) for synthetic data sets, two different grids (5x5 and 10x10) for small UCI data sets, and four sizes (10x10, 20x20, 30x30 and 40x40) for medium or large sized data sets. Due to the randomness in the spectral clustering and in the k-means clustering, each SOM is clustered 50 times by these methods, and clustering accuracies (percentage of the correctly clustered samples) are calculated by averaging over these 50 runs. Contrarily, the partitioning obtained by hierarchical agglomerative clustering methods is unique for each SOM. In addition, since the SOM learning has also randomness (due to the initialization, random selection of samples in training), experiments were repeated 10 times to obtain 10 different SOMs for each data set and for each grid size, and average accuracies were calculated. The average accuracies are given and discussed below. We note that the statistical variances of the accuracies are not provided due to the fact that they have insignificant differences among clustering methods.

3.1 Synthetic Data Sets

Figure 1 shows the synthetic data sets (Lsun, Wingnut, Chainlink) used in the study. Despite a few number of clusters with clear separation among them, clustering of these data sets is challenging due to their specific properties: Lsun has

two rectangular clusters close to each other and a spherical cluster; Wingnut has varying within-cluster density distributions; whereas Chainlink has clusters which are linearly inseparable. For all these data sets, the best partitionings are achieved by CONN linkage [10], where similarities are defined by detailed local density distribution, with average accuracies very close to 100% (Table 1). Spectral clustering with optimal σ is the runner up for the three data sets, and produces significantly better partitionings than k-means and hierarchical clustering with distance based linkages, despite the challenges in these data sets. The use of local σ, however, achieves accuracies similar to the accuracies of k-means.



(a) (b) (c)

Fig. 1. Three synthetic data sets in [18]. The cluster labels of the data samples are shown by different symbols. (a) Lsun (b) Wingnut (c) Chainlink.

Table 1. Accuracies for SOM clustering of synthetic data sets in [18]. SC is spectral clustering, SC1 is with optimal σ (σ is 0.3, 0.1, 0.2 for Lsun, Wingnut, and Chainlink respectively), SC2 is with local σ; whereas average, centroid, Ward and CONN represent different (dis)similarity measures used in hierarchical agglomerative clustering. The best accuracy for each data set is shown in boldface.

Data set	# of samples	# of clusters	SOM clustering method						
			SC1	SC2	k-means	average	centroid	Ward	CONN
Lsun	400	3	98.2	84.00	81.22	87.75	91.30	95.20	**99.58**
Wingnut	1016	2	98.57	95.76	95.94	96.72	98.25	95.08	**99.91**
Chainlink	1000	2	95.38	67.61	66.73	75.78	78.10	78.48	**97.82**

3.2 Data Sets from UCI Machine Learning Repository

We used six data sets from UCI machine learning repository [19]. Three of them (Iris, Wine, Breast Cancer Wisconsin) are relatively small. Iris has 150 4-dimensional samples equally distributed into 3 groups; Wine data set has 178 13-dimensional samples (59, 71, 48 samples in 3 classes respectively); and Breast Cancer Wisconsin has 699 9-dimensional samples grouped into two classes (benign or malignant). The other three data sets (Image Segmentation, Statlog,

Table 2. Accuracies for SOM clustering of UCI data sets with small sizes [19]. SC2 is spectral clustering with local σ. The best accuracy obtained for each SOM size is shown in boldface and the best accuracy for each data set is underlined.

Data set	# of samples	# of clusters	SOM size	SOM clustering method					
				SC2	k-means	average	centroid	Ward	CONN
Iris	150	3	5x5	57.11	85.92	78.40	76.40	**88.00**	83.07
			10x10	**88.92**	84.57	83.27	83.80	86.60	57.27
Wine	178	3	5x5	61.80	67.43	66.52	66.52	**70.00**	69.72
			10x10	**69.54**	68.62	61.01	59.94	65.84	45.84
Breast Cancer-W	699	2	5x5	95.50	95.06	94.90	94.90	94.90	**96.14**
			10x10	**96.16**	95.90	95.82	95.74	95.68	84.39

Table 3. Accuracies for SOM clustering of UCI data sets with medium sizes [19]. SC2 is spectral clustering with local σ. The best accuracy obtained for each SOM size is shown in boldface and the best accuracy for each data set is underlined.

Data set	# of samples	# of clusters	SOM size	SOM clustering method					
				SC2	k-means	average	centroid	Ward	CONN
Segmentation	2391	7	10x10	51.94	51.06	45.63	42.18	53.63	**61.26**
			20x20	**55.96**	52.40	36.71	29.21	53.08	34.26
			30x30	**54.39**	51.99	29.67	29.03	53.13	41.73
			40x40	**54.06**	52.30	28.97	28.96	52.01	21.47
Statlog	6435	6	10x10	58.54	62.44	61.49	53.60	**69.19**	68.96
			20x20	**73.34**	63.52	53.89	52.88	56.41	69.01
			30x30	**73.94**	63.75	52.90	51.64	60.53	58.71
			40x40	**73.91**	63.84	53.72	51.39	62.50	29.32
Pen Digits	10992	10	10x10	50.36	64.03	62.96	62.44	67.66	**73.70**
			20x20	66.61	65.25	64.33	59.81	64.82	**74.40**
			30x30	66.76	66.36	62.84	59.91	67.73	**70.55**
			40x40	68.01	66.86	64.70	56.53	**68.67**	20.08

Pen digits) are of medium sizes. Segmentation has 2310 samples with 19 features grouped into 7 classes, Statlog has 6435 samples with 4 features divided into 6 classes, and Pen Digits has 10992 samples with 16 features in 10 classes. Further details on the data sets can be found in the UCI machine learning repository.

Table 2 and Table 3 show the resulting accuracies for SOM clustering of these data sets. Out of these six data sets, spectral clustering with local σ produces the best partitioning for three of them (Iris, Breast Cancer Wisconsin and Statlog), CONN linkage for two data sets (Segmentation, Pen digits) and Ward's measure

for one data set (Wine). Even though it may be possible to achieve better partitioning by using the optimum σ for each data set, it requires multiple runs to find the optimum value unique for each data set. However, even the use of local σ in spectral clustering often outperforms other methods (in this study) using distance based similarity measures.

3.3 Boston Remote Sensing Data Set

This data set describes a remotely sensed area with 360x600 pixels, resulting in 216000 samples, where each sample has 41 features. There are eight classes: beach, ocean, ice, river, road, park, residential, industrial. 29,003 samples, which are labeled as one of the eight classes, represent the ground truth information to be used in the accuracy assessment. The performance of spectral clustering with local σ is quite high (93.83%), outperforming all other methods in the study.

Table 4. Accuracies for SOM clustering of Boston data set used in [20]. SC2 is spectral clustering with local σ. The best accuracy obtained for each SOM size is shown in boldface and the best accuracy for each data set is underlined.

Data set	# of samples	# of clusters	SOM size	SOM clustering method					
				SC2	k-means	average	centroid	Ward	CONN
Boston	216000	8	10x10	53.95	88.81	88.69	90.32	**92.54**	87.02
			20x20	**93.82**	92.17	91.48	92.97	90.17	87.81
			30x30	93.32	92.35	92.05	**93.56**	91.67	90.28
			40x40	**93.39**	92.41	93.06	93.38	92.28	89.13

3.4 Computational Complexity

Spectral clustering has a time complexity of $O(n^3)$ and a space complexity of $O(n^2)$ [12], where n is the number of patterns (SOM prototypes in this study) to be clustered. In contrast, k-means clustering has a time and space complexity of $O(nkl)$ (l is the number of iterations) and $O(k)$, whereas HAC methods have a time and space complexity of $O(n^2 \log n)$ and $O(n^2)$ respectively [21]. Moreover, with recent efficient algorithms [22], time complexity of HAC methods can be reduced to $O(n^2)$. However, due to the fact that spectral clustering often produces relatively more accurate partitioning (as shown above), its greater computational complexity is hardly a concern in clustering the SOM, since the number of SOM prototypes is usually chosen as at most a couple of thousands.

4 Conclusions

In this paper we used spectral clustering as a SOM segmentation method. The SOM produces a topology preserving mapping, while its spectral clustering produces eigenvector decomposition of the distance based similarity matrix of SOM

units, which is another mapping. This method extracted clusters accurately, even in the case of nonlinear separation boundary. It also outperformed hierarchical agglomerative clustering and k-means clustering, for SOM clustering of the data sets used in this study. The success of two consecutive mapping indicates that SOMs may be used as an initial vector quantization method to make spectral clustering able to partition large data sets where it is not possible to use spectral clustering due to its computational complexity.

References

1. Vesanto, J.: SOM-based data visualization methods. Intelligent Data Analysis 3(2), 111–126 (1999)
2. Taşdemir, K., Merényi, E.: Exploiting data topology in visualization and clustering of Self-Organizing Maps. IEEE Transactions on Neural Networks 20(4), 549–562 (2009)
3. Ultsch, A.: Self-organizing neural networks for visualization and classification. In: Lausen, O.B., Klar, R. (eds.) Information and Classification-Concepts, Methods and Applications, pp. 307–313. Springer, Heidelberg (1993)
4. Murtagh, F.: Interpreting the Kohonen self-organizing map using contiguity-constrained clustering. Pattern Recognition Letters 16, 399–408 (1995)
5. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11(3), 586–600 (2000)
6. Cottrell, M., Rousset, P.: The Kohonen algorithm: A powerful tool for analyzing and representing multidimensional quantitative and qualitative data. In: Cabestany, J., Mira, J., Moreno-Díaz, R. (eds.) IWANN 1997. LNCS, vol. 1240, pp. 861–871. Springer, Heidelberg (1997)
7. Halkidi, M., Vazirgiannis, M.: A density-based cluster validity approach using multi-representatives. Pattern Recognition Letters (6), 773–786 (2008)
8. Wu, S., Chow, W.: Clustering of the self-organizing map using a clustering validity index based on inter-cluster and intra-cluster density. Pattern Recognition (37), 175–188 (2004)
9. Brugger, D., Bogdan, M., Rosenstiel, W.: Automatic cluster detection in Kohonen's SOM. IEEE Transactions on Neural Networks 19(3), 442–459 (2008)
10. Taşdemir, K., Milenov, P.: An automated SOM clustering based on data topology. In: Proc. 18th European Symposium on Artificial Neural Networks (ESANN 2010), Bruges, Belgium, D-Facto, April 27-30, 2010, pp. 375–380 (2010)
11. von Luxburg, U.: A tutorial on spectral clustering. Technical Report TR-149, Max Planck Institute for Biological Cybernetics (March 2007)
12. Wang, L., Leckie, C., Ramamohanarao, K., Bezdek, J.: Approximate spectral clustering. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS, vol. 5476, pp. 134–146. Springer, Heidelberg (2009)
13. Zhang, X., Jiao, L., Liu, F., Bo, L., Gong, M.: Spectral clustering ensemble applied to SAR image segmentation. IEEE Transactions on Geoscience and Remote Sensing 46(7) (July 2008)
14. Saalbach, A., Twellmann, T., Nattkemper, T.W.: Spectral clustering for data categorization based on self-organizing maps. In: Nasrabadi, N.M., Rizvi, S.A. (eds.) SPIE Proceedings, Applications of Neural Networks and Machine Learning in Image Processing IX, vol. 5673, pp. 12–18 (2005)

15. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
16. Ng, A., Jordan, M., Weiss, Y.: On spectral clustering: analysis and an algorithm. In: Dietterich, T., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems 14, MIT Press, Cambridge (2002)
17. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. In: Advances in Neural Information Processing Systems (2004)
18. Ultsch, A.: Maps for the visualization of high-dimensional data spaces. In: WSOM, vol. 3, pp. 225–230 (2003)
19. Asuncion, A., Newman, D.: UCI machine learning repository (2007)
20. Carpenter, G.A., Martens, S., Ogas, O.J.: Self-organizing information fusion and hierarchical knowledge discovery: a new framework using ARTMAP neural networks. Neural Networks 18(3), 287–295 (2005)
21. Jain, A., Murty, M.N., Flynn, P.: Data clustering: A review. ACM Computing Surveys 31(3), 264–323 (1999)
22. Christopher, D., Manning, P.R., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)

Sparse Functional Relevance Learning in Generalized Learning Vector Quantization

Thomas Villmann and Marika Kästner

University of Applied Sciences Mittweida,
Dep. of Mathematics, Natural and Computer Sciences
Technikumplatz 17,09648 Mittweida, Germany
{villmann,kaestner}@hs-mittweida.de

Abstract. We propose a functional relevance learning for learning vector quantization of functional data. The relevance profile is taken as a superposition of a set of basis functions depending on only a few parameters compared to standard relevance learning. Moreover, the sparsity of the superposition is achieved by an entropy based penalty function forcing sparsity.

Keywords: functional vector quantization, relevance learning, information theory.

1 Introduction

During the last years prototype based models became one of the widely used paradigms for clustering and classification. Thereby, different strategies are proposed in classification. Whereas support vector machines (SVMs) emphasize the class borders by the support vectors while maximizing the separation margin, the family of learning vector quantization (LVQ) algorithms is motivated by class representative prototypes to achieve high classification accuracy. Based on the original but heuristically motivated standard LVQ introduced by Kohonen [3] several more advanced methods were proposed. One key approach is the generalized LVQ (GLVQ) suggested by Sato&Yamada [10], which approximates the accuracy by a differentiable cost function to be minimized by stochastic gradient descent. This algorithm was extended to deal with metric adaptation to weight the data dimensions according to their relevance for classification [1]. Usually, this relevance learning is based on weighting the Euclidean distance, and, hence, the data dimensions are treated independently leading to large number of weighting coefficients, the so-called relevance profile, to be adapted in case of high-dimensional data. If the data dimension is huge, as it is frequently the case for spectral data or time series, the relevance determination may become crucial and instable. However, functional data have in common that the vectors can be seen as discrete realizations of functions, i.e. the vectors are so-called functional data. For those data the index of the vector dimensions is a representative of the respective independent function variable, i.e. frequency, time or position etc. In this sense the data dimensions are not longer uncorrelated.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 79–89, 2011.

The aim of the relevance learning method here is to make use of this interpretation. Then, the relevance profile can be also assumed as a discrete representation of a relevance function. We suggest to approximate these relevance functions as a superposition of only a few basis functions depending on a drastically decreased number of parameters compared to the huge number of independent relevance weights. We call this algorithm *Generalized Functional Relevance LVQ* (GFRLVQ). Further, we propose the integration of a sparseness criterion for minimizing the number of needed basis functions based on an entropy criterion resulting in *Sparse GFRLVQ* (S-GFRLVQ).

The paper structure is: After a short review of the GLVQ and GRLVQ schemes we introduce the GFRLVQ followed by S-GFRLVQ. The experiment section shows the abilities of the new algorithms for illustrative data sets.

2 Functional Relevance Learning for GLVQ

In this section we first briefly review standard relevance learning in GLVQ. Thereafter, we turn over to the functional aspect of relevance learning.

2.1 Relevance Learning in GLVQ – GRLVQ

As mentioned before, GLVQ is an extension of standard LVQ based on an energy function E approximating the accuracy. Given a set $V \subseteq \mathbb{R}^D$ of data vectors $\mathbf{v}$ with class labels $x_\mathbf{v} \in \mathcal{C} = \{1, 2, \ldots C\}$, the prototypes $\mathbf{w} \in W \subset \mathbb{R}^D$ with class labels y_j $(j = 1, \ldots, N)$ should be distributed in such a way that they represent the data classes as accurately as possible. In particular, the following cost function is minimized

$$E(W) = \frac{1}{2} \sum_{\mathbf{v} \in V} f(\mu(\mathbf{v})) \text{ with } \mu(\mathbf{v}) = \frac{d^+(\mathbf{v}) - d^-(\mathbf{v})}{d^+(\mathbf{v}) + d^-(\mathbf{v})} \tag{1}$$

where f is a monotonically increasing function usually chosen as sigmoidal or the identity function. The function $\mu(\mathbf{v})$ is the classifier function where $d^+(\mathbf{v}) = d(\mathbf{v}, \mathbf{w}^+)$ denotes the distance between the data vector $\mathbf{v}$ and the closest prototype $\mathbf{w}^+$ with the same class label $y_{\mathbf{w}^+} = x_v$, and $d^-(\mathbf{v}) = d(\mathbf{v}, \mathbf{w}^-)$ is the distance to the best matching prototype $\mathbf{w}^-$ with a class label $y_{\mathbf{w}^-}$ different from $x_\mathbf{v}$. The similarity measure $d(\mathbf{v}, \mathbf{w})$ is supposed differentiable with respect to the second argument but not necessarily to be a mathematical distance. More general similarity measure could be considered. One possible choice is the standard Euclidean distance or its weighted counterpart

$$d_{\boldsymbol{\lambda}}(\mathbf{v}, \mathbf{w}) = \sum_{i=1}^{D} \lambda_i (v_i - w_i)^2 \tag{2}$$

with relevance weights restrictions

$$\lambda_i \geq 0 \text{ and } \sum_i \lambda_i = 1. \tag{3}$$

The vector $\boldsymbol{\lambda}$ is called relevance profile.

Learning in GLVQ of $\mathbf{w}^+$ and $\mathbf{w}^-$ is done by stochastic gradient descent[1] learning with respect to the cost function $E(W)$ according to

$$\frac{\partial_S E(W)}{\partial \mathbf{w}^+} = \xi^+ \cdot \frac{\partial d^+}{\partial \mathbf{w}^+} \text{ and } \frac{\partial_S E(W)}{\partial \mathbf{w}^-} = \xi^- \cdot \frac{\partial d^-}{\partial \mathbf{w}^-}$$

with $\xi^+ = f' \cdot \frac{2 \cdot d^-(\mathbf{v})}{(d^+(\mathbf{v}) + d^-(\mathbf{v}))^2}$, $\xi^- = -f' \cdot \frac{2 \cdot d^+(\mathbf{v})}{(d^+(\mathbf{v}) + d^-(\mathbf{v}))^2}$ and f' denotes the first derivative of f. Relevance learning in this model can be performed by adaptation of the relevance weights again by gradient descent learning:

$$\frac{\partial E_S(W)}{\partial \lambda_j} = \xi^+ \cdot \frac{\partial d^+_{\boldsymbol{\lambda}}}{\partial \lambda_j} + \xi^- \cdot \frac{\partial d^-_{\boldsymbol{\lambda}}}{\partial \lambda_j} \,. \tag{4}$$

The respective algorithm is named Generalized Relevance LVQ – GRLVQ [1]. Yet, in this model the relevance weights as well as the vector components are treated independently as it is the natural way in the Euclidean distance or their weighted variant.

2.2 Functional Relevance Learning

As we have seen, the data dimensions are handled independently in GRLVQ. This leads to a huge number of relevance weights to be adjusted, if the data vector are really high-dimensional which is the case in many applications. For example, processing of hyperspectral data frequently requires the consideration of hundreds or thousands of spectral bands; time series may consist of a huge number of time steps. This huge dimensionality may lead to unstable behavior of relevance learning in GRLVQ. However, if the data vectors are discrete representations of functions, relevance learning can make use of this functional property to reduce the number of parameters in relevance learning.

More precisely, we assume in the following that data vectors $\mathbf{v} = (v_1, \ldots, v_D)^T$ are representations of functions $v_i = v(t_i)$. Then the relevance profile can be interpreted as a function $\lambda(t)$ with $\lambda_j = \lambda(t_j)$, too. In the recently proposed *generalized functional relevance* LVQ (GFRLVQ) [12], now the relevance function $\lambda(t)$ is supposed to be a superposition $\lambda(t) = \sum_{l=1}^{K} \beta_l \mathcal{K}_l(t)$ of simple basis functions $\mathcal{K}_l$ depending on only a few parameters with the restrictions $\beta_i \geq 0$ and $\sum_{l=1}^{K} \beta_l = 1$. Famous examples are Gaussians or Lorentzians:

$$\mathcal{K}_l(t) = \frac{1}{\sigma_l \sqrt{2\pi}} \exp\left(-\frac{(t - \Theta_l)^2}{2\sigma_l^2}\right) \text{ and } \mathcal{K}_l(t) = \frac{1}{\eta_l \pi} \frac{\eta_l^2}{\eta_l^2 + (t - \Theta_l)^2}$$

Now, relevance learning takes place by adaptation of the parameters β_l, Θ_l,σ_l and η_l, respectively. For this purpose, again a stochastic gradient scheme is applied. For an arbitrary parameter ϑ_l of the dissimilarity measure d we have

$$\frac{\partial_S E}{\partial \vartheta_l} = \xi^+ \cdot \frac{\partial d^+}{\partial \vartheta_l} + \xi^- \cdot \frac{\partial d^-}{\partial \vartheta_l}$$

[1] Here, the stochastic gradient operator $\frac{\partial_S E(W)}{\partial \mathbf{w}^\pm}$ is carried out by taking the gradient $\frac{\partial f(\mu(\mathbf{v}))}{\partial \mathbf{w}^\pm}$ for a stochastically chosen $\mathbf{v}$ according to the data distribution $\mathcal{P}(V)$.

Using the convention $t_j = j$ we get in the case of Gaussians for the weighting coefficient β_l, the center Θ_l and the width σ_l for

$$\frac{\partial d(\mathbf{v},\mathbf{w})}{\partial \beta_l} = \frac{1}{\sigma_l\sqrt{2\pi}} \sum_{j=1}^{D} \exp\left(-\frac{(j-\Theta_l)^2}{2\sigma_l^2}\right) \cdot (v_j - w_j)^2 \tag{5}$$

$$\frac{\partial d(\mathbf{v},\mathbf{w})}{\partial \Theta_l} = \frac{\beta_l}{\sigma_l^3\sqrt{2\pi}} \sum_{j=1}^{D} (j-\Theta_l) \cdot \exp\left(-\frac{(j-\Theta_l)^2}{2\sigma_l^2}\right) \cdot (v_j - w_j)^2 \tag{6}$$

$$\frac{\partial d(\mathbf{v},\mathbf{w})}{\partial \sigma_l} = \frac{\beta_l}{\sigma_l^2\sqrt{2\pi}} \sum_{j=1}^{D} \left(\frac{(j-\Theta_l)^2}{\sigma_l^2} - 1\right) \cdot \exp\left(-\frac{(j-\Theta_l)^2}{2\sigma_l^2}\right) \cdot (v_j - w_j)^2 \tag{7}$$

whereas for the Lorentzian we obtain

$$\frac{\partial d(\mathbf{v},\mathbf{w})}{\partial \beta_l} = \frac{1}{\pi} \sum_{j=1}^{D} \frac{\eta_l}{\eta_l^2 + (j-\Theta_l)^2} (v_j - w_j)^2 \tag{8}$$

$$\frac{\partial d(\mathbf{v},\mathbf{w})}{\partial \Theta_l} = \frac{\beta_l}{\pi} \sum_{j=1}^{D} \frac{2\eta_l (j-\Theta_l)}{\left(\eta_l^2 + (j-\Theta_l)^2\right)^2} (v_j - w_j)^2 \tag{9}$$

$$\frac{\partial d(\mathbf{v},\mathbf{w})}{\partial \eta_l} = \frac{\beta_l}{\pi} \sum_{j=1}^{D} \frac{(j-\Theta_l)^2 - \eta_l^2}{\left(\eta_l^2 + (j-\Theta_l)^2\right)^2} (v_j - w_j)^2 \tag{10}$$

After $\boldsymbol{\beta}$-update a renormalization is necessary to ensure the restrictions (3). Instabilities may occur if the center locations Θ_l, Θ_k become very similar for $l \neq k$. To avoid this phenomena a penalty term

$$P_R = \sum_{l=1}^{K} \sum_{m=1}^{K} \exp\left(-\frac{(\Theta_m - \Theta_l)^2}{2\xi_l\xi_m}\right)$$

is added to the cost function (1), which now reads as $E_{GFRLVQ} = E(W) + \varepsilon_r P_R$ with a properly chosen penalty weight $\varepsilon_R > 0$. For Gaussian basis functions we set $\xi_k = \sigma_k$, and for the Lorentzians we take $\xi_k = \eta_k$. The penalty can be interpreted as a repulsion with an influence range determined by the local correlations $\xi_l\xi_m$. The resulting additional update term for Θ_l-learning is

$$\frac{\partial P}{\partial \Theta_l} = \sum_{m=1}^{K} \frac{(\Theta_m - \Theta_l)}{\xi_l\xi_m} \exp\left(-\frac{(\Theta_m - \Theta_l)^2}{2\xi_l\xi_m}\right)$$

leading to a minimum spreading of the basis function centers Θ_l. Analogously, an additional term occurs for the adjustemts of the ξ_l according to $\frac{\partial P_R}{\partial \xi_l}$, which has to be taken into account for the update of σ_k and η_k for Gaussians and Lorentzians, respectively.

3 Sparse GFRLVQ – The S-GFRLVQ Model

In the GFRLVQ model the number K of basis functions can be freely chosen so far. Obviously, if the K-value is too small, an appropriate relevance weighting is impossible. Otherwise, a K-value too large complicates the problem more than necessary. Hence, a good way to choose K is needed. This problem can be seen as sparseness in functional relevance learning. A common methodology to judge sparsity is the information theory. In particular, the Shannon entropy H of the weighting coefficients $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_K)$ can be applied. Maximum sparseness, i.e. minimum entropy, is obtained, iff $\beta_l = 1$ for exactly one certain l whereas the other β_m are equal to zero. However, maximum sparseness may be accompanied by a decreased classification accuracy and/or increased cost function value E_{GFRLVQ}.

To achieve an optimal balancing, we propose the following strategy: The cost function E_{GFRLVQ} is extended to

$$E_{S-GFRLVQ} = E_{GFRLVQ} + \gamma(\tau) \cdot H(\boldsymbol{\beta}) \tag{11}$$

with τ counting the adaptation steps. Let τ_0 be the final time step of the usual GFRLVQ-learning. Then $\gamma(\tau) = 0$ for $\tau < \tau_0$ holds. Thereafter, $\gamma(\tau)$ is slowly increased in an adiabatic manner [2], such that all parameters can immediately follow the drift of the system. An additional term for β_l-adaptation occurs for non-vanishing $\gamma(\tau)$-values according to this new cost function (11):

$$\frac{\partial E_{S-GFRLVQ}}{\partial \beta_l} = \frac{\partial E_{GFRLVQ}}{\partial \beta_l} + \gamma(\tau) \frac{\partial H}{\partial \beta_l} \tag{12}$$

with $\frac{\partial H}{\partial \beta_l} = -(\log(\beta_l) + 1)$. This term triggers the $\boldsymbol{\beta}$-vector to become sparse. The adaptation process is stopped, if the E_{GFRLVQ}-value or the classification error shows a significant increase compared to that at time τ_0.

4 Experiments

We tested the GFRLVQ for the classification of two well known real world spectral data sets obtained from StatLib and UCI: *The Tecator data set*, [11], consists of 215 spectra obtained for several meat probes. The spectral range is between 850nm−1050nm wavelength (100 spectral bands). The data are labeled according to the two fat levels (low/high). *The Wine data set*, [8], contains 121 absorbing infrared spectra of wine between wavenumbers $4000\text{cm}^{-1} - 400\text{cm}^{-1}$ (256 bands) split into 91 training and 30 test data. The data are classified according to their two alcohol levels (low/high) as given in [5].

For the Tecator data we started the simulations with 10 Gaussians to describe the relevance profile and trained the model using GFRLVQ. The achieved accuracy was above 88%. Thereafter the penalty term $\gamma(\tau)$ in (11) and accordingly in (12) is continuously increased in an adiabatic manner such that the system is pushed to become sparse. As depicted in Fig.1, the increasing the penalty term

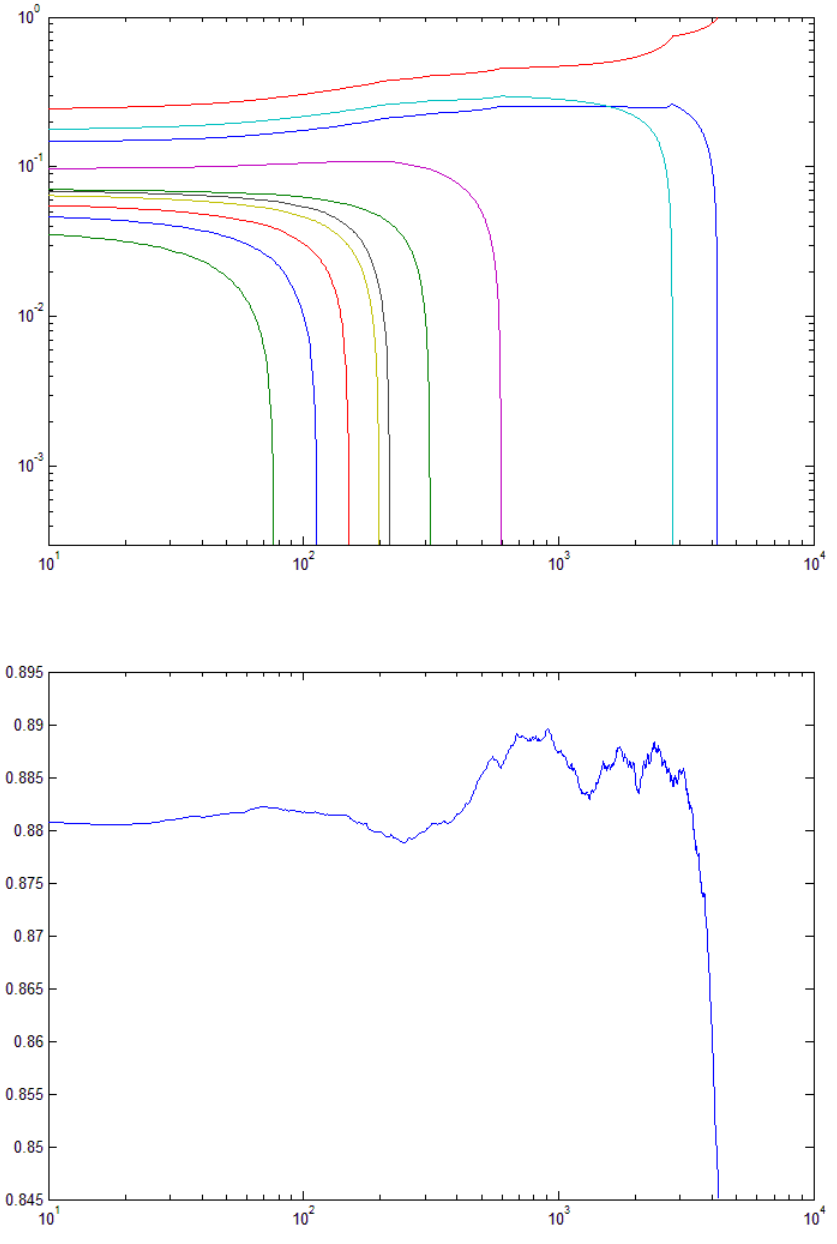


Fig. 1. Time development of the S-GFRLVQ for the Tecator data. In the beginning there are 10 Gaussian basis functions with weighting coefficients β_l obtained from usual GFRLVQ. Increasing the penalty term $\gamma(\tau) \cdot H(\boldsymbol{\beta})$ leads to sparsity in the $\boldsymbol{\beta}$-vector (top, β_l-weights in dependence of the increasing sparsity weight $\gamma(\tau)$ through learning time τ). If the sparsity constraint does not dominate, a high accuracy is still available. If the sparsity becomes too high, accuracy significantly drops down (bottom).

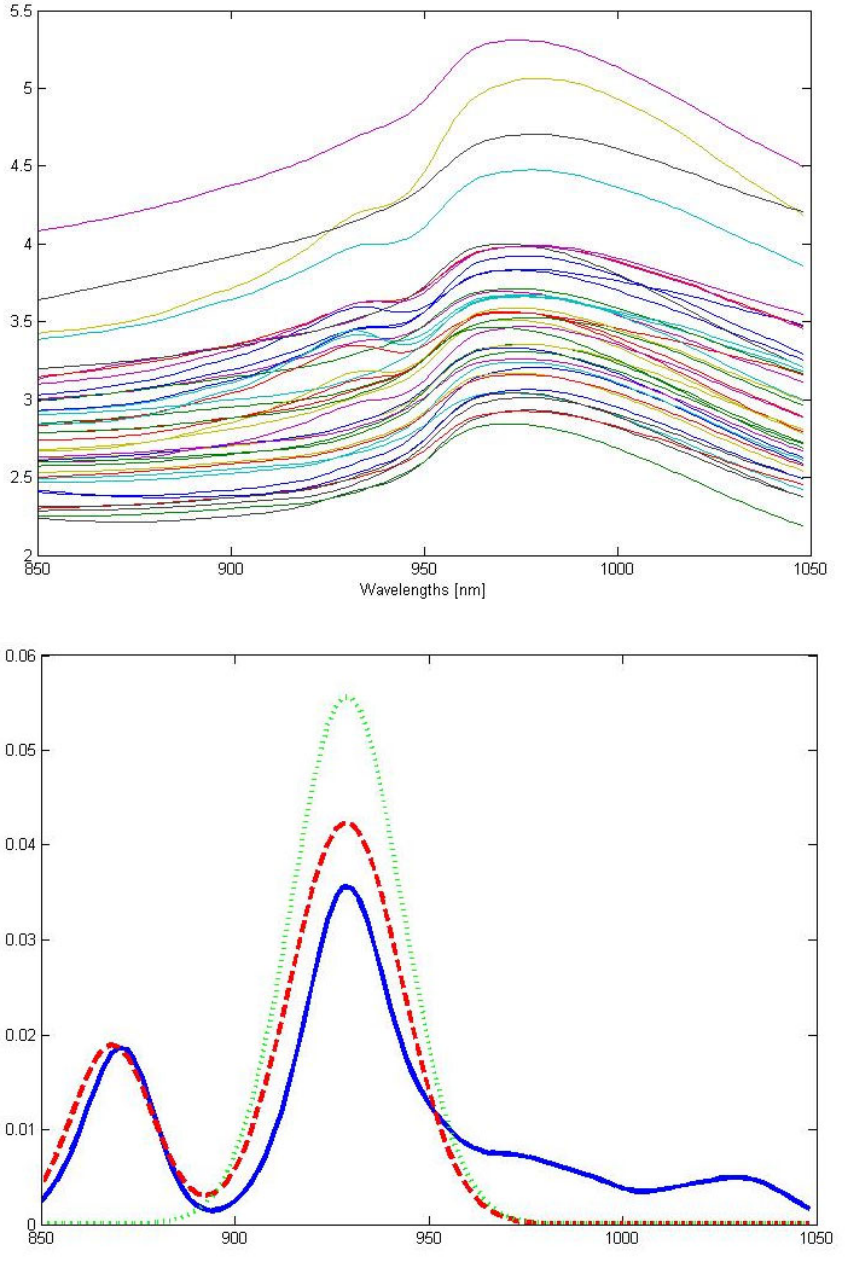


Fig. 2. top: Tecator data; bottom: Resulting relevance profiles: *solid* – at the end of usual GFRLVQ, *dashed* – at the maximum sparseness without significant accuracy loss, *dotted* – maximum sparseness (i.e. only one single remaining basisfunction).

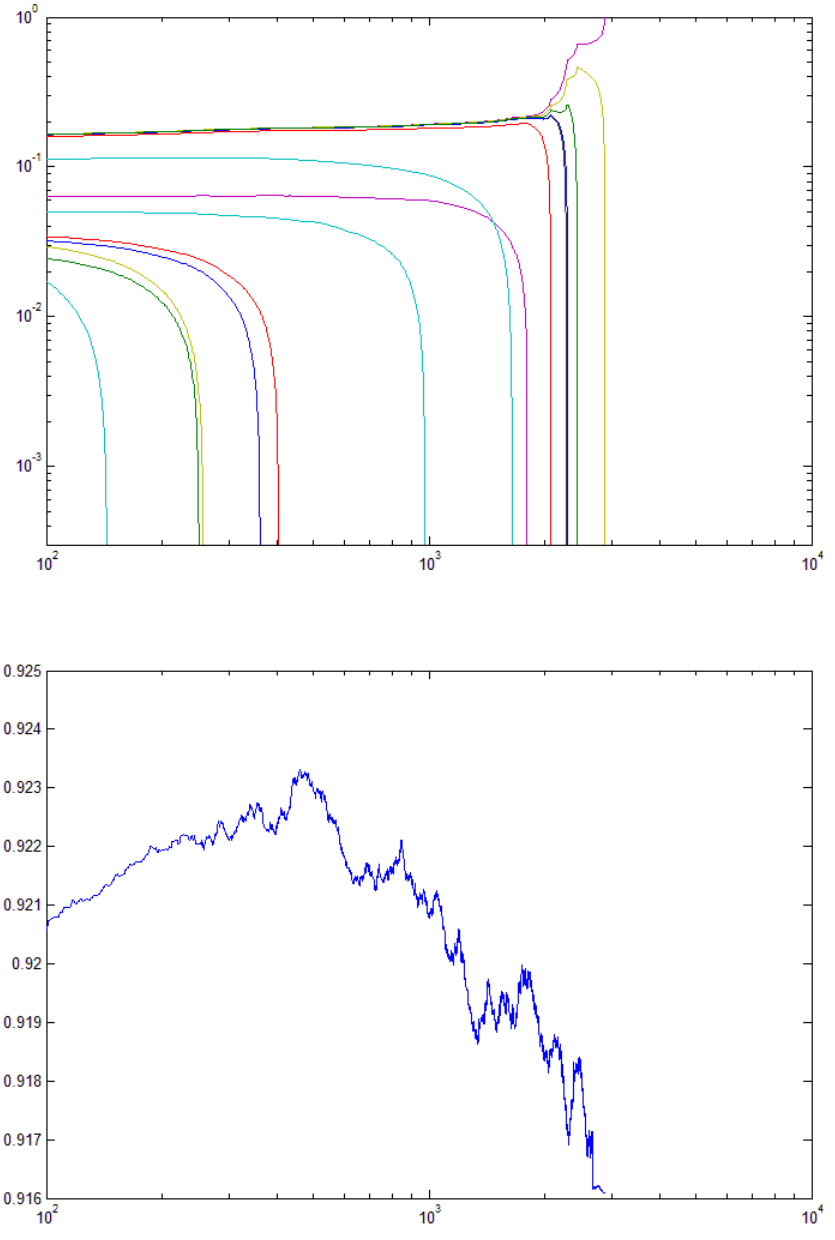


Fig. 3. The same as in Fig. 1 but now for the Wine data set. In this example 15 Lorentzians are taken as basis functions.

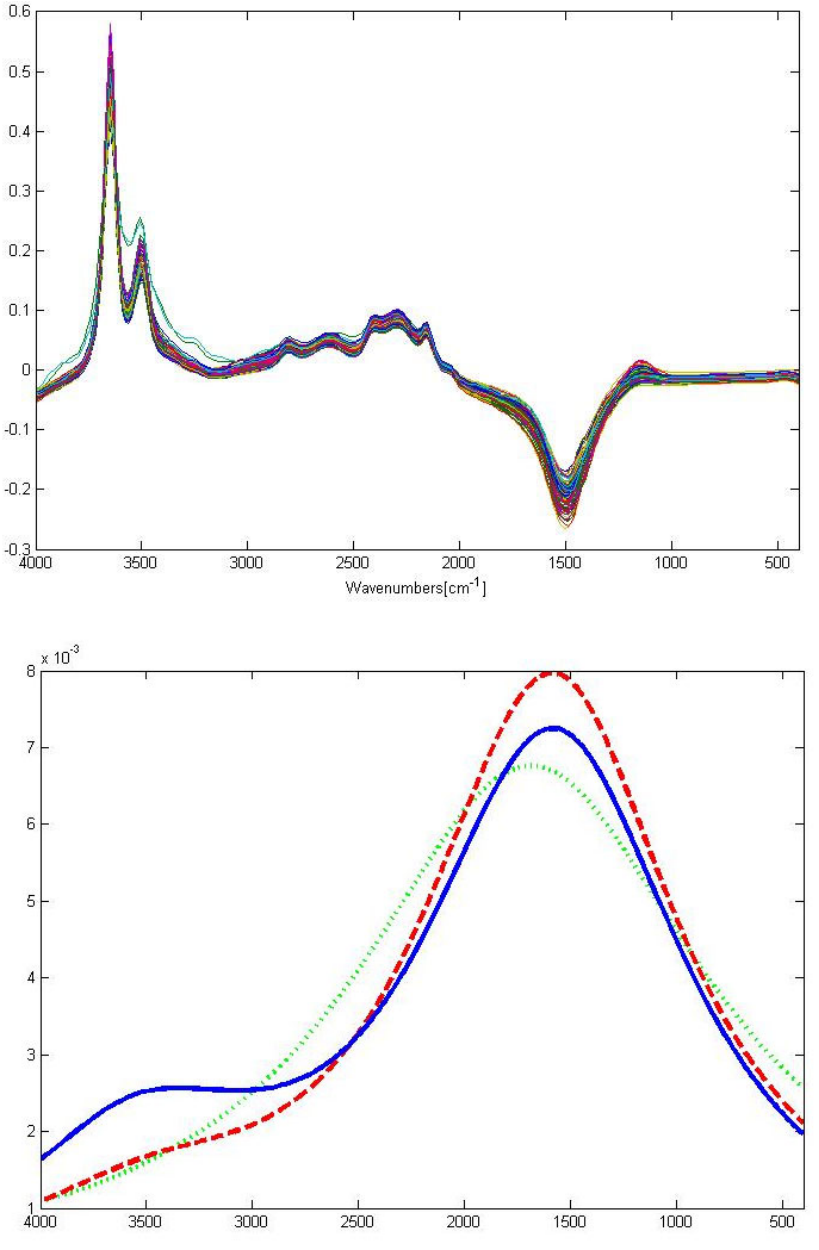


Fig. 4. top: Wine data; bottom: Resulting relevance profiles: *solid* – at the end of usual GFRLVQ, *dashed* – at the maximum sparseness without significant accuracy loss, *dotted* – maximum sparseness (i.e. only one single remaining basis function).

$\gamma(\tau) \cdot H(\boldsymbol{\beta})$ leads to a sparsity in the weighting $\boldsymbol{\beta}$ while a high accuracy is still achieved in the beginning of the optimization. If the sparsity becomes dominating, the accuracy significantly drops down, which is the case in this example for less than three non-vanishing basis functions. The resulting relevance profiles are displayed in in Fig.2. One can clearly observe the decrease in structural richness with increasing sparsity. The relevance profiles for the Tecator data are in nice agreement with the results found in [9]. In particular, the dominating relevance area between the bands 25 and 55, corresponding to wavelengthes 850nm–890nm, is also suggested to be important for classification in previous investigations [5].

For the Wine data set 15 Lorentzians were taken in the beginning GFRLVQ-learning phase. As one can observe in Fig.3, after a short stabilization phase the accuracy significantly decreases when 5 weighting coefficients are converged to zero with a further loss of accuracy the more weighting coefficients tend to become zero, too. The respective relevance profiles can be found Fig.4. For the Wine data set earlier findings are also justified. The influence of the data bands 130 – 230, corresponding to wavenumbers between 2000cm^{-1} and 1000cm^{-1} seems to be class distinguishing [4],[5]. Another, relevant range for differentiation is 3700cm^{-1}–3200cm^{-1}, which is weakly taken in shape only for the full set of 15 basis functions. For optimum sparseness this areas is indicated as not relevant and for the maximum sparseness completely ignored.

5 Conclusion

We propose the *Sparse GFRLVQ* for optimal model generation with respect to functional relevance learning. Functional relevance learning supposes that data are representations of functions such that the relevance profile can be assumed as a function, too. This allows the description in terms of a superposition of basis functions. Sparsity is judged in terms of the entropy of the respective weighting coefficients. The approach is demonstrated for two exemplary data sets with two different kinds of basis functions, Gaussians and Lorentzians whereas for the similarity measure the weighted Euclidean distance was used, for simplicity. Obviously, the Euclidean distance is not based on a functional norm. Yet, the transfer of the functional relevance approach to real functional norms and distances like Sobolev norms [15], Lee-norm [6,7], kernel based LVQ-approaches [14] or divergence based similarity measures [13], which carry the functional aspect inherently, is straight forward and topic of future investigations.

References

1. Hammer, B., Villmann, T.: Generalized relevance learning vector quantization. Neural Networks 15(8-9), 1059–1068 (2002)
2. Kato, T.: On the adiabatic theorem of quantum mechanics. Journal of the Physical Society of Japan 5(6), 435–439 (1950)
3. Kohonen, T.: Self-Organizing Maps. Springer Series in Information Sciences, vol. 30. Springer, Heidelberg (1995), 2nd extended edn. (1997)

4. Krier, C., Rossi, F., François, D., Verleysen, M.: A data-driven functional projection approach for the selection of feature ranges in spectra with ICA or cluster analysis. Chemometrics and Intelligent Laboratory Systems 91, 43–53 (2008)
5. Krier, C., Verleysen, M., Rossi, F., François, D.: Supervised variable clustering for classification of NIR spectra. In: Verleysen, M. (ed.) Proceedings of 16th European Symposium on Artificial Neural Networks (ESANN), Bruges, Belgique, pp. 263–268 (2009)
6. Lee, J., Verleysen, M.: Generalization of the l_p norm for time series and its application to self-organizing maps. In: Cottrell, M. (ed.) Proc. of Workshop on Self-Organizing Maps (WSOM) 2005, Paris, Sorbonne, pp. 733–740 (2005)
7. Lee, J., Verleysen, M.: Nonlinear Dimensionality Reduction. In: Information Sciences and Statistics. Springer Science+Business Media, New York (2007)
8. Meurens, M.: Wine data set, http://www.ucl.ac.be/mlg/index.php?page=databases.meurens.bnut.ucl.ac.be
9. Rossi, F., Lendasse, A., François, D., Wertz, V., Verleysen, M.: Mutual information for the selection of relevant variables in spectrometric nonlinear modelling. Chemometrics and Intelligent Laboratory Systems 80, 215–226 (2006)
10. Sato, A., Yamada, K.: Generalized learning vector quantization. In: Touretzky, D.S., Mozer, M.C., Hasselmo, M.E. (eds.) Proceedings of the 1995 Conference on Advances in Neural Information Processing Systems 8, pp. 423–429. MIT Press, Cambridge (1996)
11. Thodberg, H.: Tecator meat sample dataset, http://lib.stat.cmu.edu/datasets/tecator
12. Villmann, T., Cichocki, A., Principe, J.: Information theory related learning. In: Verleysen, M. (ed.) Proc. of European Symposium on Artificial Neural Networks (ESANN 2011), Evere, Belgium, d-side publications (2011) (page in press)
13. Villmann, T., Haase, S.: Divergence based vector quantization. Neural Computation 23(5), 1343–1392 (2011)
14. Villmann, T., Hammer, B.: Theoretical aspects of kernel GLVQ with differentiable kernel. IfI Technical Report Series (IfI-09-12), pp. 133–141 (2009)
15. Villmann, T., Schleif, F.-M.: Functional vector quantization by neural maps. In: Chanussot, J. (ed.) Proceedings of First Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS 2009), pp. 1–4. IEEE Press, Los Alamitos (2009); ISBN 978-1-4244-4948-4

Relevance Learning in Unsupervised Vector Quantization Based on Divergences

Marika Kästner[1], Andreas Backhaus[2], Tina Geweniger[1], Sven Haase[1], Udo Seiffert[2], and Thomas Villmann[1]

[1] Computational Intelligence Group,
University of Applied Sciences Mittweida, 09648 Mittweida, Germany
{kaestner,geweniger,haase,villmann}@hs-mittweida.de
[2] Biosystems Engineering Group, Fraunhofer Inst. f. Fabrikbetrieb
und -automatisierung IFF,39106 Magdeburg, Germany
{Andreas.Backhaus,Udo.Seiffert}@iff.fraunhofer.de

Abstract. We propose relevance learning for unsupervised online vector quantization algorithm based on stochastic gradient descent learning according to the given vector quantization cost function. We consider several widely used models including the neural gas algorithm, the Heskes variant of self-organizing maps and the fuzzy c-means. We apply the relevance learning scheme for divergence based similarity measures between prototypes and data vectors in the vector quantization schemes.

Keywords: vector quantization, relevance learning, divergence learning.

1 Introduction

Machine learning algorithms for unsupervised vector quantization (VQ) comprise a broad variety of models ranging from statistically motivated approaches to strong biologically realistic models. The main task is to describe given data in a faithful way such that the main properties of the data are preserved as most as possible by a set of few prototypes. These properties could be the probability density [28], the shape of data in sense of possibly non-linear principle component analysis (PCA), [23],[25], or visualization skills like in topology preserving mapping [30] or the usual reconstruction error. For the different goals several approaches exist [35], whereby we have further to differentiate according to the type of adaptation: Batch methods use all the data at the same time whereas online models adapt incrementally. Usually, the Euclidean distance is used in all these models. Yet, modern methods also include non-standard dissimilarity measures like functional norms [17], Sobolev norms [33], kernel based approaches [32], or divergences [31].

Relevance learning introduced for supervised learning vector quantization [10] and its generalization, the so-called matrix learning [26], were recently extended to unsupervised batch learning in topographic mapping [1]. Relevance or matrix learning weights and correlates the data dimensions to achieve better classification results in supervised learning and the signal to noise ratio in unsupervised

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 90–100, 2011.

models. For unsupervised VQ the method of relevance learning is strongly related to local PCA learning [18],[19],[21],[27].

For high-dimensional data standard techniques the curse of dimensionality causes further difficulties and more specifically designed approaches are required [4]. In case of spectral data or generalizations thereof, divergences as dissimilarity measure between data and prototypes maybe an appropriate alternative to standard dissimilarity measures to address the specificity of data and, thus, decrease the influence of the curse of dimensionality [31].

In this paper we propose divergence based unsupervised vector quantization in combination with relevance learning. For this purpose we demonstrate the idea with three different popular vector quantizers: the neural gas [20], the Heskes variant of the self-organizing map [11],[15] and, the fuzzy-c-means (FCM) [9]. As exemplary real world data examples we use spectral data measured by a NASA Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) for earth surface investigation and spectral data obtained from the normalized multi-scale bending energy of leaf shapes.

2 The Vector Quantization Models

In general, vector quantization comprises unsupervised methods for data compression of vectorial data $\mathbf{v} \in V \subseteq \mathbb{R}^n$ with probability density $\mathcal{P}$ by prototypes $\mathbf{w} \in W \subset \mathbb{R}^n$. The similarity between data vectors $\mathbf{v}$ and prototypes $\mathbf{w}$ is judged in terms of a dissimilarity measure $d(\mathbf{v}, \mathbf{w})$ frequently taken as the Euclidean distance. Yet, more advanced dissimilarity measures, not necessarily supposed to be a mathematical distance, can be applied. If the underlying cost function

$$E_{VQ} = \int \mathcal{P}(\mathbf{v}) \cdot L(\mathbf{v}, W, d(\mathbf{v}, \mathbf{w}))\, d\mathbf{v} \tag{1}$$

of the vector quantization algorithm is minimized by stochastic gradient learning with respect to $\mathbf{w}$, the used dissimilarity measure has to be assumed as a *differentiable* functional with respect to $\mathbf{w}$. $L(\mathbf{v}, W)$ are *local costs* differing for the several algorithms.

A robust alternative to the classical *c-means* algorithm is the *neural gas* (NG) [20] with local costs

$$L_{NG}(\mathbf{v}, W, d(\mathbf{v}, \mathbf{w})) = \sum_{i \in A} \frac{h_\sigma(i, \mathbf{v}, W)}{2C(\sigma, N)} d(\mathbf{v}, \mathbf{w}_i) \tag{2}$$

with the neighborhood function $h_\sigma(i, \mathbf{v}, W) = \exp\left(\frac{-k_i(\mathbf{v}, W)}{2\sigma^2}\right)$, whereby $k_i(\mathbf{v}, W)$ yields the number of prototypes $\mathbf{w}_j$ for which $d(\mathbf{v}, \mathbf{w}_j) \leq d(\mathbf{v}, \mathbf{w}_i)$ holds. $C(\sigma, N)$ is a normalization constant, and A is the index set of W. After training the vector quantization assignments $\Psi_s(\mathbf{v})$ are set to one iff $s(\mathbf{v}) = \arg\min_{i \in A} d(\mathbf{v}, \mathbf{w}_i)$ and zero elsewhere realizing a winner-take-all decision.

T. Heskes introduced a variant of the *self-organizing map* (SOM), originally proposed by T. Kohonen [15], such that the SOM-model is based on local costs

$$L_{SOM}\left(\mathbf{v}, W, d\left(\mathbf{v}, \mathbf{w}\right)\right) = \Psi_{\mathbf{s}(\mathbf{v})}\left(\mathbf{v}\right) \cdot e_{\mathbf{r}}\left(W, \mathbf{v}\right) \tag{3}$$

with local errors

$$e_{\mathbf{r}}\left(W, \mathbf{v}\right) = \sum_{\mathbf{r}' \in A} h_{\mathbf{r}\mathbf{r}'} d\left(\mathbf{v}, \mathbf{w}_{\mathbf{r}'}\right) \text{ and } \mathbf{s}\left(\mathbf{v}\right) = \arg\min_{\mathbf{r} \in A} e_{\mathbf{r}}\left(W, \mathbf{v}\right) \tag{4}$$

where $h_{\mathbf{r}\mathbf{r}'} = \exp\left(\frac{-d_A\left(\mathbf{r},\mathbf{r}'\right)}{2\sigma^2}\right)$ and $d_A\left(\mathbf{r}, \mathbf{r}'\right)$ denotes some dissimilarity measure in the index space A, which is equipped with a topological structure for SOMs. The assignments $\Psi_{\mathbf{r}}\left(\mathbf{v}\right)$ are one iff $\mathbf{r} = \mathbf{s}\left(\mathbf{v}\right)$ and zero elsewhere but here based on the local errors (4).

The standard *fuzzy c-means* (FCM) was originally developed by Dunn [9] and improved by Bezdek [3]. Its local costs are given as

$$L_{FCM}\left(\mathbf{v}, W, d\left(\mathbf{v}, \mathbf{w}\right)\right) = \frac{1}{2} \sum_{i \in A} \left(\Psi_i\left(\mathbf{v}\right)\right)^m d(\mathbf{v}, \mathbf{w}_i) \ , \tag{5}$$

where $\Psi_i\left(\mathbf{v}\right)$ are the fuzzy assignments, and the exponent m determines the fuzziness commonly set to $m > 1$. For $m \to 1$ the clustering becomes crisp. Frequently, the fuzziness is chosen as $m = 2$.

Minimizing each of these cost functions by stochastic gradient descent learning or batch mode algorithms realizes the respective vector quantization algorithm distributing the prototypes in the data space.

3 Divergences as Dissimilarity Measure for Spectral Data and Relevance Learning

So far we have not discussed any specific choice of the dissimilarity measure $d\left(\mathbf{v}, \mathbf{w}\right)$. Frequently the Euclidean distance is used. To improve the performance in SOM and NG, recently the distance measure

$$d_{\mathbf{\Lambda}}\left(\mathbf{v}, \mathbf{w}\right) = \left(\mathbf{v} - \mathbf{w}\right)^T \mathbf{\Lambda} \left(\mathbf{v} - \mathbf{w}\right) \tag{6}$$

has been suggested, with a positive definite matrix $\mathbf{\Lambda}$ to be adapted by gradient descent learning [1], and reducible to an Euclidean distance if $\mathbf{\Lambda}$ is decomposable into $\mathbf{\Lambda} = \mathbf{\Omega}^T \mathbf{\Omega}$ [5]. For a diagonal $\mathbf{\Lambda}$ the classical *relevance learning* is obtained.

Processing spectral data vectors using the Euclidean distance may not be appropriate. Specific dissimilarity measures for densities or positive functions as well as generalizations thereof, as spectral data are supposed to be, are *(generalized) divergences* [8],[22],[31]. The idea of relevance learning can also be applied to divergences weighting the spatial range [31]. Here we focus to the most prominent examples (generalized) *Kullback-Leibler-Divergence* $D_{GKL}\left(\mathbf{v}||\mathbf{w}\right)$, (generalized) *Rényi-divergence* $D_{\alpha}^{GR}\left(\mathbf{v}||\mathbf{w}\right)$, the γ-divergence $D_{\gamma}\left(\mathbf{v}||\mathbf{w}\right)$, and the η-divergence

D_η. These read for relevance weighted arguments $\tilde{\mathbf{v}} = \boldsymbol{\lambda} \circ \mathbf{v}$ and $\tilde{\mathbf{w}} = \boldsymbol{\lambda} \circ \mathbf{w}$ in the discrete case as

$$D_{GKL}\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}\right) = \sum_i \lambda_i v_i \log\left(\frac{v_i}{w_i}\right) - \left(\lambda_i v_i - \lambda_i w_i\right) \tag{7}$$

$$D_\alpha^{GR}\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}\right) = \frac{1}{\alpha - 1} \log\left(1 + \sum_i \left[\lambda_i v_i^\alpha w_i^{1-\alpha} - \alpha \cdot \lambda_i v_i + (\alpha - 1)\,\lambda_i w_i\right]\right) \tag{8}$$

$$D_\gamma\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}\right) = \log\left[\frac{\left(\sum_i \left(\lambda_i v_i\right)^{\gamma+1}\right)^{\frac{1}{\gamma(\gamma+1)}} \cdot \left(\sum_i \left(\lambda_i w\right)^{\gamma+1}\right)^{\frac{1}{\gamma+1}}}{\left(\sum_i \left(\lambda_i\right)^{\gamma+1} v_i w_i^\gamma\right)^{\frac{1}{\gamma}}}\right] \tag{9}$$

$$D_\eta\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}\right) = \sum_i \left(\lambda_i v_i\right)^\eta + (\eta - 1) \cdot \left(\lambda_i w_i\right)^\eta - \left(\lambda_i\right)^\eta \eta \cdot v_i \cdot w_i^{(\eta-1)} \tag{10}$$

and $\boldsymbol{\lambda} \circ \mathbf{v}$ denotes the Hadamard product between the relevance vector $\boldsymbol{\lambda}$ and $\mathbf{v}$ [31]. It should be noted that D_η becomes the quadratic Euclidean distance for $\eta = 2$. Further, both D_α^{GR} and D_γ converge to D_{GKL} in the limit $\alpha \to 0$ and $\gamma \to 0$, respectively. The divergence D_γ for $\gamma = 1$ is the Cauchy-Schwarz-divergence D_{CS} [24].

Relevance learning by stochastic gradient learning in the above mentioned vector quantization algorithms can be performed by taking for a presented data vector $\mathbf{v}$ during learning the respective derivatives

$$\triangle\lambda_t \sim -\frac{\partial L\left(\mathbf{v}, W, D\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}\right)\right)}{\partial \lambda_t} = \frac{\partial L\left(\mathbf{v}, W, D\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}\right)\right)}{\partial D\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}\right)} \frac{\partial D\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}\right)}{\partial \lambda_t}$$

for a considered divergence D [31].

In case of high-dimensional spectra the number of relevance weights λ_i is huge and may cause instable behavior. To prevent these instabilities, *functional relevance learning* was recently proposed. In this approach the relevance profile vector $\boldsymbol{\lambda}$ is replaced by a function $\lambda(t)$ assumed as a superposition $\lambda(t) = \sum_{l=1}^K \beta_l \mathcal{K}_l(t)$ of simple basis functions $\mathcal{K}_l$ depending on only a few parameters with the restrictions $\beta_i \geq 0$ and $\sum_{l=1}^K \beta_l = 1$. Well-known examples are Gaussians or Lorentzians:

$$\mathcal{K}_l(t) = \frac{1}{\sigma_l\sqrt{2\pi}} \exp\left(-\frac{(t - \Theta_l)^2}{2\sigma_l^2}\right) \text{ and } \mathcal{K}_l(t) = \frac{1}{\eta_l \pi} \frac{\eta_l^2}{\eta_l^2 + (t - \Theta_l)^2} \tag{11}$$

Now, relevance learning takes place by adaptation of the parameters β_l, Θ_l,σ_l and η_l, respectively [29].

4 Experiments

4.1 Data

Leave shape data: The leave form dataset originates from a study of nine genotypes and their shape characteristics of *Arabidopsis thaliana* (L.) Heynh

published in [2]. It is well known that gene networks as well as environmental cues control leaf shapes in a highly orchestrated manner. In order to clarify the role of genes involved in leaf development, precise description, quantification and categorization of leaf phenotypes e.g. leaf shapes is vital. Two wild type and seven mutant lines were used: *col-0*, *ws-2*, *angustifolia (an)*, *elongata1 (elo1-1)*, *gpa1-1* and *gpa1-2* variants, *jagged1 (jag1)*, *rotendufolia3 (rot-3)* and *serrate (se)*. Each condition held between 14 and 24 samples. Leaves were fixed at 28 days after sowing and carefully flattened while keeping the leaf margin intact. Images of flattened leaves were taken using a CCD camera mounted on a stereomicroscope at 13,919 DPI and stored as 8-bit grey value bitmap. Fig. 1a shows two examples per line as polygons created by the shape quantification procedure described below. Leaves were segmented from the background by a grey value threshold and a Canny edge detector was used to create the boundary trace. A point distribution model (PDSM) was then fitted by an Active Contour [13]. Then a plane curve was created through spline interpolation and 500 points in total sampled by arc length parametrization. The shape was aligned with its longest extension along the y-axis and the base was cut where the petiole has increased by 40% from its average width. As shape description we used the normalized multi-scale bending energy (NMBE) [7]. A contour is convoluted with a Gaussian function of increasing sigma thereby smoothing the curve continuously leading to contours of lower details (see Fig. 1b). At each scale, the normalized shape curvature is calculated, squared and integrated along the curve leading to the bending energy per scale, additionally a log is applied. As shape descriptor

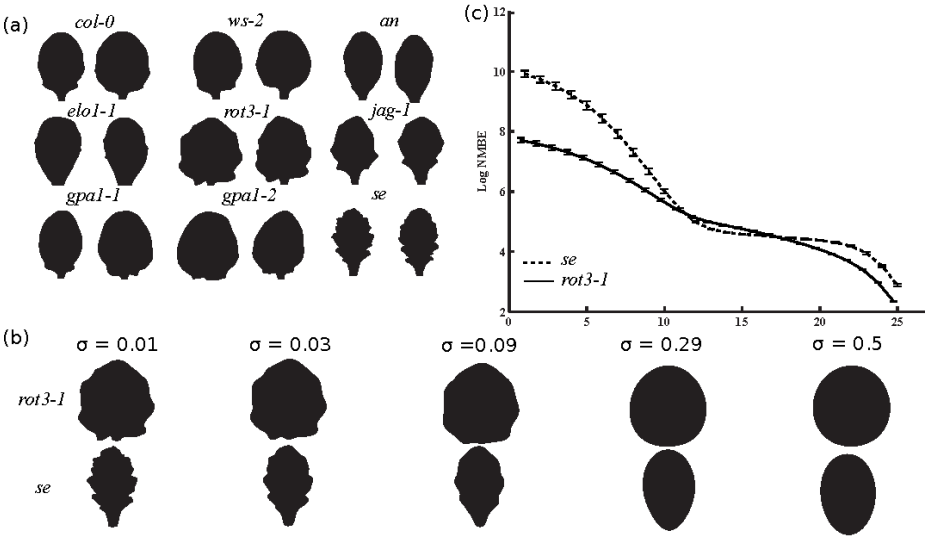


Fig. 1. (a) Two wild types (*col-0*, *ws-2)* and seven mutations of Arabidopsis have been used. (b) For the curvegram, a contour is continuously smoothed with Gaussians of increasing sigma; (c) the log-normalized multi-scale bending energy as the shape descriptor for different genotypes, depicted are mean and standard deviation per scale.

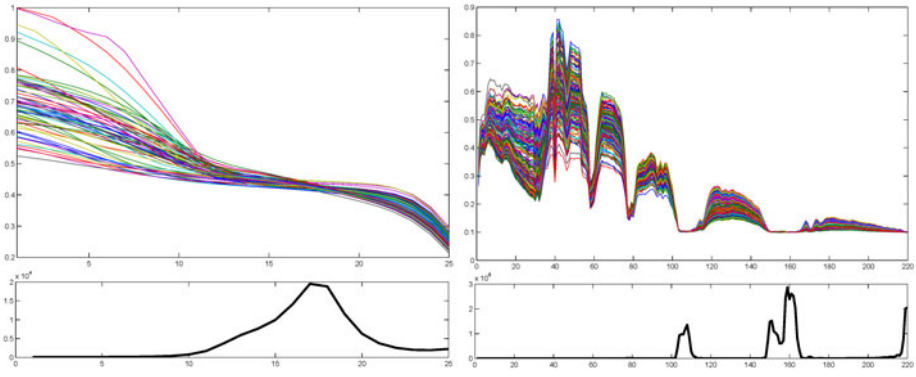


Fig. 2. Visualization of data (top) and the inverse variance (bottom) of the frequencies for leaf shape data (left) and AVIRIS data (right)

the bending energy has a number of advantages: it is invariant to rotation, translation, and through the normalization with the perimeter also invariant to shape scaling. The bending energy results from elasticity theory and provides a means for expressing the amount of energy that is needed to transform the contour of a specific shape into its minimum-energy state, namely, a circle with the same perimeter as the original object. For shapes related to real objects, such as biological shapes (e.g. membranes, neurons, organs), the bending energy provides a particularly meaningful physical interpretation in terms of the energy that has to be applied in order to produce or modify specific objects [36,7].The bending energy profile or curvegram [7] serves as input vector to the vector quantization methods. In Fig.1c the mean profiles for two exemplary genotype are depicted. For this paper, 25 different scales with a minimum scale of 0.01 and maximum scale of 0.5 are used.

Remote sensing data: The remote sensing data set is the publicly available Indian Pines data from NASA Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) consisting of 145×145 pixels [16]. AVIRIS acquires data in 220 bands of 10nm width from $400 - 2500$nm. For this data set 20 noisy bands can be identified $(104 - 108, 150 - 163, 220)$ due to atmospheric water absorption and, therefore, safely be removed [6]. We did not use the knowledge about the 16 identified classes because of unsupervised learning. The data sets are visualized in Fig.2 .

4.2 Validity Measures

Although we are not focussing on clustering rather than representation by vector quantization we compare the solutions for the different algorithms and divergences by cluster validity measures to show the improvement obtained by relevance learning. The used measures should not make explicit or implicit use of the properties of the Euclidean distance. Further, we applied two different measures

reflecting the fact that clustering is an ill-posed problem and, therefore, different measures emphasize different aspects. Yet, they all have in common that they judge compactness and separability. We selected two measures, which are applicable to fuzzy (FCM) as well as to crisp (NG, SOM) vector quantization, the modified $SVF-$index [37]:

$$SVF\left(V,W\right)=\frac{\mathcal{S}}{\mathcal{C}}=\frac{\sum_{i\in A}avg_D\left(\tilde{\mathbf{w}}_i\right)}{\sum_{i\in A}\max_{\mathbf{v}\in V}\left(\left(\Psi_i\left(\mathbf{v}\right)\right)^m\cdot D\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}_i\right)\right)} \tag{12}$$

where $avg_D\left(\tilde{\mathbf{w}}_i\right)=\left\langle D\left(\tilde{\mathbf{w}}_j||\tilde{\mathbf{w}}_i\right)+D\left(\tilde{\mathbf{w}}_i||\tilde{\mathbf{w}}_j\right)\right\rangle_{j\in A}$ denotes the mean distance value of other prototypes to $\tilde{\mathbf{w}}_i$. The value $\mathcal{S}$ estimates the separation whereas $\mathcal{C}$ appraises the compactness. The second measure is the modified fuzzy index introduced by Xie & Beni [34]:

$$XiB=\frac{\sum_{i\in A}\sum_{\mathbf{v}\in V}\left(\Psi_i\left(\mathbf{v}\right)\right)^m\cdot D\left(\tilde{\mathbf{v}}||\tilde{\mathbf{w}}_i\right)}{\frac{1}{\#A}\sum_{i\in A}\left(avg_D\left(\tilde{\mathbf{w}}_i\right)\right)^2}\quad . \tag{13}$$

The higher the SVF- and the lower the XiB-value are for a chosen divergence D, the better results are achieved.

Table 1. SVF- and XiB-Index for leaf shape (top) and the AVIRIS-dataset (bottom) with the different methods

		SVF			XiB		
		NG	SOM	FCM	NG	SOM	FCM
Euclid	no rel.	0.538	0.090	9.442	1.575	1.933	0.123
	with rel.	0.688	0.219	10.424	1.212	1.266	0.119
	func. rel.		0.235			1.057	
Cauchy-	no rel.	1.472	0.330	9.108	0.908	1.274	0.162
Schwarz	with rel.	2.011	0.350	9.870	0.608	1.102	0.147
div.	func. rel.		0.312			1.224	
Kullback-	no rel.	1.499	0.238	11.537	0.460	1.545	0.125
Leibler	with rel.	1.856	0.255	11.270	0.608	1.497	0.140
div.	func. rel.		0.339			1.082	
Rényi	no rel.	1.760	0.235	8.997	0.467	1.661	0.149
div.	with rel.	2.384	0.245	10.116	0.415	1.514	0.149
($\alpha=2$)	func. rel.		0.348			1.112	
Euclid	no rel.	0.152	0.029	6.624	0.727	1.549	0.170
	with rel.	0.176	0.049	1.407	0.504	0.855	0.170
	func. rel.		0.082			0.670	
Cauchy-	no rel.	0.138	0.047	0.3091	0.330	1.952	0.1691
Schwarz	with rel.	0.125	0.135	0.3496	0.240	1.811	0.2738
div.	func. rel.		0.136			0.986	
Kullback-	no rel.	0.258	0.034	3.791	0.320	1.800	0.169
Leibler	with rel.	0.336	0.036	3.809	0.240	1.633	0.168
div.	func. rel.		0.036			1.622	
Rényi	no rel.	0.258	0.046	4.052	0.266	1.042	0.168
div.	with rel.	0.349	0.038	4.005	0.205	1.803	0.170
($\alpha=2$)	func. rel.		0.036			1.914	

4.3 Experimental Results

We applied all three algorithms NG, SOM, and FCM with $m = 2$ to both data sets with and without relevance learning using 20 prototypes for each algorithm. We used the quadratic Rényi-divergence D_2^{GR}, the η-divergence D_η for $\eta = 2$ (Euclidean distance), the Cauchy-Schwarz divergence D_{CS} and the Kullback-Leibler-divergence D_{GKL}. Additionally, we performed functional relevance learning for the SOM scheme. All settings are trained as usual in online vector quantization, i.e. by decreasing learning rate until convergence, and adiabatic relevance adaptation [14]. The results are depicted in Tab.1, whereby value variances (10-fold cross validation) are at least two relative magnitudes lower.

We observe that for low-dimensional leaf shape data NG and SOM show consistently an improvement by relevance learning. Further, according to the better XiB- and SVF-values, NG is superior to SOM in general as expected according to well-known earlier findings [20]. FCM also profits from relevance learning

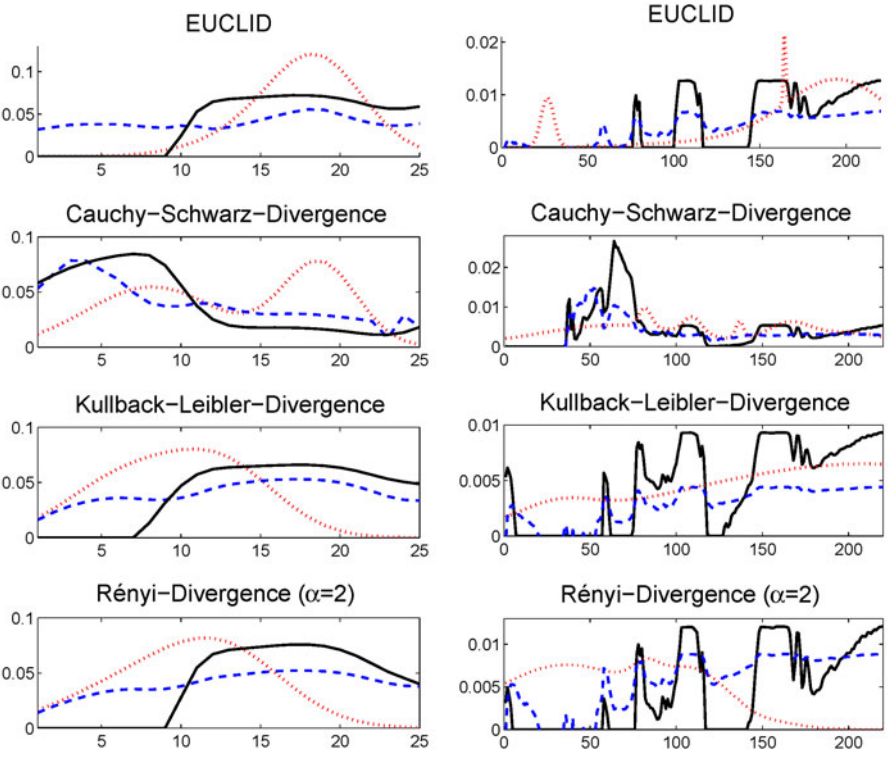


Fig. 3. Relevance profiles obtained according to the different divergences and algorithms for both data sets leaf shape (left) and AVIRIS data (right): NG - solid, SOM - dotted (functional relevance), FCM - dashed

except for the Kullback-Leibler-divergence D_{GKL}.[1] This inconsistent behavior is strengthened in case of the high-dimensional AVIRIS spectra yielding weak results for all dissimilarity measures. This could be related to the instable behavior of FCM [12]. Yet, NG and SOM generally benefit both from relevance learning also for the high-dimensional AVIRIS data. Again, the NG has an advantage in performance as awaited. Another point to be mentioned is that functional relevance learning for SOM yields a further improvement such that one can expect a similar behavior for the other algorithms too, but this is left for future investigations. On the other hand, all algorithms generate similar relevance profiles, i.e. indicating similar spectral ranges as important for high vector quantization performance, see Fig.3. Moreover, these relevance profiles look similar to the inverse variance profile of the data sets, see Fig.2. This behavior is in agreement with the theoretical results published in [1] and [21].

5 Conclusions

In this article we propose relevance learning for unsupervised online vector quantization using weighted divergences as dissimilarity measure. We demonstrate that generally an improvement of the performance can be expected. However, the behavior seems to be sensitive (at least) to the algorithm of choice. Further, difficulties may arise in case of high-dimensional data because in that case the number of parameters to be optimized (relevance weights) becomes huge. In the light of the SOM results, here *functional relevance learning* could offer an alternative [29].

References

1. Arnonkijpanich, B., Hasenfuss, A., Hammer, B.: Local matrix adaptation in topographic neural maps. Neurocomputing 74(4), 522–539 (2011)
2. Backhaus, A., Kuwabara, A., Bauch, M., Monk, N., Sanguinetti, G., Fleming, A.: LEAFPROCESSOR: a new leaf phenotyping tool using contour bending energy and shape cluster analysis. New Phytologist 187(1), 251–261 (2010)
3. Bezdek, J.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York (1981)
4. Bouveyron, C., Girard, S., Schmid, C.: High-dimensional data clustering. Computational Statistics and Data Analysis 57(1), 502–519 (2007)
5. Bunte, K., Hammer, B., Wismüller, A., Biehl, M.: Adaptive local dissimilarity measures for discriminative dimension reduction of labeled data. Neurocomputing 73, 1074–1092 (2010)
6. Camps-Valls, G., Bruzzone, L.: Kernel-based methods for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing 43(6), 1351–1362 (2005)

[1] A direct comparison of the values is not admissible because of the quadratic fuzzy valued assignments $(\Psi_i(\mathbf{v}))^2$.

7. Cesar, R.M., da Fontoura Costa, L.: Application and assessment of multiscale bending energy for morphometric characterization of neural cells. Rev. Sci. Instrum. 68(5), 2177–2186 (1997)
8. Cichocki, A., Amari, S.-I.: Families of alpha- beta- and gamma- divergences: Flexible and robust measures of similarities. Entropy 12, 1532–1568 (2010)
9. Dunn, J.: A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. Journal of Cybernetics 3, 32–57 (1973)
10. Hammer, B., Villmann, T.: Generalized relevance learning vector quantization. Neural Networks 15(8-9), 1059–1068 (2002)
11. Heskes, T.: Energy functions for self-organizing maps. In: Oja, E., Kaski, S. (eds.) Kohonen Maps, pp. 303–316. Elsevier, Amsterdam (1999)
12. Izakian, H., Abraham, A.: Fuzzy c-means and fuzzy swarm for fuzzy clustering problem. Expert Systems with Applications 38(3), 1835–1838 (2011)
13. Kass, M., Witkin, A., Terzopoulos, D.: Snakes - Active Contour models. International Journal of Computer Vision 1(4), 321–331 (1987)
14. Kato, T.: On the adiabatic theorem of quantum mechanics. Journal of the Physical Society of Japan 5(6), 435–439 (1950)
15. Kohonen, T.: Self-Organizing Maps. Springer Series in Information Sciences, vol. 30. Springer, Heidelberg (1995), 2nd extended edn. (1997)
16. Landgrebe, D.: Signal Theory Methods in Multispectral Remote Sensing. Wiley, Hoboken, New Jersey (2003)
17. Lee, J., Verleysen, M.: Generalization of the lp norm for time series and its application to self-organizing maps. In: Cottrell, M. (ed.) Proc. of Workshop on Self-Organizing Maps (WSOM) 2005, Paris, Sorbonne, pp. 733–740 (2005)
18. Liu, Z.-Y., Xu, L.: Topological local principal component analysis. Neurocomputing 55(3-4), 739–745 (2003)
19. López-Rubio, E., noz Pérez, J.M., Gómez-Ruiz, J.: A principal components analysis self-organizing map. Neural Networks 2(2), 261–270 (2004)
20. 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)
21. Möller, R., Hoffmann, H.: An extension of neural gas to local PCA. Neurocomputing 62, 305–326 (2004)
22. 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)
23. Oja, E., Lampinen, J.: Unsupervised learning for feature extraction. In: Zurada, J.M., Marks II, R.J., Robinson, C.J. (eds.) Computational Intelligence Imitating Life, pp. 13–22. IEEE Press, Los Alamitos (1994)
24. Principe, J.C., III, J.F., Xu, D.: Information theoretic learning. In: Haykin, S. (ed.) Unsupervised Adaptive Filtering, Wiley, New York (2000)
25. Rubner, J., Tavan, P.: A self-organizing network for principle-component analysis. Europhys. Letters 7(10), 693–698 (1989)
26. Schneider, P., Hammer, B., Biehl, M.: Adaptive relevance matrices in learning vector quantization. Neural Computation 21, 3532–3561 (2009)
27. Tipping, M., Bishop, C.: Mixtures of probabilistic principal component analyzers. Neural Computation 11, 443–482 (1999)
28. van Hulle, M.M.: Faithful representations with topographic maps. Neural Networks 12(6), 803–823 (1999)

29. Villmann, T., Cichocki, A., Principe, J.: Information theory related learning. In: Verleysen, M. (ed.) Proc. of European Symposium on Artificial Neural Networks (ESANN 2011), Evere, Belgium, d-side publications (2011) (page in press)
30. Villmann, T., Der, R., Herrmann, M., Martinetz, T.: Topology Preservation in Self-Organizing Feature Maps: Exact Definition and Measurement. IEEE Transactions on Neural Networks 8(2), 256–266 (1997)
31. Villmann, T., Haase, S.: Divergence based vector quantization. Neural Computation 23(5), 1343–1392 (2011)
32. Villmann, T., Hammer, B.: Theoretical aspects of kernel GLVQ with differentiable kernel. IfI Technical Report Series (IfI-09-12), pp. 133–141 (2009)
33. Villmann, T., Schleif, F.-M.: Functional vector quantization by neural maps. In: Chanussot, J. (ed.) Proceedings of First Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS 2009), pp. 1–4. IEEE Press, Los Alamitos (2009); ISBN 978-1-4244-4948-4
34. Xie, X., Beni, G.: A validity measure for fuzzy clustering. IEEE Transactions on Pat 13(8), 841–847 (1991)
35. Xu, R., Wunsch, D.: Survey of clustering algorithms. IEEE Transactions on Neural Networks 16(3), 645–678 (2005)
36. Young, I.T., Walker, J.E., Bowie, J.E.: An analysis technique for biological shape. i. Information and Control 25(4), 357–370 (1974)
37. Zalik, K., Zalik, B.: Validity index for clusters of different sizes and densities. Pattern Recognition Letters 32, 221–234 (2011)

Requirements for the Learning of Multiple Dynamics

Takashi Ohkubo, Tetsuo Furukawa, and Kazuhiro Tokunaga

Kyushu Institute of Technology, Kitakyushu 808-0196, Japan
furukawa@brain.kyutech.ac.jp
http://www.brain.kyutech.ac.jp/~furukawa

Abstract. The aim of this work is to discover the principles of learning a group of dynamical systems. The learning mechanism, which is referred to as the Learning Algorithm of Multiple Dynamics (LAMD), is expected to satisfy the following four requirements. (i) Given a set of time-series sequences for training, estimate the dynamics and their latent variables. (ii) Order the dynamical systems according to the similarities between them. (iii) Interpolate intermediate dynamics from the given dynamics. (iv) After training, the LAMD should be able to identify or classify new sequences. For this purpose several algorithms have been proposed, such as the Recurrent Neural Network with Parametric Bias and the modular network SOM with recurrent network modules. In this paper, it is shown that these types of algorithms do not satisfy the above requirements, but can be improved by normalization of estimated latent variables. This confirms that the estimation process of latent variables plays an important role in the LAMD. Finally, we show that a fully latent space model is required to satisfy the requirements, for which purpose a SOM with a higher-rank, such as a SOM^2, is best suited.

1 Introduction

The focus of this work is to develop learning algorithms that enable us to deal with a group of dynamical systems. More precisely, the aim of this work is to discover common principles shared by Learning Algorithms of Multiple Dynamics (LAMD), which have the ability to learn, represent, estimate, quantize, interpolate, and classify a class of dynamics. The LAMD is a useful tool when the system dynamics changes depending on the context or the environment. In particular, autonomous intelligent robots need an LAMD with excellent performance, because such robots are required to act in various environments, and to deal with various objects in a variety of ways.

For this purpose, modular network architectures have often been adopted. One of the representative architectures is MOdular Selection And Identification for Control (MOSAIC) proposed by Wolpert & Kawato [1]. Sometimes a SOM is combined with the modular network. Minamino et al. applied a SOM with recurrent neural network modules (RNN-SOM) to the humanoid QRIO, providing QRIO with the capability of multi-schema learning [2]. Minatohara & Furukawa developed the Self-Organizing Adaptive Controller (SOAC) [3,4] based on the mnSOM [5]. SOMAR also belong to this type [6]. In some cases, Jordan's plan unit [7] has been adopted as an alternative to the modular network architecture. Tani developed the Parametric Bias (PB) method based on Jordan's plan unit and applied it to multi-schema acquisition tasks for robots [8].

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 101–110, 2011.

Despite the fact that these algorithms seem to have yielded some success, little attention has been paid to the problems often encountered by these algorithms, such as learning instability, calculation time, confusion, and the interference of memories. The reason these problems have not been exposed is that robotic tasks are too complicated for theoretical analysis, while toy problems are often too simple to elucidate them. Ohkubo & Furukawa showed that these problems occur regularly even in ordinary learning tasks, and naive algorithms are only effective in limited cases, such as toy problems [9,10]. To make matters worse, the risk of such problems seems to increase when the system possesses latent variables, e.g., internal state variables. These are common in the case of dynamical systems.

In this paper we report that conventional naive algorithms do not work adequately in learning tasks of multiple dynamics. We also point out the importance of latent variable estimation, which should be carried out carefully so that the latent variables are consistent for all dynamics.

2 Requirements of the Learning Algorithm of Multiple Dynamics

First, let us clarify what we expect from the Learning Algorithm of Multiple Dynamics (LAMD). Suppose that we have a set of sequences, i.e., time-series data, $X = \{x_1(t), \ldots, x_N(t)\}$. Then, we expect the LAMD to learn these N sequences and represent their dynamics. In other words, we expect the LAMD to estimate the dynamics (e.g., input-output relations or differential/difference equations) and reproduce the given N sequences without the noise component. The simplest way to achieve this is to prepare N independent learning architectures of single dynamics, and to train them separately. In this situation, we do not encounter any problems of instability, interference, confusion, and so on. However, we would hesitate to call such an architecture the LAMD, of course, because we cannot do anything more than what can be done with a learning algorithm of single dynamics. Therefore, we implicitly expect the LAMD to have some additional abilities associated with multi-dynamics, such as interpolation, quantization, identification, etc. Consequently the LAMD is required to have a measure of difference or similarity between two dynamics or time-series.

Now let us consider a typical case dealt with by the LAMD shown in Fig. 1. In this case the dynamics is assumed to be represented by a difference equation

$$\mathbf{z}(t+1) = f(\mathbf{z}(t); \theta), \tag{1}$$

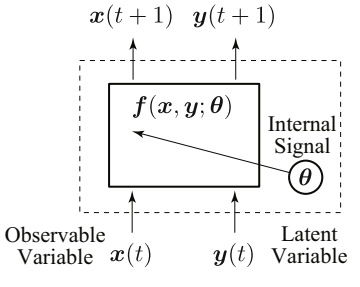


Fig. 1. Time-series generation model in a framework for learning of multiple dynamics

which is modified by the internal signal θ. Here, $\mathbf{z}(t)$ is the state variable consisting of an observable part $\mathbf{x}(t)$ and latent (unobservable) part $\mathbf{y}(t)$, i.e., $\mathbf{z}(t) = (\mathbf{x}(t), \mathbf{y}(t))$. (In this paper, the sequence of the latent variable $\mathbf{y}(t)$ is called a 'latent sequence'). It is also assumed that the internal signal θ alters the dynamics continuously, so that similar values of θ produce similar dynamics. Using this model, a set of training sequences can be obtained in the following way. First, a set of internal signals $\Theta = \{\theta_1, \ldots, \theta_N\}$ is randomly generated (the prior distribution $p(\theta)$ can be defined, if needed), and $f_n(\mathbf{z}) \triangleq f(\mathbf{z}; \theta_n)$ is obtained for each θ_n. Then, a time-series is generated for each θ_n using the difference equation $\mathbf{z}_n(t+1) = f_n(\mathbf{z}_n(t))$. (External noise $\boldsymbol{\varepsilon}(t)$ can be added to $\mathbf{z}(t)$ if necessary). Finally a set of sequences $X = \{\mathbf{x}_n(t)\}$ are observed, while $\{\theta_n\}$, $\{\mathbf{y}_n(t)\}$, and the function set $\{f_n(\mathbf{z})\}$ are all hidden from the LAMD.

With this framework, the LAMD is expected to satisfy the following requirements.

Requirement 1: The LAMD learns the dynamics of the training sequences $X = \{\mathbf{x}_n(t)\}$ and thus needs to estimate the function set $\{f_n(\mathbf{z})\}$ and the set of latent sequences $\{\mathbf{y}_n(t)\}$.

Requirement 2: The LAMD measures the distances or similarities between the given sequences, and orders them. Thus, the LAMD needs to estimate the internal signal $\{\theta_n\}$ and to sort the sequences by θ_n.

Requirement 3: By giving an intermediate θ, the LAMD represents the intermediate dynamics between given sequences.

Requirement 4: If a new sequence $\mathbf{x}_{\text{new}}(t)$ is given after training has finished, the LAMD classifies it, and allocates it to the appropriate position between the training sequences. Thus, the LAMD needs to identify the internal signal θ_{new} of the new sequence.

We define those algorithms satisfying the above four requirements as the LAMD in this paper. Note that the LAMD refers to a theoretical concept of multi-dynamics learning rather than a particular algorithm. There can be various implementations of the LAMD, and the aim of this paper is to discover the common principles of the LAMD.

3 Conventional Methods

3.1 Representative Architectures

To implement the LAMD using a neural network, there are two major approaches (Fig. 2). One is a modular network structure such as SOM with RNN modules or with autoregressive (AR) modules. The modular network SOM with RNNs (RNN-mnSOM) shown in Fig. 2(a) is representative of this approach [11]. In the RNN-mnSOM, every nodal unit of the SOM is replaced by an RNN module, and the location of the winner (i.e., the best matching module of the given sequence) in the feature map space represents the control signal θ. This approach is sometimes referred to as the *local representation*, because each dynamics is represented separately in an RNN module.

The other approach is to use Jordan's plan unit [7]. A representative architecture is the recurrent network with parametric bias (RNNPB), shown in Fig. 2(b) [8]. RNNPB is a variation of the multi-layer perceptron (MLP) with recurrent connections, and the

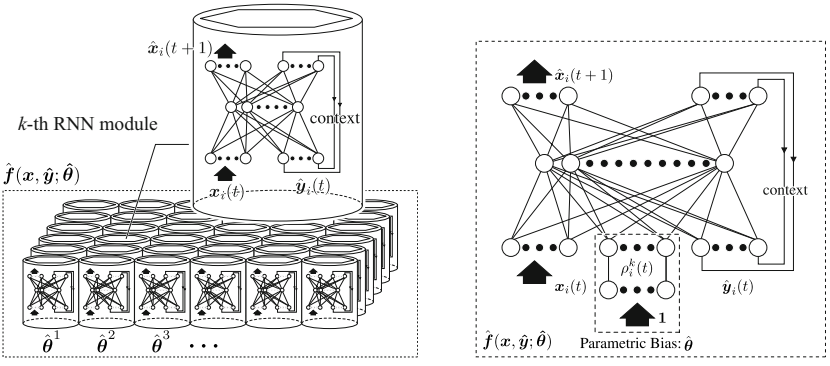


Fig. 2. Conventional learning algorithms of multiple dynamics. (a) RNN-mnSOM. (b) RNNPB.

input-output relation is modified by additional bias units called *parametric bias*. The parametric bias vector (i.e., the set of parametric bias values) represents the internal signal θ, which is determined by a back-propagation algorithm. This approach is referred to as the *distributed representation*, because the representation of the dynamics is distributed across the network.

3.2 Naive Algorithm

Though the appearance of these two architectures is quite different, in essence the two approaches have much in common.

1. For each training sequence, the least error node (e.g. RNN-module), or the least error parametric bias vector becomes the winner of the sequence.
2. The winner and its neighbors are trained to represent the dynamics of the training sequence. In the RNN-mnSOM, the neighbors are determined explicitly by the neighborhood function, whereas in the RNNPB they are determined implicitly by the sigmoid curve of the neurons.
3. As a result, the network is trained by a set of sequences, which are mixed with the weights determined by the neighborhood. That is, for a particular θ (i.e., each RNN-module or each parametric bias vector), the network is trained so as to minimize the weighted sum of the square error of multiple training sequences.

In this paper, algorithms using this strategy are called *naive algorithms*. In the case of the RNN-mnSOM, this naive algorithm is formulated as follows. Let $\hat{\mathbf{x}}_n^k(t)$ be the output of the k-th RNN module. Then, the square error between the given sequence $\mathbf{x}_n(t)$ and the output of the k-th module $\hat{\mathbf{x}}_n^k(t)$ is given by

$$E_n^k = \frac{1}{2} \sum_{t=1}^{T} \|x_n(t) - \hat{x}_n^k(t)\|^2. \tag{2}$$

The winner module and the estimated internal signal are expressed as

$$k_n^* = \arg\min_k E_n^k \tag{3}$$

$$\hat{\theta}_n = \theta^{k_n^*}. \tag{4}$$

Here, θ^k denotes the coordinates of the k-th module in the feature map space. Using the neighborhood function, the learning coefficients are determined for every module.

$$a_n^k = \frac{\exp\left[-\|\theta^k - \hat{\theta}_n\|^2/2\sigma^2\right]}{\sum_{n'} \exp\left[-\|\theta^k - \hat{\theta}_{n'}\|^2/2\sigma^2\right]} \tag{5}$$

Finally, the connection weight $\mathbf{w}^k$ of each RNN module is updated by a back-propagation algorithm (more precisely, the back-propagation through time algorithm) as follows.

$$E^k = \sum_{n=1}^{N} a_n^k E_n^k \tag{6}$$

$$\Delta\mathbf{w}^k = -\eta \frac{\partial E^k}{\partial \mathbf{w}^k} = -\eta \sum_{n=1}^{N} a_n^k \frac{\partial E_n^k}{\partial \mathbf{w}^k} \tag{7}$$

Thus, the entire objective function becomes

$$E = \sum_{k=1}^{K} E^k = \sum_{k=1}^{K} \sum_{n=1}^{N} a_n^k E_n^k. \tag{8}$$

Note that this objective function is for the update process of the RNN modules, and is not the objective function for self-organization.

3.3 What Is the Problem?

In the naive algorithm presented above, users expect the intermediate dynamics to be represented by minimizing the weighted sum of the square errors. Thus, the update process described in (7) determines whether the algorithm works as expected. Unfortunately, this naive algorithm fails to satisfy user expectations. To ascertain what the problem is, let us consider the simplest situation, in which we only have one RNN module and two training sequences. The task of the RNN module is to represent the intermediate dynamics of two training sequences. In this case, the naive algorithm becomes

$$\Delta\mathbf{w} = -\eta\left(\frac{1}{2}\frac{\partial E_1}{\partial \mathbf{w}} + \frac{1}{2}\frac{\partial E_2}{\partial \mathbf{w}}\right). \tag{9}$$

For simulation, training sequences were generated by the Hénon map,

$$\begin{cases} x_n(t+1) &= 1 - a_n x^2(t) + y(t) \\ y_n(t+1) &= b x_n(t), \end{cases} \tag{10}$$

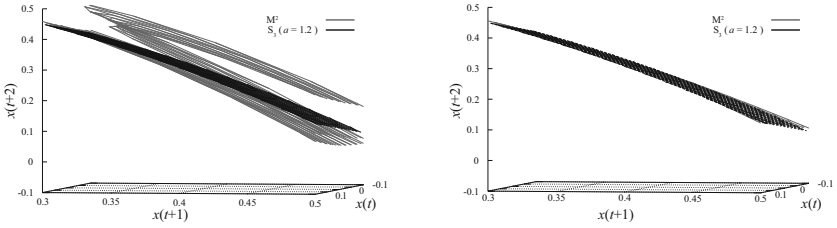


Fig. 3. Single RNN is trained by two time-series of a Hénon map with different parameters, $a = 1.4$ and $a = 1.0$. The gray surface of each panel represents the dynamics expressed by the RNN, while the black surface represents the desired dynamics when an intermediate parameter is used, i.e., $a = 1.2$. (a) The result of the naive algorithm. The RNN represents both training dynamics ($a = 1.0$ and $a = 1.4$) within the same network. (b) The result of the natural algorithm. The RNN succeeds in representing the intermediate dynamics.

where $x_n(t)$ and $y_n(t)$ are the observable and latent variables, respectively. To generate $x_1(t)$ and $x_2(t)$, $a_1 = 1.4$ and $a_2 = 1.0$ were used. Thus, the task of the RNN is to estimate the dynamics with parameter $a = 1.2$. The result, illustrated in Fig. 3(a), shows that the RNN represents *both* training dynamics simultaneously, but does not represent the intermediate dynamics.

One may be suspicious of why a single RNN can represent two dynamics simultaneously, instead of representing the intermediate dynamics. The reason is the arbitrariness of the latent variable estimation. To reproduce a training sequence, it is allowed to transform $y(t)$ in scale and bias. For example, if the latent variable $y(t)$ is transformed as $y'(t) = \alpha y(t) + \beta$, it is easy to modify the difference equations so as to produce the same $x(t)$. It is also possible to transform $y(t)$ by a monotonic nonlinear function. Therefore, the latent sequence estimation is an ill-posed problem. This fact allows the network to represents two dynamics simultaneously. Suppose that $y(t)$ is always positive, i.e., $y(t) > 0$, when the RNN represents $x_1(t)$, whereas $y(t) < 0$ for $x_2(t)$. Then the RNN can represent both dynamics within a single architecture by segregating the latent variable regions. Obviously this case minimizes the square error for both training sequences. It is also worth stressing that no training data for the intermediate dynamics is given to the RNN.

4 Natural Algorithm for LAMD

4.1 Improvement of Naive Algorithm

As pointed out above, the arbitrariness of the latent variable estimation causes the problem. Though it is rather a heuristic improvement, let us first try to fix the problem by the following ad hoc modifications.

Modification 1: In the winner module k_n^*, the estimated latent variable is normalized so that its probability density is almost equal for all training sequences. One of the easiest ways is to normalize the latent sequence, so that the maximum and minimum

of $\hat{y}_n^*(t)$ are both ± 1. The weight connections of the winner module also need to be compensated so as to produce the same output.

Modification 2: The normalized latent sequence of the winner module is shared by all modules. Thus, the estimated latent sequences in the non-winner modules are all rejected, and they are replaced by the winner's latent sequence.

Modification 1 makes the probability density $p(\hat{y}_n^k)$ consistent for all training sequences, i.e., independent of n, whereas Modification 2 makes $\hat{y}_n^k$ consistent for all modules, i.e., independent of k. This modified algorithm is hereafter referred to as the *natural algorithm*.

Simulation results for the natural algorithm using a Hénon map are shown in Fig. 3(b). Using the natural algorithm, the network succeeds in representing the intermediate dynamics. Again it is worth noting that there is no training data for the intermediate dynamics.

4.2 RNN-mnSOM and RNNPB with the Natural Algorithm

To compare the naive and natural algorithms, both algorithms were programmed into an RNN-mnSOM and RNNPB. To investigate the difference, time-series of simple harmonic waves were used for simulation. The difference equation is given by

$$\begin{pmatrix} x_n(t+1) \\ y_n(t+1) \end{pmatrix} = \begin{pmatrix} \cos\omega_n & \sin\omega_n \\ -\sin\omega_n & \cos\omega_n \end{pmatrix} \begin{pmatrix} x_n(t) \\ y_n(t) \end{pmatrix}, \tag{11}$$

where $x_n(t)$ and $y(t)$ denote the observable and latent sequences, respectively. The parameters used for the training sequences were $\omega_n = 0.8, 1.0, 1.2, 1.4$, while other values between 0.7 and 1.5 were used for the test.

The tasks for the RNN-mnSOM and RNNPB were set as follows: (i) learn four training sequences, (ii) sort them into a one-dimensional parameter space, and (iii) interpolate between the dynamics of the training sequences. To achieve these, the RNN-mnSOM was given 7 RNN modules with a one-dimensional feature space, while the RNNPB had 1 parametric bias unit.

Fig. 4 gives the results for the RNN-mnSOM and RNNPB. In both cases, the naive algorithm failed to order the given sequences, while the natural algorithm succeeded.

5 What Is Required in Learning of Multiple Dynamics?

As observed above, consistent estimation of latent variables seems to be vitally important, and appears to be the common principle in the LAMD. In this section we attempt to outline the theory underlying the LAMD.

To discuss this issue, we first need to define the distance measure that governs the task. The most natural would be to use the distance measure in function space. Thus, the distance between two dynamics is defined as follows.

$$L^2(f_1, f_2) \triangleq \int \|f_1(\mathbf{z}) - f_2(\mathbf{z})\|^2 p(\mathbf{z}) d\mathbf{z} \tag{12}$$

$$= \int \|f_1(\mathbf{x}, \mathbf{y}) - f_2(\mathbf{x}, \mathbf{y})\|^2 p(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y} \tag{13}$$

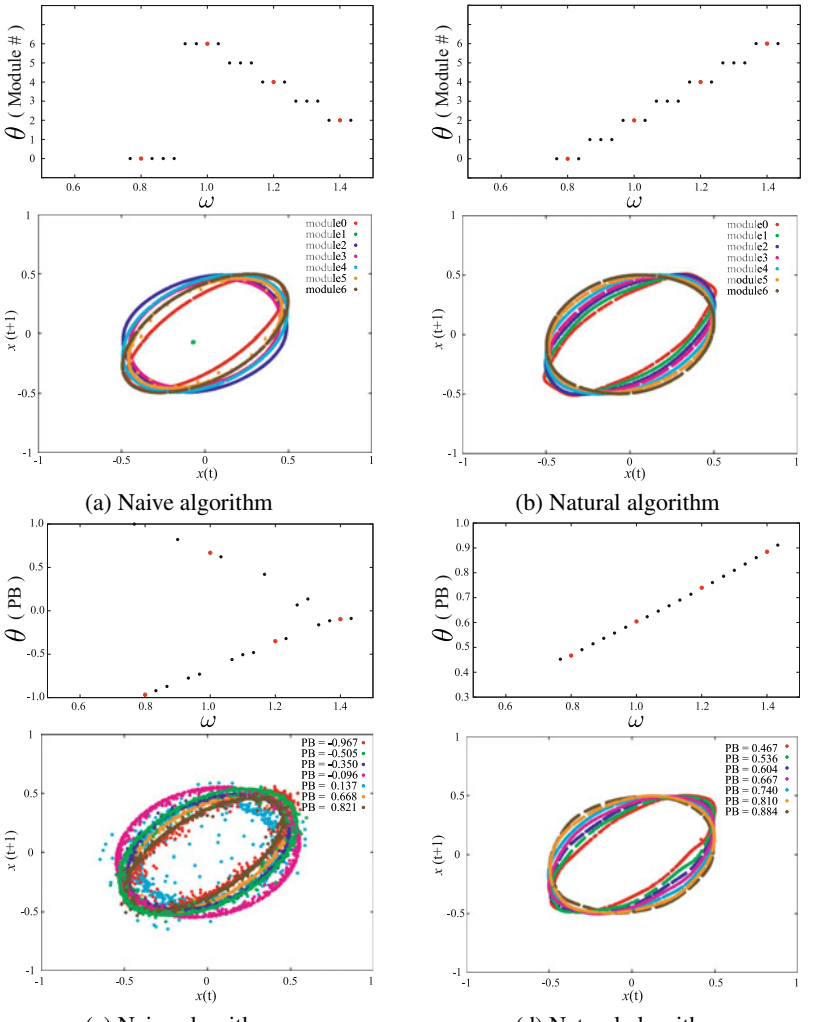


Fig. 4. Results for RNN-mnSOM (a) (b) and RNNPB (c) (d) with Naive and Natural algorithms, respectively

Here, f_1 and f_2 are the difference or the differential equations of the two dynamics. It is necessary to emphasize that the aim of the architecture is to deal with a set of dynamics which can be defined by a set of difference or differential equations. Therefore the distance between equations is more essential than the distance between observed time sequences. In (13) $p(\mathbf{z}) = p(\mathbf{x}, \mathbf{y})$ is the probability density function of the state variable. If the probabilities of the observable and latent variables are assumed to be independent, then (13) becomes

$$L^2(f_1, f_2) = \int \|f_1(\mathbf{x}, \mathbf{y}) - f_2(\mathbf{x}, \mathbf{y})\|^2 p(\mathbf{x}) p(\mathbf{y}) d\mathbf{x} d\mathbf{y}. \tag{14}$$

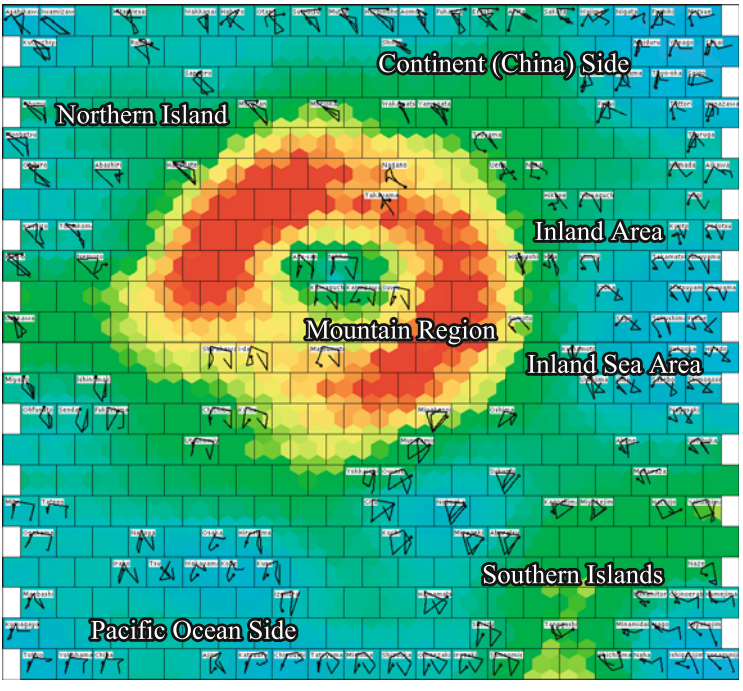


Fig. 5. A map of weather dynamics organized by a SOM^2. Weather trajectories during one week for 153 cities are indicated in the winner node.

It is also worth stressing that the density function $p(\mathbf{x}, \mathbf{y})$ should be common for all $f_n(\mathbf{x}, \mathbf{y})$, otherwise the distance cannot be defined. As mentioned above, the latent variable estimation suffers from the arbitrariness problem, and the PDF $p(\mathbf{y})$ changes depending on the modules and the sequences. This is why consistent estimation of the latent variable in the natural algorithm is necessary.

Though the above heuristic natural algorithm for the RNN-mnSOM and RNNPB performed well in the simulations, the algorithm still has difficulty in more practical cases. One of the reasons is that the probability density of the observable variable $p(\mathbf{x})$ also depends on the internal parameter θ in many cases. Furthermore, they are usually high dimensional vectors distributed in a nonlinear subspace in a high dimensional data space. In such cases, a simple normalization of each component of $\mathbf{x}$ is not enough, because each component is not usually independent to others. Therefore, we need a true natural algorithm that can be derived theoretically, rather than heuristically.

Now let us consider the case in which $p(\mathbf{x}, \mathbf{y})$ depends on the internal parameter θ. Furthermore, $\mathbf{x}(t)$ is supposed to be distributed nonlinearly in a high-dimensional space. In such a case, the dynamics is formulated as follows.

$$\boldsymbol{\xi}(t+1) = f(\boldsymbol{\xi}(t)) \tag{15}$$

$$\mathbf{z} = (\mathbf{x}, \mathbf{y}) = g(\boldsymbol{\xi}; \theta) \tag{16}$$

Here, $\boldsymbol{\xi}$ is the low-dimensional intrinsic state variable that governs the class of dynamics, and the mapping from intrinsic state to actual variables is supposed to be modified by θ. In other words, an intrinsic dynamics for $\boldsymbol{\xi}(t)$ exists, and the observation is altered by the context or the environment. Obviously this formulation includes the previous simple cases. By applying embedded theory, (16) is equivalent to

$$\tilde{\mathbf{z}}(t) \triangleq (\mathbf{x}(t), \mathbf{x}(t-1), \ldots, \mathbf{x}(t-L+1)) \tag{17}$$

$$\tilde{\mathbf{z}}(t) = \tilde{g}(\boldsymbol{\xi}; \theta). \tag{18}$$

Therefore, such a class of dynamics can be described by a homotopy $\tilde{g}(\boldsymbol{\xi}; \theta)$. Thus, what we need is an algorithm for self-organizing homotopy learning. The best suited solution for this purpose is a 'SOM of SOMs', that is, a SOM^2, which is an extension of the self-organizing map to a homotopy [12]. The SOM^2 represents the intrinsic state by a set of 'fibers' and the entire class of dynamics is represented by a fiber bundle.

A map of dynamics organized by a SOM^2 is shown in Fig. 5. This is a map of weather dynamics for 153 cities in Japan. The SOM^2 succeeds in representing and ordering the weather dynamics for the given cities.

6 Conclusion

In this paper, we discussed what is required for learning of multiple dynamics, and pointed out the importance of the natural algorithm. Currently, the SOM^2 is the best solution, but a theoretical derivation has not yet been done. This remains a future work for us.

Acknowledgement. This work is supported by KAKENHI 22120510A10.

References

1. Wolpert, D.M., Kawato, M.: Neural Networks 11, 1317–1329 (1998)
2. Minamino, K.: Developing Intelligence. Intelligence Dynamics Series, vol. 3, pp. 73–111. Springer, Japan (2008) (in Japanese)
3. Minatohara, T., Furukawa, T.: Proc. WSOM 2005, pp. 41–48 (2005)
4. Minatohara, T., Furukawa, T.: IJICIC 7 (in press, 2011)
5. Tokunaga, K., Furukawa, T.: Neural Networks 22, 82–90 (2009)
6. Yin, H., Ni, H.: Generalized Self-Organizing Mixture Autoregressive Model. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 353–361. Springer, Heidelberg (2009)
7. Jordan, M.I.: ICS Report, 8604 (1986)
8. Tani, J.: IEEE Trans. SMC Part A 33, 481–488 (2003)
9. Ohkubo, T., Tokunaga, K., Furukawa, T.: Int. Congress Series, vol. 1301, pp. 168–171 (2007)
10. Ohkubo, T., Tokunaga, K., Furukawa, T.: IEICE Trans. Inf. & Sys. E92-D, 1388–1396 (2009)
11. Kaneko, S., Furukawa, T.: Proc. WSOM 2005, pp. 537–544 (2005)
12. Furukawa, T.: Neural Networks 22, 463–478 (2009)

Growing Graph Network Based on an Online Gaussian Mixture Model

Kazuhiro Tokunaga

Kyushu Institute of Technology,
2-4 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka, Japan
tokunaga@brain.kyutech.ac.jp

Abstract. In this paper, the author proposes a growing neural network based on an online Gaussian mixture model, in which mechanisms are included for growing Gaussian kernels and finding topologies between kernels using graph paths. The proposed method has the following advantages compared with conventional growing neural networks: no permanent increase in nodes (Gaussian kernels), robustness to noise, and increased speed of constructing networks. This paper presents the theory and algorithm for the proposed method and the results of verification experiments using artificial data.

Keywords: Growing Neural Networks, Gaussian Mixture Model, Graph, Online.

1 Introduction

A growing neural network (GNN), such as the Growing Neural Gas (GNG) [1], represents the topology of data with a graph network using online-learning data vectors input sequentially. Finding the topology of input data vectors is important in various applications, such as object recognition, character recognition, structure recognition, and so on. It is expected that a GNN could be used in the learning system for a mobile robot, since the GNN can find the topology of data dynamically using online learning.

In the past, various algorithms for GNNs have been proposed. However, all these algorithms build the graph network directly from observed data vectors. In other words, the graph networks built by conventional GNNs do not represent the generative model of the observed data. Conventional GNNs are prone to the following problems: sensitivity to noise, generation of redundant nodes, and having the learning results depend heavily on the learning parameters.

The aim of this work is to develop an algorithm for the GNN from the perspective of a generative model. In this paper, an algorithm for the GNN based on an online Gaussian Mixture Model (GMM) is proposed; henceforth, this proposed method is referred to as the "GGMM: Growing GMM". The GMM represents a probability density function (PDF) combining multiple Gaussian kernels. Thus, the GMM builds the generative model represented by the PDF. The GGMM is extension of the GMM that includes the following mechanisms: online

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 111–120, 2011.

learning, growing of Gaussian kernels and finding topologies between kernels using graph paths.

This paper discusses the theory and an algorithm for the GGMM. In addition, results of experiments comparing the GGMM with three typical GNNs (Growing Neural Gas, Evolving SOM [2], and Self-Organizing Incremental Neural Network [3]) are presented.

2 Framework and Theory

The proposed method, the GGMM, performs three processes concurrently: (1) online parameter estimation of the GMM, (2) controlling the generation of Gaussian kernels using an information criterion, and (3) generation of a path representing the topology between kernels and updating the path's strength. First, a framework for the proposed method is presented, and then each of the above processes are explained in turn.

2.1 Framework

The framework of tasks to which the proposal method will be applied, is given below.

- Sequentially-observed data vectors cannot be stored in memory.
- The appropriate number of Gaussian kernels is unknown. (In the first stage of learning, the number of kernels is one.)
- The parameters (mixing parameter, and mean and covariance matrices) of each kernel are unknown.
- The class information for data vectors is unknown.
- Noise is added to the data vector.

Under the above conditions, processes to find the generative model of the input data and generate the paths are performed simultaneously in online learning.

2.2 (1) Online GMM

The GMM is widely known as a nonparametric approach for estimating the probability density function and represents the probability density function with a mixture of multiple Gaussian kernels.

Now, at time t, let a d-dimensional data vector $\boldsymbol{x}_t$ be observed. Then, in the GMM composed of K kernels, the probability density function $p(\boldsymbol{x}_t)$ is described as

$$p(\boldsymbol{x}_t) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \quad \sum_{k=1}^{k} \pi_k = 1 , \tag{1}$$

where

$$\mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi)^{d/2} \mid \boldsymbol{\Sigma}_k \mid^{1/2}} \exp\left\{ -\frac{1}{2} (\boldsymbol{x}_t - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_t - \boldsymbol{\mu}_k) \right\} . \tag{2}$$

Here, the Gaussian kernel is represented by a normal probability distribution $\mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_k, and \boldsymbol{\Sigma}_k)$. π^k is the mixing parameter of the k-th kernel. Moreover, $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ are the mean and covariance matrices, respectively, in the k-th Gaussian kernel.

Online Parameter Estimation

In online parameter estimation for an online GMM, the extremum of the objective function

$$L(\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \log \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} - \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right) , \tag{3}$$

defined using Lagrange's method for undetermined multipliers, is found by the hill climbing method. Updating expressions for each parameter are defined as follows:

$$\boldsymbol{\mu}_k^{\text{new}} = \boldsymbol{\mu}_k^{\text{old}} + \epsilon \, (\boldsymbol{\Sigma}_k^{\text{old}})^{-1} \gamma_k(\boldsymbol{x}_t) \left\{ \boldsymbol{x}_t - \boldsymbol{\mu}_k^{\text{old}} \right\}, \tag{4}$$

$$\boldsymbol{\Sigma}_k^{\text{new}} = \boldsymbol{\Sigma}_k^{\text{old}} + \frac{1}{2} \, \epsilon \, \gamma_k(\boldsymbol{x}_t) (\boldsymbol{\Sigma}_k^{\text{old}})^{-1} \left\{ (\boldsymbol{x}_t - \boldsymbol{\mu}_k^{\text{new}})(\boldsymbol{x}_t - \boldsymbol{\mu}_k^{\text{new}})^T (\boldsymbol{\Sigma}_k^{\text{old}})^{-1} - \boldsymbol{I} \right\}, \tag{5}$$

$$\pi_k^{\text{new}} = \pi_k^{\text{old}} + \epsilon \left\{ \frac{\gamma_k(\boldsymbol{x}_t)}{\pi_k^{\text{old}}} - 1 \right\} . \tag{6}$$

Here, ϵ is a learning rate, such that $0 < \epsilon < 1.0$. Moreover, $\gamma_k(\boldsymbol{x}_t)$ is the posterior probability (responsibility) defined as:

$$\gamma_k(\boldsymbol{x}_t) = \frac{\pi_k^{\text{old}} \mathcal{N}\left(\boldsymbol{x}_t; \boldsymbol{\mu}_k^{\text{old}}, \boldsymbol{\Sigma}_k^{\text{old}}\right)}{\sum_{k'=1}^{K} \pi_{k'}^{\text{old}} \mathcal{N}\left(\boldsymbol{x}_t; \boldsymbol{\mu}_{k'}^{\text{old}}, \boldsymbol{\Sigma}_{k'}^{\text{old}}\right)} . \tag{7}$$

2.3 (2) Generation of Gaussian Kernel Using Information Criterion

Typical information criteria include Akaike's Information Criterion (AIC) [4] and the Bayesian Information Criterion (BIC) [5]. In this paper, AIC is used to generate the Gaussian kernel, since . The choice of information criterion is not important, as the difference in learning results using the AIC and BIC is negligible. The AIC is defined as:

$$\text{AIC}(K) = -2 \ln l(K) + 2 \, C(K) . \tag{8}$$

Here, K, $L(K)$, and $C(K)$ are the number of kernels, the maximum likelihood in K kernels, and the degrees of freedom of the model, respectively. In the GMM, the number of kernels is chosen to maximize AIC(K). In other words, the number of kernels is determined so that the maximum likelihood is given by as few kernels as possible. Generally, in the GMM, the number of kernels is controlled offline using all the input data. However, the GGMM controls the generation of the kernel from a single observed data vector in online learning.

Next, the mechanism for the online-generation of the kernel using the AIC is explained. Suppose that data vectors $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_{T-1}$ have been observed at time $T-1$. Now, at time T, let a new input data vector $\boldsymbol{x}_T$ be observed. At this time, a new kernel is generated if $\mathrm{AIC}(K+1) > \mathrm{AIC}(K)$, that is,

$$\ln l(K+1) - \ln l(K) > C(K+1) - C(K) \tag{9}$$

is implemented. $l(K+1)$ is the likelihood of the GMM, in which a new kernel has been added. Here, $l(K+1)$ and $l(K)$ are expressed as:

$$l(K) = \prod_{t=1}^{T} p_K(\boldsymbol{x}_t), \tag{10}$$

$$l(K+1) = \prod_{t=1}^{T} p_{K+1}(\boldsymbol{x}_t) \ . \tag{11}$$

However, it is impossible to calculate Equation (11), since previously processed input data cannot be stored in this framework (see Section 2.1). Therefore, Equation (11) is evaluated approximately using a current input data vector $\boldsymbol{x}_T$. The approximate evaluation of Equation (11) is computed as

$$\ln p_{K+1}(\boldsymbol{x}_T) - \ln p_K(\boldsymbol{x}_T) > C(K+1) - C(K) + 1 \ . \tag{12}$$

The derivation of Equation (12) is described below.

Derivation of Equation (12)

Let the probability density function $p_{K+1}(\boldsymbol{x})$ at the addition of a new kernel be defined as follows:

$$p_{K+1}(\boldsymbol{x}) = \sum_{k=1}^{K} \pi'_k \mathcal{N}\left(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) + \pi_{K+1} \mathcal{N}\left(\boldsymbol{x}; \boldsymbol{\mu}_{K+1}, \boldsymbol{\Sigma}_{K+1}\right) \ . \tag{13}$$

Here, let $\pi'_k = (1-\tau)\pi_k$ where $0 < \tau < 1$. Moreover, the initial values of parameters, π_{K+1}, $\boldsymbol{\mu}_{K+1}$, and $\boldsymbol{\Sigma}_{K+1}$ in the new kernel are given by

$$\pi_{K+1} = \tau, \ \boldsymbol{\mu}_{K+1} = \boldsymbol{x}_T, \ \boldsymbol{\Sigma}_{K+1} = \alpha \boldsymbol{I} \ , \tag{14}$$

where α is an arbitrary invariable.

The following equation is derived from Equations (11) and (13):

$$l(K+1) = \prod_{t=1}^{T} \left\{ \sum_{k=1}^{K} \pi'_k \mathcal{N}\left(\boldsymbol{x}_t; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) + \pi_{K+1} \mathcal{N}\left(\boldsymbol{x}_t; \boldsymbol{\mu}_{K+1}, \boldsymbol{\Sigma}_{K+1}\right) \right\} \ . \tag{15}$$

Here, in the second term on the right-hand side, suggested that the contribution of foregone input data $\boldsymbol{x}_t, \ \ t = 1, 2, ..., T-1$ can be ignored. Then, Equation (15) can be approximated as:

$$l(K+1) \simeq \left\{ \prod_{t=1}^{T-1} \left(\sum_{k=1}^{K} \pi'_k \mathcal{N}\left(\boldsymbol{x}_t; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) \right) \right\} p_{K+1}(\boldsymbol{x}_T) \tag{16}$$

$$= \left\{ \prod_{t=1}^{T-1} (1-\tau) \left(\sum_{k=1}^{K} \pi_k \mathcal{N}\left(\boldsymbol{x}_t; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right) \right) \right\} p_{K+1}(\boldsymbol{x}_T) \tag{17}$$

$$= \left\{ \prod_{t=1}^{T-1} (1-\tau)\, p_K(\boldsymbol{x}_t) \right\} p_{K+1}(\boldsymbol{x}_T) \tag{18}$$

$$= (1-\tau)^{T-1} \frac{l(K)}{p_K(\boldsymbol{x}_T)} p_{K+1}(\boldsymbol{x}_T) \ . \tag{19}$$

Therefore,

$$\frac{l(K+1)}{l(K)} \simeq (1-\tau)^{T-1} \frac{p_{K+1}(\boldsymbol{x}_T)}{p_K(\boldsymbol{x}_T)} \ . \tag{20}$$

Finally, Equation (12) is derived by taking the logarithm of both sides of Equation (20). Here, when $T \gg 1$,

$$(T-1)\ln(1-\tau) \simeq -1. \tag{21}$$

Therefore, it is possible to control the generation of the kernel using only the current input data.

2.4 (3) Generating and Updating the Path

The GGMM simultaneously generates paths between the existing kernels and the new kernel. These paths represent the topology between the data distributions on the Gaussian kernels. In addition, a strength, which can be decreased or increased with learning, is associated with each path.

The Mahalanobis distance between kernels is used to generate and update the path. If the Mahalanobis distance between kernels is small, then the probability that data vectors are distributed between these kernels is high. Thus, the probability that each kernel belongs to the same class is high. By contrast, if the Mahalanobis distance between kernels is large, the probability that each kernel is independent is high. Thus, the strength of the path represents the statistical distance between kernels. Next, the underlying theory for "generating the path" and "updating the path" are explained.

Generating the Path

A path is generated between the additional kernel and both the first and second winner kernels, where the first and second winner kernels, k^* and k^{**}, are defined as:

$$k^* = \arg\min_k \gamma_k(\boldsymbol{x}_T) \quad {}^\forall k, \tag{22}$$

$$k^{**} = \arg\min_k \gamma_k(\boldsymbol{x}_T) \quad {}^\forall k \notin k^* \ . \tag{23}$$

In the first generation of the path, the initial strength of the path is set according to the following definitions. Here, the notations for the variables are given as follows. Suppose that $s(K+1,k^*)$ and $s(K+1,k^{**})$ are the strengths of the paths between $K+1$ and k^*, and $K+1$ and k^{**}, respectively. In addition, the Mahalanobis distance between $\boldsymbol{\mu}^{K+1}$ and $\boldsymbol{\mu}^{k^*}$ on the additional kernel $K+1$ is expressed as d_{K+1}^{K+1,k^*}. Then, the following equations describe the initial strength of the path.

$$s(K+1,k^*) = \beta \left(\frac{1}{\left(d_{K+1}^{K+1,k^*} + d_{k^*}^{K+1,k^*} \right)/2} \right) \tag{24}$$

$$s(K+1,k^{**}) = \beta \left(\frac{1}{\left(d_{K+1}^{K+1,k^{**}} + d_{k^{**}}^{K+1,k^{**}} \right)/2} \right) \tag{25}$$

β is an arbitrary parameter, such that $0 < \beta < 1.0$.

Updating the Path

A necessary and an unnecessary path are highlighted by updating the strength of the path through learning. All paths connected to the first winner kernel are updated using the equation below.

$$s(k^*,k')^{\text{new}} = (1-\beta)s(k^*,k')^{\text{old}} + \beta \left(\frac{1}{\left(d_{k^*}^{k^*,k'} + d_{k'}^{k^*,k'} \right)/2} \right) \tag{26}$$

Thus, the strength of the path represents the expected value of the reciprocal of the Mahalanobis distance between the kernels.

3 Algorithm

The algorithm for the GGMM consists of four processes: (1) evaluation process, (2-A) generation process, (2-B) update process, and (3) deletion process.

The following processes are repeated during learning.

(0) Initial Process

The initial number of kernels is 0. When the first observed data vector $\boldsymbol{x}_1$ is given, the first kernel composed of the initial parameter in Equation (14) is generated.

(1) Evaluation Process

The probability density function $p_K(\boldsymbol{x}_t)$ in Equation (1) and the assumed probability density function $p_{K+1}(\boldsymbol{x}_t)$ in Equation (13) are calculated from the observed data vector $\boldsymbol{x}_t$. Next, whether to generate a new kernel is determined by Equation (12). If Equation (12) is true, then go to (2-A), the generation process. If Equation (12) is false, then go to (2-B), the update process.

(2-A) Generation Process

In this process, a new kernel is generated. The initial parameters of the new kernel are set according to Equation (14). Next, paths are generated between the additional kernel and both the first and second winner kernels, using Equation (25). Go to (3) when the above process terminates.

(2-B) Update Process

The parameters of the generated kernels are updated by Equations (4), (5), and (6). Next, the paths connected to the first winner kernel are updated by Equation (26). Furthermore, if the first and second winner kernels are not connected, a new path is generated between their kernels. Go to (3) when the above process terminates.

(3) Deletion Process

A kernel whose mixing parameter is close to 0 is deleted. In this work, a kernel that falls below the threshold parameter is also deleted. This kernel deletion is performed at each learning step.

Furthermore, the strength of path that falls below the threshold parameter is deleted. In this work, the threshold parameter is $1/7$, which is the reciprocal of the Mahalanobis distance 7, since the strength of path represents the expected value of the reciprocal of the Mahalanobis distance between kernels. It is recommended that the threshold is determined according to the tasks. This deletion is performed at intervals.

Return to (1) if a data vector is observed. Otherwise, the learning is complete.

4 Simulation

In this simulation, the proposal method is compared with three typical GNNs (GNG, ESOM, and SOINN).

The Swiss roll data, shown in Fig. 1, were used as artificial data for the simulation. Data vectors are distributed on two spirals. Two kinds of Swiss roll data (Type 1, Type 2), differing with respect to the width of noise as shown in Fig. 1 were used, with a total of 10000 data vectors.

It is difficult to evaluate each method within the same framework, since the behavior of the learning parameters is different for each method. Therefore, in this simulation, the parameters for each method were set so that two spiral graph networks could be generated for Type 1 data. For both Type 1 and Type 2 data, the learning parameters for each GNN were the same.

The learning results obtained from data vectors of Type 1 for each GNN are shown in Fig. 2. Similarly, the learning results for Type 2 data are shown in Fig. 3. According to the results for Type 1 data, all methods can successfully represent two spiral graph networks. By contrast, in the results for the GNG, SOINN, and ESOM with Type 2 data, the spiral data distribution cannot be represented, since some nodes are allocated to noise. It is difficult to obtain the

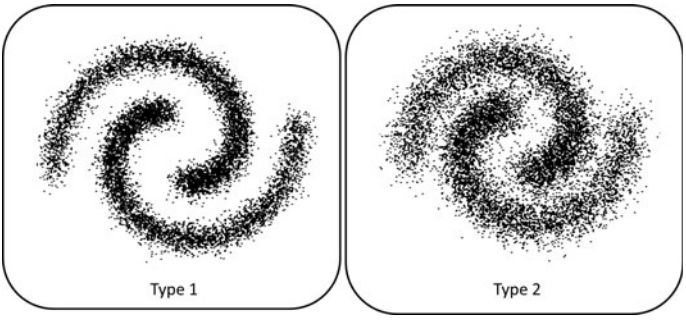


Fig. 1. Two types of Swiss roll data

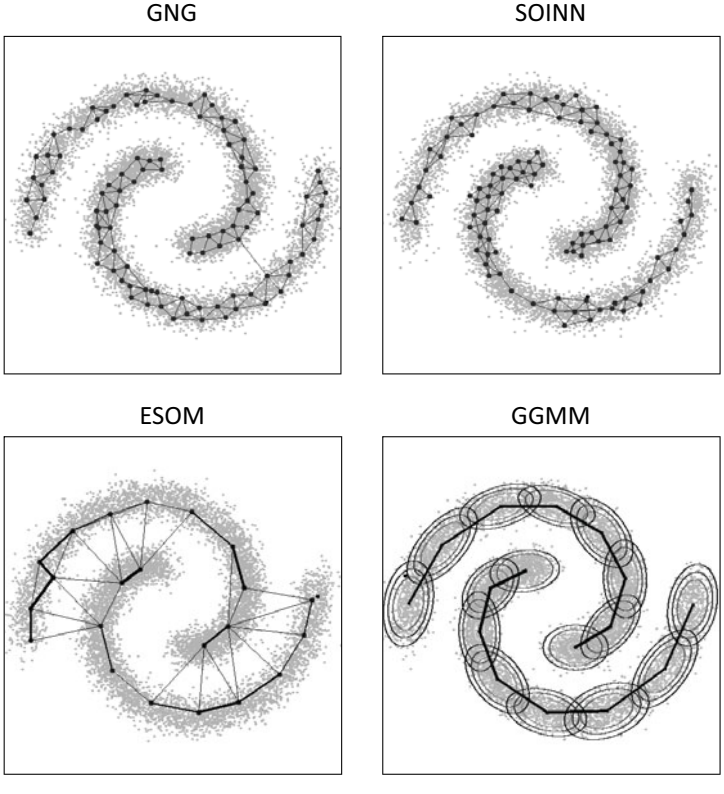


Fig. 2. Results for Type 1 data

desired results using those methods, for which the probability density functions of the data distributions cannot be estimated. Additionally, with the Type 2 data, the GNG, ESOM, and SOINN cannot generate two spiral graph networks no matter how the parameters are adjusted. By contrast, the GGMM successfully

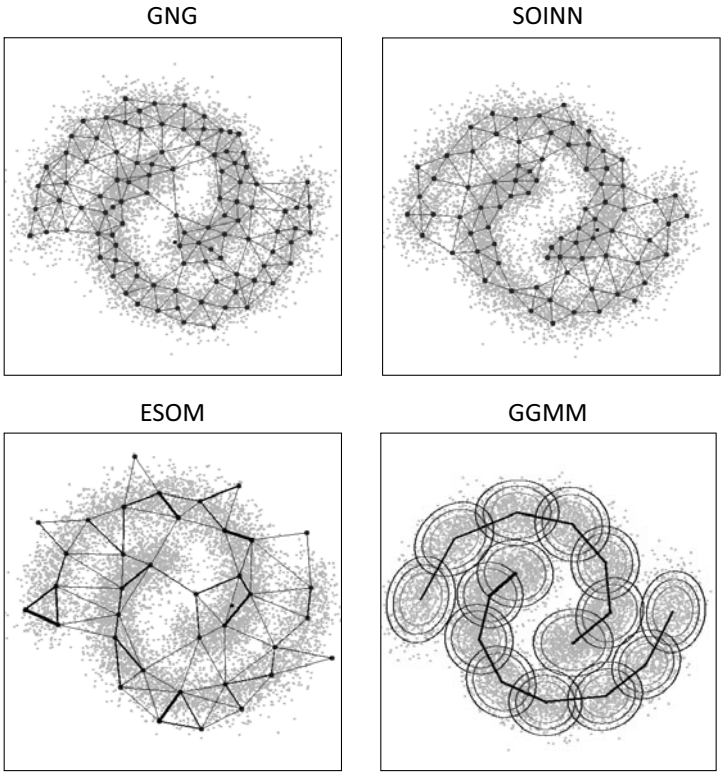


Fig. 3. Results for Type 2 data

formed graph networks, in which the two spirals were well separated. Moreover, in several-time experiments, almost the same results were obtained, since the influence of the parameters of the GGMM on the result is small. In addition, since the number of nodes in the GGMM does not increase permanently, almost the same results were obtained in the several-time experiments.

These results confirm that the proposed method yields stable learning results and is more robust to noise than conventional GNNs.

5 Summary

In this paper, the author proposed a growing neural network based on an online type Gaussian mixture model, which includes mechanisms to grow Gaussian kernels and find topologies between kernels using graph paths. In the simulation, the proposed method obtains stable learning results, and is more robust to noise than conventional GNNs. It is expected that the proposed method can be applied as the fundamental algorithm in a system where a cognitive model is developed from dynamically input sensor data in mobile robots. In a past work, the Self-Evolving Modular network (SEEM) was proposed as a method for multi-dynamic

learning in mobile robots. The SEEM increases the number of modules and represents two or more dynamics. It is anticipated that this proposed method can be applied as the backbone algorithm of the SEEM. In the future, author aim to derive more theoretically consolidated GNN algorithms based on Bayesian estimation.

Acknowledgement

This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research on Innovative Areas, 22120510A10.

References

1. Fritzke, B.: A growing neural gas network learns topologies. Advances in Neural Information Processing Systems 7, 625–632 (1995)
2. Deng, D., Kasabov, N.: On-line pattern analysis by evolving self-organizing maps. Neurocomputing 51, 87–103 (2003)
3. Shen, F., Hasegawa, O.: An incremental network for on-line unsupervised classification and topology learning. Neural Networks 19(1), 90–106 (2006)
4. Akaike, H.: A new look at the statistical model identification. IEEE Transaction on Automatic Control 19(6), 716–723 (1974)
5. Schwarz, Gideon, E.: Estimating the dimension of a model. Annals of Statistics 6(2), 461–464 (1978)

Evolving a Self-Organizing Feature Map for Visual Object Tracking

José Everardo B. Maia[1], Guilherme A. Barreto[2], and André Luís V. Coelho[3]

[1] Department of Statistics and Computing
State University of Ceará (UECE), Fortaleza, Ceará, Brazil
jebmaia@gmail.com
[2] Department of Teleinformatics Engineering
Federal University of Ceará (UFC), Fortaleza, Ceará, Brazil
guilherme@deti.ufc.br
[3] Postgraduate Program in Applied Informatics
University of Fortaleza (UNIFOR), Fortaleza, Ceará, Brazil
acoelho@unifor.br

Abstract. An extension of a recently proposed evolutionary self-organizing map is introduced and applied to the tracking of objects in video sequences. In the proposed approach, a geometric template consisting of a small number of keypoints is used to track an object that moves smoothly. The coordinates of the keypoints and their neighborhood relations are associated with the coordinates of the nodes of a self-organizing map that represents the object. Parameters of a local affine transformation associated with each neuron are updated by an evolutionary algorithm and used to map each template's keypoint in the previous frame to the current one. Computer simulations indicate that the proposed approach presents better results than those obtained by a direct method approach.

Keywords: Self-organizing neural networks, evolutionary algorithms, object tracking, video sequences.

1 Introduction

Visual tracking is the act of consistently locating a region in each image of a video sequence that matches the given object [8]. It is a critical step in many machine vision applications such as surveillance [2], driver assistance systems [9], remote sensing, defense systems [1], human-computer interactions [6].

In this paper, we propose a novel strategy that consists in using simple representations for the patch centered on keypoints and computationally efficient measures of matching to compare patches. For this purpose, we build a self-organizing map on the object model image with the nodes located in the keypoints using distance and the neighborhood relations to connect them. The neighborhood relations imposed by the topographic map impose constraints that prevent the drift of the points in successive iterations. In our approach the quality of a solution considers the matching of all patches and also the correlation of distances between points in the center of each patch.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 121–130, 2011.

The remainder of this paper is organized as follows. In Section 2 and its subsections the problem is defined and the proposed approach is introduced. In Section 3 the simulation results are presented and discussed. The article is concluded in Section 4.

2 The Proposed Approach

Firstly, we need to define the reference, current and candidate templates. Let $I = \{I_0, I_1, .., I_i\}$ be a sequence of indexed images and $T_0, .., T_i$ are gray-level intensities of templates defined on these images. The template (or patch) defined in the first frame, T_0, is referred to as the *reference template* (or the reference patch). When tracking from frame i to frame $i+1$, we refer to frame i as the *current frame*, and the template within this frame, T_i, as the *current template*. The frame $i+1$ is referred to as the *target frame*, and a template within this frame, T_{i+1}, as a *candidate template*.

The Sum of Squared Differences (SSD) is used as a measure of matching between templates. Let $\mathbf{x} \in T_0$ be a feature point in the corresponding template. Thus, the problem of finding a transformation parameter vector $\mathbf{p}$ between T_0 and T_i is formulated using SSD as

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{\mathbf{x} \in T_0} \left[T_i(\mathbf{x}') - T_0(\mathbf{x})\right]^2 = \arg\min_{\mathbf{p}} \sum_{\mathbf{x} \in T_0} \left[T_i(w(\mathbf{x}, \mathbf{p})) - T_0(\mathbf{x})\right]^2, \quad (1)$$

where $\mathbf{x}' = w(\mathbf{x}, \mathbf{p})$ is the projection of the feature point $\mathbf{x} \in T_0$ onto the current frame i. The SSD-based tracking problem can thus be stated as the task whose goal is to select and track feature points from images I_0 to I_{i+1}. Assuming that the transformation $w(\mathbf{x}, \hat{\mathbf{p}})$ from frame 0 to the current frame i is known, the problem reduces to finding an increment $\Delta\mathbf{p}$ for the transformation parameter vector between T_i and T_{i+1} through an iterative method that solves

$$\Delta\hat{\mathbf{p}} = \arg\min_{\Delta\mathbf{p}} \sum_{\mathbf{x}' \in T_i} \left[T_{i+1}(w(\mathbf{x}', \Delta\mathbf{p})) - T_i(\mathbf{x}')\right]^2, \quad (2)$$

By function composition, we find the whole transformation imposed to the feature point $\mathbf{x} \in T_0$ from frame 0 (reference) to frame $i+1$ (target): $\mathbf{x}^* = w(\mathbf{x}', \Delta\hat{\mathbf{p}}) \circ w(\mathbf{x}, \hat{\mathbf{p}})$, where the feature point $\mathbf{x}'$ belongs to the current frame i (i.e. $\mathbf{x}' \in T_i$). The transformation $w : R^2 \rightarrow R^2$ is the warp function corresponding to a transformation whose parameters are specified in $\mathbf{p}$. Usually, it is given by an affine transformation $\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{b}$, defined as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} s \cdot \cos(\theta) & s \cdot \sin(\theta) \\ -s \cdot \sin(\theta) & s \cdot \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} b_x \\ b_y \end{pmatrix} \quad (3)$$

where the matrix $\mathbf{A} \in \mathbb{R}^{2\times 2}$ accounts for rotations and scaling, since θ is an angle of rotation and s is a scale factor, while $\mathbf{x} = (x, y)$, $\mathbf{x}' = (x', y')$ and $\mathbf{b} = (b_x, b_y)$ denote respectively the original positions, the transformed ones and a translation vector.

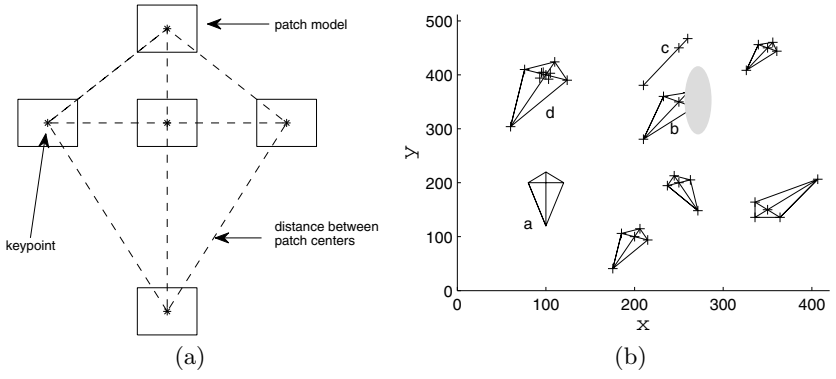


Fig. 1. (a) A kite-shaped template with 5 patches and 8 distance links. (b) Typical aspects that a kite-shaped template can assume during the tracking problem.

We assume that the object is executing a smooth movement whose evolution from one frame to the next is represented by affine transformations whose parameter vectors $\mathbf{p} = (b_x, b_y, \theta, s)$. This condition is approximately satisfied when the object is rigid, the camera is stationary and the object's movement is slow relative to the video frame rate such that location, scalar velocity and direction of motion of a given point change little from one frame to the next.

2.1 Object Representation

We tackle the tracking problem as a problem of matching detected keypoints between successive frames. By keypoint we denote a point in an image that is sufficiently different from its neighbors so that it can be easily distinguished from other similarly extracted points in the same or another image. It is assumed that a small neighborhood is also moving together with the point and therefore a small image patch around the point, called *model patch*, can be considered for analysis.The keypoints are manually marked by the user on the object image in the first frame.

By taking the keypoints selected as vertices, a template in the form of an undirected graph (or grid) is built to represent the object (see Figure 1a). A planar grid is built by establishing links between some of the keypoints, which impose significant constraints on the geometric appearance of the object. Although the links can be arbitrary, distances and neighborhood relations between the keypoints should be considered to define the links. The vertices of the template can then be interpreted as the coordinates of a non-regular grid defining a self-organizing feature map (SOFM) that represents the object. The weight vectors of each node in the output grid are updated by an evolutionary algorithm and used to locate the object in a frame by frame basis. For this example, we defined five keypoints (hence, five patches) and eight links to build the SOFM.

Note that for the standard SOM network [3], the output grid is regular, in the sense that it has a well-defined geometric structure, e.g. rectangular, cylindrical or toroidal, and the coordinates of the nodes are located at equally-spaced positions, so that the distances between neighboring coordinates are equal. In the proposed approach, the coordinates of the nodes correspond to the position of the chosen keypoints, which do not need to be equally-spaced one from another. The only constraint is that, once the coordinates of the keypoints have been chosen, the neighborhood relations between them should be preserved.

Figure 1b shows typical aspects and positions that the 'kite'-shaped template shown in Figure 1a can assume when subjected to affine transformations. This synthetic template is used in one of our experiments to represent the object to be tracked. In the next section, we summarize the theory of the evolutionary self-organizing neural network model used by the proposed object tracking algorithm.

2.2 The Evolutionary Self-Organizing Map (EvSOM)

EvSOM is a recently-proposed evolutionary algorithm for topologically ordered map formation [4]. For its description, we use the following notation: N is the number of neurons, P is the input space dimension, L is the number of data samples, $\mathbf{w}_j \in \mathbb{R}^P$, $j = 1, ..., N$, is the weight vector of the j-neuron in a fixed output array, $\mathbf{w}_i$ is the weight vector of the winning neuron and $\mathbf{x}_l \in \mathbb{R}^P$ is the l-th input vector.

The central idea of EvSOM is the optimization of a fitness function comprised of the linear combination of the Quantization Error (QE) and the Pearson Correlation Coefficient (PCC):

$$Fitness(\widetilde{\mathbf{W}}) = \alpha \cdot PCC(\widetilde{\mathbf{W}}) - \beta \cdot QE(\widetilde{\mathbf{W}}), \tag{4}$$

where $\widetilde{\mathbf{W}} = \{\mathbf{w}_1, ..., \mathbf{w}_N\}$ denotes the whole set of weight vectors, and the parameters $\alpha, \beta \in [0, 10]$ weigh the relative importance of the indices with respect to each other. The QE index assess how good is the map in providing a compact representation of the original data set. Mathematically, QE is defined as

$$QE(\widetilde{\mathbf{W}}) = \frac{1}{L} \sum_{l=1}^{L} \left\| \mathbf{x}_l - \mathbf{w}_{i(\mathbf{x}_l)} \right\|, \tag{5}$$

where $i(\mathbf{x}_l)$ is the winning neuron for pattern vector $\mathbf{x}_l$, being determined as

$$i(\mathbf{x}_l) = \arg \min_{\forall j} \{ \| \mathbf{x}_l - \mathbf{w}_j \| \} \tag{6}$$

where $\| \cdot \|$ denotes the euclidean norm.

Since $(\mathbf{r}_m, \mathbf{r}_n)$ are the coordinates of pairs of nodes in the output grid and $(\mathbf{w}_m, \mathbf{w}_n)$ are the corresponding pairs of weight vectors, the PCC index is the cross-correlation of the distances between nodes in the output space, $d(\mathbf{r}_m, \mathbf{r}_n)$, and the distances between weight vectors in the input space, $d(\mathbf{w}_m, \mathbf{w}_n)$. Mathematically, we have

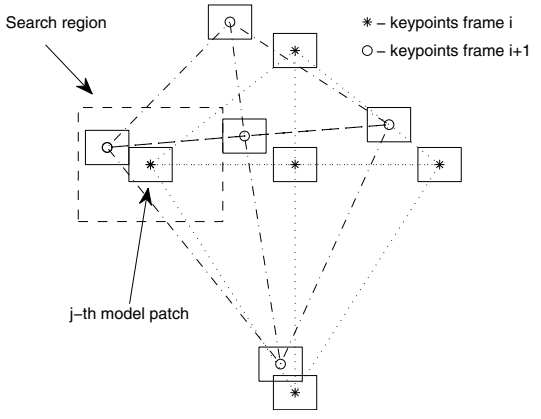


Fig. 2. Region search for candidate patches in the vicinity of a model patch

$$PCC(\widetilde{\mathbf{W}}) = \frac{\sum_{m=1}^{N}\sum_{n=1}^{N} d(\mathbf{r}_m, \mathbf{r}_n) d(\mathbf{w}_m, \mathbf{w}_n)}{(N-1)\, S_r S_w}, \tag{7}$$

where S_r and S_w are respectively the standard deviations of $\{d(\mathbf{r}_m, \mathbf{r}_n)\}$ and $\{d(\mathbf{w}_m, \mathbf{w}_n)\}$, $m, n = 1, ..., N$.

The larger the value of PCC, the higher the correlation between distances in the output space, $d(\mathbf{r}_m, \mathbf{r}_n)$, and distances in the input space, $d(\mathbf{w}_m, \mathbf{w}_n)$. The smaller the value of QE, the better the quantization of the input space. By reversing the sign of the QE index, this multi-objective optimization problem reduces to the maximization of the single index PCC.

The evolutionary algorithm comprises the following conventional steps [5]. In the next section a variant of the EvSOM algorithm is developed and applied to the tracking problem.

2.3 Object Location and Representation Updating

We introduce an object tracking algorithm that uses a variant of the EvSOM algorithm to update the object template. A parameter vector is associated with each node of the output grid which projects the corresponding model patch in the current frame i onto the target frame $i+1$. A possible strategy for evolutionary search for a solution to the problem of tracking is to search for the best fitness in some range of values within the parameter vector space $\mathbf{p}$.

The input to each tracking stage is the updated template resulting from the previous stage. This template indeed defines the EvSOM topology, i.e. the keypoints correspond to the coordinates $\mathbf{r}_j$, $j = 1, ..., N$, of the nodes comprising the EvSOM output grid (see Subsection 2.2). The weight vector $\mathbf{w}_j = \mathbf{p}_j = \left(b_x^{(j)}, b_y^{(j)}, \theta^{(j)}, s^{(j)}\right)$ represents the parameters of an affine transformation that projects the coordinates of the j-th keypoint in the current frame onto

the next frame. The coordinates of the projected keypoints correspond to the new coordinates of the nodes comprising the output grid of the EvSOM for the next frame.

At the first stage, the algorithm selects a set of candidate patches at each keypoint. These candidate patches are randomly searched in the current frame (i.e. frame i) in the vicinity of the j-th model patch of the frame i (see Figure 2). Thus, for a template with N keypoints, the result of the search process is N sets of candidate patches. It is worth pointing out that the first stage of the proposed algorithm aims at transforming the search space into a discrete set of candidate solutions. For a 'candidate solution' we mean a set of new node coordinates for the EvSOM, which is equivalent to new positions for the template keypoints.

The maximum number of candidate patches per keypoint is a prespecified value. Additionally, each candidate patch and the corresponding model patch must satisfy a measure of matching whose value must be smaller than a given threshold λ_{th}. Just as an example, assuming that the number of candidate patches per keypoint is M (all of them satisfying the required measure of matching), then for N keypoints there are M^N potential solutions.

At the second stage, the proposed procedure for dealing with the joint task of locating the object and updating its representation consists in evolving one EvSOM per frame. For the initial frame (i.e. frame 0), the keypoints of the manually selected initial template defines the coordinates of the nodes of the EvSOM for the frame 0. From frame 1 onwards, we initialize the coordinates of the nodes of the EvSOM for the frame $i+1$ with the coordinates of the nodes of the EvSOM for the frame i. At each frame, the complete set of candidate patches defines a discrete search space within which the best solution is searched for.

By evolving the EvSOM for the frame i we mean finding, using an evolutionary algorithm, the optimum weight vector of the j-th node that encodes the parameters of the affine transformation that maps the coordinates of that node from frame i to frame $i+1$. It is worth mentioning that learning the mappings from the keypoints of frame i to the keypoints of frame $i+1$ is equivalent to locating (or tracking) the moving object.

In order to evaluate the degree of similarity between regions in two images, a measure of matching between the reference patch and a candidate patch is required by the fitness function. In this paper, we use a SSD-related index, defined as

$$SSD = \sum_{k} \sum_{j} \left[T_{i+1}(k,j) - T_i(k,j) \right]^2 , \tag{8}$$

where $T_{i+1}(k,j)$ and $T_i(k,j)$ are, respectively, the intensities of gray levels in the target and current templates. By introducing the SSD index into the EvSOM fitness function, we get

$$Fitness(\widetilde{\mathbf{W}}) = \alpha \cdot PCC(\widetilde{\mathbf{W}}) - \beta \cdot SSD(\widetilde{\mathbf{W}}). \tag{9}$$

In a sum, the goal of the EvSOM-based tracking algorithm is to determine iteratively the candidate template which represents better the evolution of the object from the current frame to the target frame. The mapping is determined by

jointly optimizing the SSD index and the PCC index. In this paper, the PCC index is a measure of correlation for the distances among neighborhood interest points of the two images. The pseudo-code of the proposed EvSOM-based tracking algorithm is given below. In this algorithm the parameters FIT_{best}, FIT_{max} and G_{max} denote, respectively, the best fitness value for the current generation, the maximum fitness value found until the current generation and the maximum number of generations.

Algorithm 1. EvSOM-based tracking algorithm

1: Set $i = 0$. Then, manually extract a template with N keypoints and L links. This is the updated template for the frame 0.
2: **for all** frame i + 1 **do**
3: Set the number of EvSOM nodes equal to the number of keypoints of the updated template in frame i, with the coordinates of the keypoints assigned as the coordinates $\mathbf{r}_j$ of the nodes in the output grid, following the topological constraints established by the distance links. Then, set the EvSOM weight vectors to $\mathbf{w}_j = \mathbf{p}_j = (0, 0, 0, 1)$, $j = 1, 2, ..., N$;
4: Perform a random search in the neighborhood of the j-th node, in order to find a set $\mathcal{C}_j$ containing at most M_j candidate patches which must satisfy $SSD_j \geq \lambda_{th}$, for $j = 1, ..., N$. The neighborhood of the j-th node is defined as $\Delta\mathbf{p}_j = (\Delta b_x^{(j)}, \Delta b_y^{(j)}, \Delta\theta^{(j)}, \Delta s^{(j)})$. For the k-th candidate patch associated with the j-th node, store its transformation vector $\mathbf{p}_k^{(j)}$ and its $SSD_k^{(j)}$ value, for $k = 1, ..., M_j$ and $j = 1, ..., N$.
5: Build the initial population of candidate templates by randomly taking a candidate patch from each set $\mathcal{C}_j$, $j = 1, ..., N$, and assessing its fitness.
6: **while** $FIT_{best} \leq FIT_{max}$ and $generation \leq G_{max}$ **do**
7: Generate the offspring and compute the fitness values;
8: Build the next population and assess its fitness;
9: **end while**
10: To avoid template drift, update each $\mathbf{p}_j$, $j = 1, ..., N$, by solving Eq. (11) through the hill-climbing algorithm.
11: Compute the transformations $w(\mathbf{r}_j, \mathbf{p}_j) = w\left(\mathbf{r}_j, b_x^{(j)}, b_y^{(j)}, \theta^{(j)}, s^{(j)}\right)$, compute the resulting RMSE and present the solution.
12: **end for**

A major feature of the proposed algorithm is that it takes into account in a very natural, inherent way the topological constraints of the template used for tracking the object of interest. This is not easily done by traditional tracking methods, as mentioned in the introduction. It becomes natural for the proposed algorithm because it is based on a topology-preserving self-organizing neural network. These topological constraints are taken into account via the PCC index included in the fitness function shown in Eq. (9). If such topological constraints are not present, the algorithm does not work suitably. This can be easily verified if one removes the PCC index from the fitness function and tries to optimize the SSD index solely.

As we emphasized in the 3rd paragraph of the Section 2.2, the grid of nodes comprising the EvSOM-based tracking algorithm is a non-regular one, unlike

e.g. the regular grid of Kohonen map, since the distances from one node to the others need not to be equal. However, the links between the nodes do define their neighborhood (i.e. topological) relationships, and must be preserved, i.e. maintained as the tracking proceeds.

In order to diminish the template drift effect due to the accumulated error in each stage, we solve Eq. (1), for each $\mathbf{p}_j$, through the hill-climbing search with a fixed number of iterations N_{hc}. The resulting template is then considered as the updated template. Finally, since the components of the weight vector $\mathbf{w}_j = \mathbf{p}_j$, $j = 1, ..., N$, are real numbers, for the coordinates of the keypoints in the image to assume only integer values, we have to quantize and interpolate the values of the coordinates of the projected keypoints to the closest integer values.

3 Results and Discussion

In order to evaluate the performance of the proposed EvSOM-based approach in object tracking we have carried out experiments with three video sequences (one artificially generated film and two real-world films). Due to lack of space, we report only results on the experiment with one of the real-world films[1]. We compare the performance of the proposed approach with the direct tracking method described in [7].

For all the experiments to be described in the next sections, the fitness function parameters were set to $\alpha = \beta = 1$, since the PCC and SSD indexes are rescaled to the range $[0, 1]$. For the computation of RMSE values, the ground-truth for reference trajectories of the object of interest were manually established. All the reported experiments were developed in Matlab, version R2009a, running under Microsoft Windows Vista, in a desktop PC with an Intel Core 2 Duo processor, clock of 1.8GHz and 4GB RAM.

We used a real-world film with 362 frames, which is publicly available[2]. Each frame in this film has 512×512 pixels. The parameters of the proposed algorithm are the following: $\Delta\mathbf{p} = ([-15,\ 15],\ [-10,\ 10],\ [-5,\ 5],\ [0.98,\ 1.02])$, $N = 5$, $M = 32$ and $\lambda_{th} = 0.20$. Model and candidate patches are of size 21×21 pixels for all the experiments reported in this section.

Figure 3 shows a sequence of four frames of the film, within which a rectangle delimits the region corresponding to the decision of the algorithm on the location of the object of interest (a cork). The initial template for the used film is shown in the frame 000 (upper row). It is worth emphasizing that the object of interest experiences changes in the illumination level along its motion. In spite of these changes in illumination, the EvSOM-based algorithm is able to track the object successfully.

Figure 4 shows the performances of the proposed approach in RMSE values compared to the performance of a direct tracking method. One can note that the proposed approach outperformed the direct method in average. The proposed

[1] For the interested reader, the other two video sequences and corresponding results are available upon request.

[2] website: http://esm.gforge.inria.fr/ESMdownloads.html

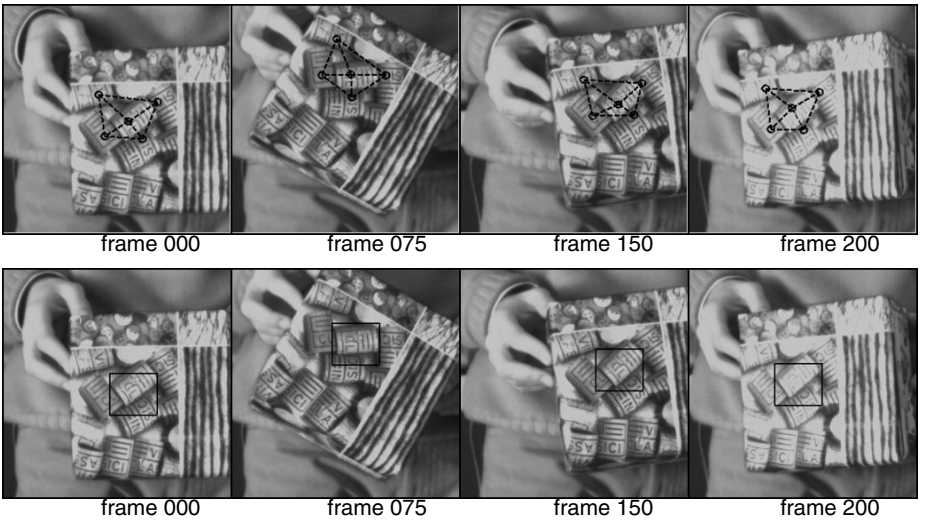


Fig. 3. Sequence of 4 frames showing the object of interest being tracked. Upper row: changes in the associated template. Lower row: estimated object's positions.

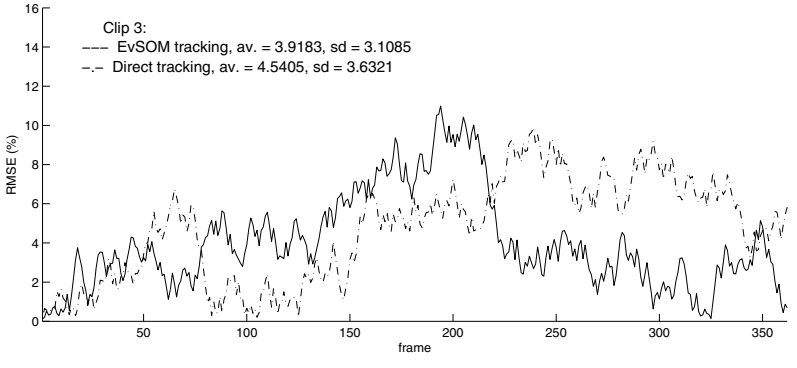


Fig. 4. Evolution of the RMSE values between true and estimated keypoints for the real-world clip used in the object tracking experiment.

method achieved an average RMSE value of 3.92 with standard deviation of 3.11, while the direct method achieved an average RMSE value of 4.54 with a standard deviation of 3.63. A conclusion that can be draw from these results is that structural information, present in the EvSOM-based approach but not in the direct tracking method, indeed improves the tracking performance.

The processing time for one run of the proposed algorithm, implemented in a non-optimized Matlab code, is 46ms (excluding the loading time of the images), which is quite remarkable. This can be explained by the fact that, at each run of the algorithm, the initial solution for the template position is already close to the final solution since the frame rate is much higher than the object speed. With a suitable code optimization those processing times can be diminished further.

4 Conclusions

An extension of the EvSOM [4] was developed and applied to object tracking. The main characteristic of the proposed approach is the inclusion of geometric or topological constraints in the determination of parameters of affine transformations that map template keypoints from one frame to the next one. Simulation results using a real-world film have shown that the proposed approach consistently outperformed a direct tracking method.

References

1. Dawoud, A., Alam, M., Bal, A., Loo, C.: Target tracking in infrared imagery using weighted composite reference function-based decision fusion. IEEE Transactions on Image Processing 15(2), 404–410 (2006)
2. Greiffenhagen, M., Comaniciu, D., Niemann, H., Ramesh, V.: Design, analysis, and engineering of video monitoring systems: An approach and a case study. Proceedings of the IEEE 89(10), 1498–1517 (2001)
3. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, Heidelberg (2001)
4. Maia, J.E.B., Barreto, G., Coelho, A.: On self-organizing feature map (SOFM) formation by direct optimization through a genetic algorithm. In: Proceedings of the 8th International Conference on Hybrid Intelligent Systems (HIS 2008), pp. 661–666 (2008)
5. Mitchell, M.: An Introduction to Genetic Algorithms, 1st edn. MIT Press, Cambridge (1998)
6. Pentland, A.: Looking at people: Sensing for ubiquitous and wearable computing. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(1), 107–119 (2000)
7. Silveira, G., Malis, E.: Unified direct visual tracking of rigid and deformable surfaces under generic illumination changes in grayscale and color images. International Journal of Computer Vision 89(51), 84–105 (2010)
8. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computing Surveys 38(114), 13–44 (2006)
9. Jia, Z., Balasuriya, A., Challa, S.: fusion-based visual target tracking for autonomous vehicles with the out-of-sequence measurements solution. Robotics and Autonomous Systems 56(2), 157–176 (2008)

Topographic Measure Based on External Criteria for Self-Organizing Map

Ken-ichi Fukui and Masayuki Numao

The Institute of Scientific and Industrial Research (ISIR), Osaka University,
8-1 Mihogaoka, Ibaraki, Osaka, 567-0047, Japan
http://www.ai.sanken.osaka-u.ac.jp

Abstract. We proposed the methodology of introducing topographic component to conventional clustering measures for the evaluation of the SOM using external criteria, i.e., class information. The topographic measure evaluates clustering accuracy together with topographic connectivity of class distribution on the topology space of the SOM. The topographic component is introduced by marginalization of basic statistics to the set-based measures, and by a likelihood function to the pairwise-based measures. Our method can extend any clustering measure based on set or pairwise of data points. The present paper examined the topographic component of the extended measure and revealed an appropriate neighborhood radius of the topographic measures.

Keywords: clustering measure, topology, neighborhood function.

1 Introduction

Self-Organizing Map (SOM)[5] has a capability to capture data distribution within the feature space and simultaneously map it into a low-dimensional representation. The SOM has been applied as a visual data mining tool in various fields such as support for exploratory analysis for a vast amount of documents[7,1], economics and medical data, and monitoring and failure diagnosis of industrial instruments[2] and among others[8].

However, the fundamental issue when using SOM as a visual data mining tool is evaluation of effectiveness of the visualization result. The most of researches evaluate the result on the basis of domain knowledge by users or domain experts. This kind of subjective evaluation cannot evaluate how much accurately desired visualization is formed on the map.

Here, the major properties of SOM are: **1. Quantization** by prototype vectors which can be considered as centroids of micro clusters and **2. Topology preservation** of neighbor connectivity of data points within the feature space. There exists internal/external quantitative criteria for the evaluation of the SOM learning result:

Internal criteria. The measures that use internal criteria evaluate how much data distribution is caught accurately in the feature space. As for such measures, quantization error and various topographic errors[4,3,9] are proposed.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 131–140, 2011.

However, the internal criteria cannot evaluate category/class distribution on the visualized map, i.e., user's perspective.

External criteria. The measures that use external criteria evaluate how accurately the correct/desired clusters are formed onto the map. There are various clustering measures based on class label, but the conventional clustering measures do not consider topology among the clusters. Very few works introduce topographical evaluation into a clustering measure, e.g., entropy-based measure[6].

Against above problem, we have proposed the methodology introducing into conventional clustering measures topographic connectivity of class distribution on the topology space of the SOM by utilizing a neighborhood function[2]. The extended measure can evaluate clustering accuracy and simultaneously topographic connectivity. The entropy-based measure proposed in [6] is limited to entropy, whereas our method can apply any conventional clustering measures. This paper studied the topographic component of the extended measure and revealed the existence of an appropriate neighborhood radius.

2 Topographic Measure Based on External Criteria

There are two types of clustering measures, namely set-based and pairwise-based measures. These two types of measures can be extended in a different manner.

2.1 Extension of Set-Based Clustering Measures

First, the way to extend set-based clustering measures[10] such as cluster purity, class F-measure[1] and entropy is described in this section.

Fig. 1 shows a learned one-dimensional SOM as an example. The neighbor data with class label are assumed to be assigned to each neuron node by the winner, i.e., the best matching unit. Here, a set of neighbor data corresponds to a micro cluster. In addition, topology of these micro clusters are assumed to be obtained by the SOM learning.

In general, it is better that samples of the same class are in neighbor and that the different classes are in distant on the map. The measure should evaluate this property. By considering topology of micro clusters, neighbor class distribution should be taken into account to the degree of certain class contained in a micro cluster. That is, data points of the same class in the neighbor clusters should be high weight, whereas that of the distant clusters should be low weight.

Based on above concept, let $N_{t,i}$ be the number of samples with class $t \in T$ in the i^{th} cluster $C_i \in C$. N_i denotes the number of samples in cluster C_i. Also N denotes the total number of samples. These basic statistics $N_{t,i}$, N_i, and N are weighted by neighbor clusters as follows:

[1] Normally, a set-based F-measure is simply called F-measure, but in this paper we call it class F-measure in order to distinguish from pairwise F-measure.

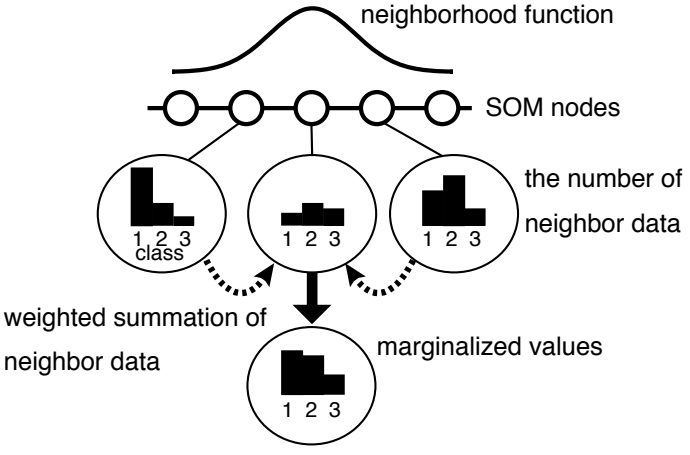


Fig. 1. Extension of a set-based clustering measure on the topology space of the SOM. The basic statistics are weighted by the neighborhood function.

$$N'_{t,i} = \sum_j h_{i,j} N_{t,j}, \tag{1}$$

$$N'_i = \sum_t N'_{t,i} = \sum_t \sum_j h_{i,j} N_{t,j}, \tag{2}$$

$$N' = \sum_i N'_i = \sum_i \sum_t \sum_j h_{i,j} N_{t,j}. \tag{3}$$

$\{N'_{t,i} | \ \forall t, i\}$ is a neighbor class distribution that is calculated by a weighted summation of the original class distribution over the topology space, $\{N'_i | \ \forall i\}$ is a neighbor data distribution that is given by a summation of the neighbor class distribution over classes, and N' is a total volume of neighbor data that is given by a summation of the neighbor data distribution over all the micro clusters. The neighborhood function $h_{i,j}$ is used as marginalization weights in the same manner as in the learning phase, but to introduce topographic connectivity of class distribution over the topology space as shown in Fig. 1. Any monotonically decreasing function is available, e.g., typically the Gaussian function:

$$h_{i,j} = \exp\Big(- \frac{||\boldsymbol{r}_i - \boldsymbol{r}_j||}{\sigma^2} \Big), \tag{4}$$

where $\boldsymbol{r}$ is a coordinate of a neuron within the topology space, and $\sigma (> 0)$ is a marginalization (neighborhood) radius. Note that the size of marginalization radius is not nesessary to use the same neighborhood radius in the SOM learning.

Then, topographic cluster purity, class F-measure, and entropy are defined by using the marginalized statistics of eq. (1), (2), and (3) as follows:

topographic Cluster Purity (tCP)

$$\mathrm{tCP}(C) = \frac{1}{N'} \sum_{C_i \in C} \max_{t \in T} N'_{t,i}. \tag{5}$$

The original purity is an average of the ratio that a majority class occupies in each cluster, whereas in the topographic purity a majority class is determined by neighbor class distribution $\{N'_{t,i}\}$.

topographic Class F-measure (tCF)

$$\mathrm{tCF}(C) = \sum_{t \in T} \frac{N_t}{N} \max_{C_i \in C} F(t, C_i), \tag{6}$$

$$\mathrm{F}(t, C_i) = \frac{2 \cdot Prec(t, C_i) \cdot Rec(t, C_i)}{Prec(t, C_i) + Rec(t, C_i)}, \tag{7}$$

where $Prec(t, C_i) = N'_{t,i}/N'_i$ and $Rec(t, C_i) = N'_{t,i}/N'_t$. The original F-measure is a harmonic average of precision and recall among class sets and cluster sets. The extended precision indicates a separation degree of different classes in a cluster and its neighbors, and the extended recall indicates density of the same class over topology space.

topographic Entropy (tEP)

$$\mathrm{tEP}(C) = \frac{1}{|C|} \sum_{C_i \in C} Entropy(C_i), \tag{8}$$

$$Entropy(C_i) = -\frac{1}{\log N'} \sum_{t \in T} \frac{N'_{t,i}}{N'_i} \log \frac{N'_{t,i}}{N'_i}. \tag{9}$$

The original entropy indicates the degree of unevenness of class distribution within a cluster, whereas the extended entropy includes unevenness of neighbor clusters.

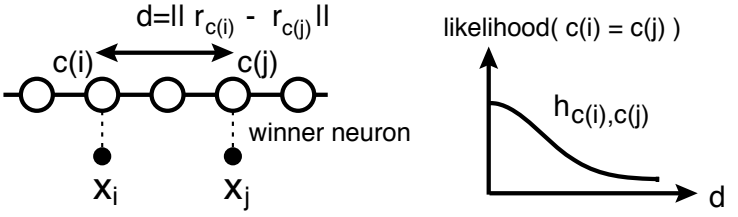
2.2 Extension of Pairwise-Based Clustering Measures

Second, this section describes an extension of pairwise-based clustering measures [11]. Table 1 shows a class/cluster cross table of data pairs, where $t(i)$ denotes the class of data point $\boldsymbol{x}_i$, $c(i)$ denotes the micro cluster that is the best matching unit of $\boldsymbol{x}_i$, and a, b, c, d are the number of data pairs where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ do or do not belong to the same class/cluster.

Here, we introduce $likelihood(c(i) = c(j))$ indicating the degree that a data pair $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ belongs to the same cluster instead of the actual number of data pairs. The likelihood is given by topological distance of the winner neurons of the data pair as illustrated in Fig. 2(a). The same neighborhood function by

Table 1. Cross table of pairwise classification

	$t(i) = t(j)$	$t(i) \neq t(j)$
$c(i) = c(j)$	a	b
$c(i) \neq c(j)$	c	d



(a) Distance of data pair on a topology space (b) Likelihood function

Fig. 2. Extension of a pairwise-based clustering measure. A likelihood function is introduced to represent the degree that the data pair belongs to the same micro cluster.

eq. (4) is available for the likelihood function (Fig. 2(b)). Then, a, b, c and d are replaced by summation of the likelihoods as follows:

$$a' = \sum_{\{i,j|t(i)=t(j)\}} h_{c(i),c(j)}, \tag{10}$$

$$b' = \sum_{\{i,j|t(i)\neq t(j)\}} h_{c(i),c(j)}, \tag{11}$$

$$\begin{aligned} c' &= \sum_{\{i,j|t(i)=t(j)\}} \left(1 - h_{c(i),c(j)}\right) \\ &= a + c - a', \end{aligned} \tag{12}$$

$$\begin{aligned} d' &= \sum_{\{i,j|t(i)\neq t(j)\}} \left(1 - h_{c(i),c(j)}\right) \\ &= b + d - b'. \end{aligned} \tag{13}$$

With these extended a', b', c' and d', the topographic pairwise accuracy and pairwise F-measure are defined as follows:

topographic Pairwise Accuracy (tPA)

$$\mathrm{tPA}(C) = \frac{a' + d'}{a' + b' + c' + d'}. \tag{14}$$

The original pairwise accuracy is a ratio of the number of pairs that the same class belong to the same cluster or different class belong to the different cluster to all of the pairs. Whereas, the topographic PA is a degree that the same class belong to the neighbor cluster or that the different class belong to the distant cluster.

topographic Pairwise F-measure (tPF)

$$\mathrm{tPF}(C) = \frac{2 \cdot P \cdot R}{P + R}, \tag{15}$$

where $P = a'/(a' + b')$ is a precision that is a ratio of the same class among each cluster, and $R = a'/(a' + c')$ is a recall that is a ratio of the same cluster among each class. The original pairwise F-measure is a harmonic average of the precision and the recall. Whereas, the topographic PF is based on a degree that the data pairs belong to the same cluster.

2.3 Neighborhood Function

For the neighborhood function for the marginalization and the likelihood, any monotonically decreasing function $h_{i,j} \geq 0$ is available such as a Gaussian or a rectangle function same as in the learning of the SOM. Note that the extended measures are exactly the same as the original measures when $h_{i,j} = \delta_{i,j}$ (δ is the Kronecker delta). As for shape of class distribution, our measure has no assumption in the original feature space, but has assumption in the topology space by the shape of the neighborhood function.

And, the neighborhood radius affect to the degree of marginalization and likelihood. Fig. 3 illustrates that the extended measure evaluates individual clusters, that is the original values, as the marginalization radius becomes zero. On the contrary, as the radius becomes larger, the finite topology space is smoothed by almost the same weights, all micro clusters are treated as one big cluster. The optimal radius depend on class distribution and function of the evaluation measure. The way to find the optimal radius is described in the next section.

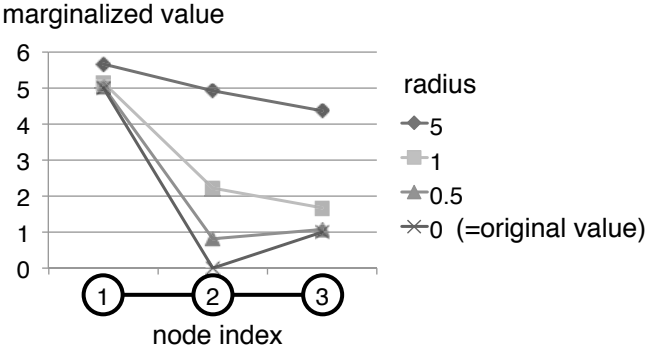
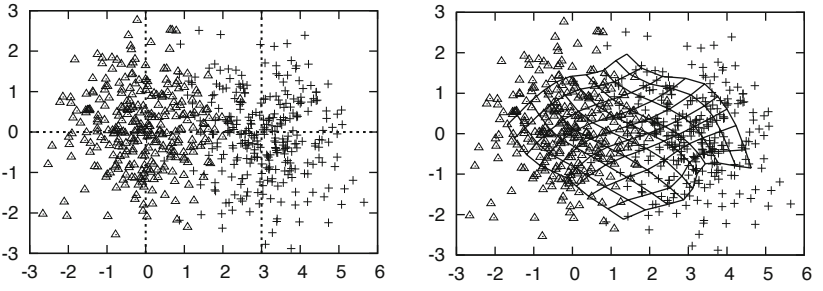


Fig. 3. Example of effect of the marginalization. The larger radius is the smoother the values over connectivity of the nodes.

3 Properties of the Topographic Measures

3.1 Datasets

For the evaluation of property of the proposed topographic measure, we prepared two classes of two dimensional synthetic data, where 300 data points for each class were generated from different Gaussian distributions (Fig.4(a)).



(a) Data distribution (b) Captured by SOM

Fig. 4. Two dimensional synthetic data. The data points were generated from two Gaussian distributions; $N(\mu_1, 1)$ and $N(\mu_2, 1)$, where $\mu_1 = (0, 0)$ and $\mu_2 = (3, 0)$.

Also well-known open datasets[2] were used as real-world data: Iris data (150 samples, 4 attributes, 3 classes), Wine data (178 samples, 13 attributes, 3 classes), and Glass Identification data (214 samples, 9 attributes, 6 classes).

3.2 Experimental Condition

We employed the batch type SOM in which Gaussian function was used as a neighborhood function together with decreasing strategy of the neighbor radius. The neurons was set to 10×10 regular grid of the most standard setup. Also the Gaussian function by eq. (4) was employed for the neighborhood function of the topographic measures. Then, the evaluation values of each measure were averaged over 100 runs to avoid dependency of initial random values of the prototypes.

3.3 Topographic Component

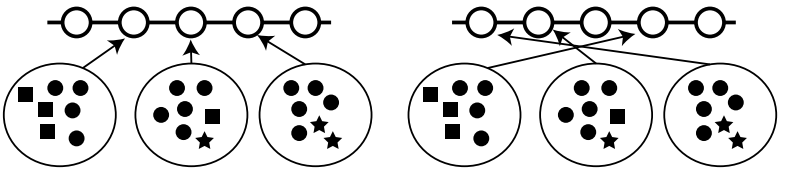
In order to evaluate the topographic component of the measure, we prepared a SOM with random topology that is generated after the learning of SOM by exchanging assignment of set of neighbor data randomly, while preserving elements of neighbor data (Fig. 5). This procedure destroys the topographic connectivity, while preserving micro clusters.

Assuming the topology of data distribution is preserved in some degree by the SOM learning, the topographic component is defined by the difference between the evaluation values for the standard SOM with topology preservation (Fig. 5(a)) and for the SOM with random topology (Fig. 5(b)). We assume the neighborhood radius that maximizes the topology component is the appropriate.

3.4 Effect of Neighborhood Radius

Synthetic Data. Fig. 6 shows the evaluation values of the topographic measures for the synthetic data. The larger value is the better except entropy.

[2] http://archive.ics.uci.edu/ml/



(a) after the SOM learning (b) SOM with random topology

Fig. 5. The way to calculate the topogoraphic component. The component is calculated by difference of the values between (a) topology preservation and (b) random topology.

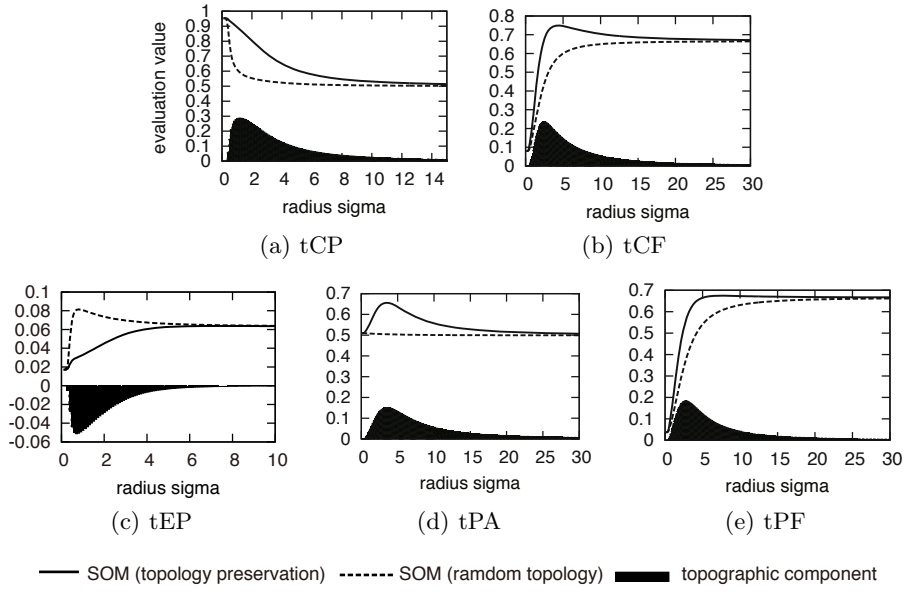


Fig. 6. The evaluation values of the topographic measures for the synthetic data with changing the neighborhood radius

Firstly, the standard SOM with topology preservation provides always better value than SOM with random topology where topographic connectivity is destroyed. This means that the proposed topographic measures evaluate both clustering accuracy and topographic connectivity.

Secondly, as the neighborhood radius becomes close to zero, the extended measure evaluates individual micro clusters without topographic connectivity. Whereas, as the radius becomes larger, the extended measure treats whole data as one big cluster as mentioned before. Therefore, the solid and the broken lines gradually become equal as the radius becomes close to zero or becomes much larger.

Thirdly, the topographic component has a monomodality against the radius in all measures, since there exists an appropriate radius to the average class distribution. Since the topographic measure is a composition of clustering accuracy

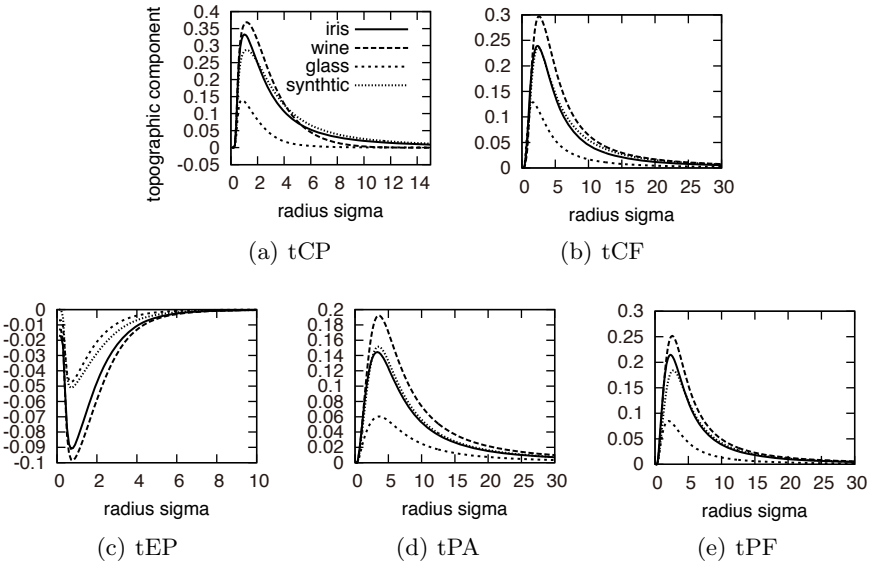


Fig. 7. Topographic components of the topographic measures for real-world data

and topographic connectivity, the radius that gives the maximum value to SOM (topology preservation) does not always match with the maximum value of topographic component. Therefore, the topographic component should be examined to find the appropriate radius. Also the appropriate radius depends on function of the measure such as purity, F-measure, or entropy. This means that the user should use different radius for each measure.

Real-World Data. Also for real-world data, there exists an appropriate radius (Fig. 7). However, the appropriate radii depend on the number of classes and the class distribution, not only depend on the function of a measure. This result indicates that depending on a measure and a dataset, a user should use different radius that gives the maximum volume of topographic component.

4 Conclusion

We proposed the topographic measures using external criteria (class information) for the evaluation of the SOM as a visual data mining tool. Our method introduces the topographic component to set-based and pairwise-based clustering measures by utilizing the neighborhood function. Since our method extends the basic statistics, any set-based or pairwise-based clustering measures can also be extended. Then, the present paper clarified the properties of the topographic measures using synthetic and real-world data. The experiments revealed the existence of an appropriate neighborhood radius. A user should use an appropriate radius depending on a measure and a dataset. As for future works, we should develop an easier way to find an appropriate radius, also should examine a usage of the proposed measure when comparing different result of SOMs.

Acknowledgements

This work was supported by KAKENHI (21700165).

References

1. Fukui, K., Saito, K., Kimura, M., Numao, M.: Sequence-based som: Visualizing transition of dynamic clusters. In: Proc. of IEEE 8th International Conference on Computer and Information Technology (CIT 2008), pp. 47–52 (2008)
2. Fukui, K., Sato, K., Mizusaki, J., Numao, M.: Kullback-leibler divergence based kernel SOM for visualization of damage process on fuel cells. In: Proc. of 22nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2010)., vol. 1, pp. 233–240 (2010)
3. Goodhill, G.J., Sejnowski, T.J.: Quantifying neighbourhood preservation in topographic mappings. In: Proc. of the 3rd Joint Symposium on Neural Computation, vol. 6, pp. 61–82 (1996)
4. Kiviluoto, K.: Topology preservation in self-organizing maps. In: Proc. of International Conference on Neural Networks (ICNN 1996), pp. 294–299 (1996)
5. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (1995)
6. Koskela, M., Laaksonen, J., Oja, E.: Entropy-based measures for clustering and SOM topology preservation applied to content-based image indexing and retrieval. In: Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004), vol. 2, pp. 1005–1009 (2004)
7. Lagus, K., Kaski, S., Kohonen, T.: Mining massive document collections by the websom method. Information Sciences 163, 135–156 (2004)
8. Oja, M., Kaski, S., Kohonen, T.: Bibliography of self-organizing map (som) papers: 1998-2001 addendum. Neural Computing Surveys 3, 1–156 (2002)
9. Uriarte, E.A., Martín, F.D.: Topology preservation in SOM. International Journal of Mathematical and Computer Sciences 1(1), 19–22 (2005)
10. Veenhuis, C., Koppen, M.: Data Swarm Clustering, ch. 10, pp. 221–241. Springer, Heidelberg (2006)
11. Xu, R., Wunsch, D.: Cluster Validity. Computational Intelligence, ch. 10, pp. 263–278. IEEE Press, Los Alamitos (2008)

Influence of Learning Rates and Neighboring Functions on Self-Organizing Maps

Pavel Stefanovič and Olga Kurasova

Vilnius University, Institute of Mathematics and Informatics,
Akademijos str. 4, LT-08663, Vilnius, Lithuania
Stefanovic.Pavel@gmail.com, Olga.Kurasova@mii.vu.lt

Abstract. In the article, the influence of neighboring functions and learning rates on self-organizing maps (SOM) has been investigated. The target of a self-organizing map is data clustering and their graphical presentation. Bubble, Gaussian, and heuristic neighboring functions and four learning rates (linear, inverse-of-time, power series, and heuristics) have been analyzed here. The learning rate has been changed according to epochs and iterations. A comparative analysis has been made with three data sets: glass, wine, and zoo. The quantization error has been measured in order to estimate the SOM quality.

Keywords: self-organizing map (SOM), learning rates, neighboring functions, quantization error.

1 Introduction

The self-organizing maps (SOM) as a type of artificial neural networks are commonly used for data clustering and visualization. It is important to research SOMs, because they are applied in various areas such as ecology, military, medicine, engineering, etc. For example, in the medicine area, SOM can be useful for analyzing breast cancer data sets that can help medics to make decisions [1]. Self-organizing maps can be combined with dimensionality reduction methods as a multidimensional scaling [2], [3], [4], [5], [6]. There the number of dimensions of the neurons winners obtained by SOM is reduced to two by multidimensional scaling and presented in a plane. It is important to train SOM so that the neurons winners correspond to the data analyzed as faithfully as possible.

The results of SOM depend on some initialization and learning parameters. In this article, three neighboring functions (bubble, Gaussian and heuristic [7]) and four learning rates (linear, inverse-of-time, power series and heuristic [7]) are investigated. In the training process, learning rates can be changed in two ways (epochs and iterations). The dependence of these ways on the results is researched, too. The quality of SOM is commonly estimated by quantization and topographic errors. The main goal of the research is to estimate dependences of learning parameters on the results obtained by SOM in the sense of quantization error. Experiments have been carried out with three data sets: glass, wine, and zoo.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 141–150, 2011.

2 Self-Organizing Maps

2.1 Principles of Self-Organizing Maps

An artificial neural network is a mathematical model of the biological neuron system. A self-organizing map (SOM) is one of mostly analyzed unsupervised models of neural network. First time, it was described by Finn scientist Teuvo Kohonen in 1982. Consequently sometimes they are called Kohonen map or networks. The model takes an important place in science and it is one of the most popular research objects to date. The main target of the SOM is to preserve the topology of multidimensional data, i. e., to get a new set of data from the input data such that the new set preserved the structure (clusters, relationships, etc) of the input data. The SOM are applied to cluster (quantize) and visualize the data. The self-organizing map is a set of nodes, connected to each other via a rectangular or hexagonal topology. SOM of rectangular topology is presented in Fig. 1. Here a circle represents a node. The connections between the inputs and the nodes have weights, so a set of weights corresponds to each node. The set of weights forms a vector $M_{ij}, i = 1,...,k_x, j = 1,...,k_y$ that is usually called a neuron or a codebook vector, where k_x is the number of rows, and k_y is the number of columns. Commonly SOM is called a self-organizing neural network. The dimension of the codebook vector is the same as that of the number of inputs [9]. When we have a SOM, we can describe a concept of the neighboring rank. All neurons adjacent to a given neuron M_c can be defined as its neighbors of the first rank ($\eta_{ij}^c = 1$), then the neurons adjacent to first-order neighbor, excluding those already considered, as neighbors of a second rank ($\eta_{ij}^c = 2$), etc. Neighboring ranks of the marked (grey) node are presented in Fig. 1.

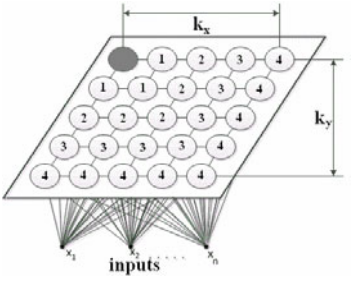


Fig. 1. Two-dimensional SOM (rectangular topology)

A self-organizing map (neural network) is learned by an unsupervised manner. The learning starts from the components of the vectors M_{ij} initialized at random or by the principal components. If *n*-dimensional vectors $X_1, X_2, ..., X_m$ are needed to map, the components of these vectors $x_1, x_2, ..., x_n$ are passed to the network as the inputs, where *m* is the number of the vectors, *n* is the number of components of the vectors.

At each learning step, an input vector $X_p \in \{X_1, X_2, ..., X_m\}$ is passed to the neural network. The vector X_p is compared with all neurons M_{ij}. Usually the Euclidean distance between this input vector X_p and each neuron M_{ij} are calculated. The vector (neuron) M_c with the minimal Euclidean distance to X_p is designated as a winner. All neurons components are adapted according to the learning rule:

$$M_{ij}(t+1) = M_{ij}(t) + h_{ij}^c(t)(X_p - M_{ij}(t)) \tag{1}$$

Here t is the number of iterations, $h_{ij}^c(t)$ is a neighboring function, c is a pair of indices of the neuron winner of vector X_p. The learning is repeated until the maximum number of iterations is reached. Here we face with two notions: training iteration and epoch. One iteration is part of the training process, when one input vector is passed to the network and the neurons are changed. An epoch is part of the training process, when all vectors of the training data set from X_1 to X_m are passed to the network once. An epoch consists of m iterations.

2.2 Neighboring Functions and Learning Rates

The neighboring function $h_{ij}^c(t)$ influences the training result. So it is important to choose a proper function [8], [9]. We can use various neighboring functions, but bubble (2) and Gaussian (3) functions are usually used.

$$h_{ij}^c = \begin{cases} \alpha(t), (i,j) \in N_c \\ 0, (i,j) \notin N_c \end{cases}, \tag{2}$$

$$h_{ij}^c = \alpha(t) \cdot e^{\left(\frac{-\|R_c - R_{ij}\|^2}{2(\eta_{ij}^c(t))^2} \right)}. \tag{3}$$

Here $\alpha(t)$ is a learning rate, and it depends on the number of iterations, N_c is the index set of neighboring nodes around the nodes with indices c [4]. The parameter η_{ij}^c is the neighboring rank of M_{ij}. The functions $\alpha(t)$ and $\eta_{ij}^c(t)$ are monotonically decreasing functions. Two-dimensional vectors R_c and R_{ij} consist indexes of M_c and M_{ij} (number of rows and columns). The indexes show a place of the neuron-winner M_c of vector X_k and the recalculated neuron M_{ij} in SOM.

The learning rate $\alpha(t)$ can be selected in various ways. As usual the functions should be decreasing. Usually, linear (4), inverse-of-time (5), and power series (6) functions are used [10].

$$\alpha(t) = \frac{1}{t} \tag{4}$$

$$\alpha(t,T) = \left(1 - \frac{t}{T}\right) \tag{5}$$

$$\alpha(t,T) = (0.005)^{\frac{t}{T}} \tag{6}$$

One more heuristic neighboring function [7] is used in the investigation. The neighboring function (7) is monotonically decreasing.

$$h_{ij}^{c} = \frac{\alpha(t)}{\alpha(t)\eta_{ij}^{c} + 1}, \tag{7}$$

$$\text{where } \alpha = \max\left(\frac{T+1-t}{T}, 0.01\right). \tag{8}$$

Here T is the number of epochs (iterations), t is the order number of a current epoch (iteration), η_{ij}^{c} is the neighboring rank between neurons M_c and M_{ij}. The neurons M_{ij} are recalculated in each epoch, if the inequality (9) is valid. This condition is applied in all the cases analyzed.

$$\eta_{ij}^{c} \le [\alpha \max(k_x k_y), 1] \tag{9}$$

Often two training phases (rough and fine tuning) are used, but in this investigation the training is not divided into two phases.

After the SOM network has been trained, its quality must be evaluated. Usually two errors (quantization and topographic) are calculated. Quantization error E_{QE} (10) shows how well neurons of the trained network adapt to the input vectors. E_{QE} is the average distance between data vectors X_p and their neuron-winners $M_{c(p)}$.

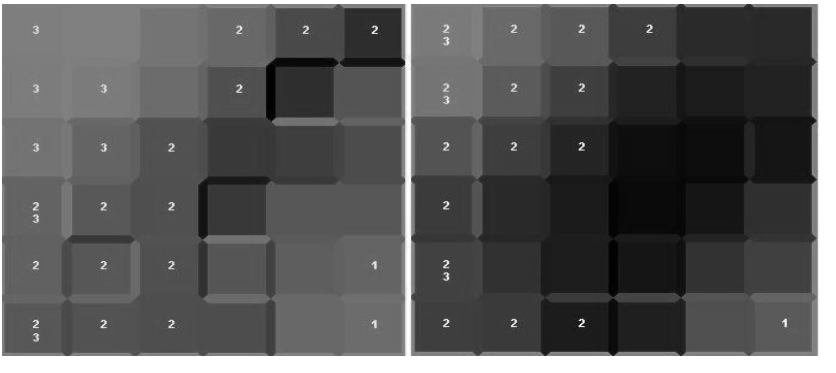


Fig. 2. Iris data set in 6x6 SOM: bubble (left) and Gaussian (right) neighboring functions are used

$$E_{QE} = \frac{1}{m} \sum_{p=1}^{m} \left\| X_p - M_{c(p)} \right\| \tag{10}$$

A way to estimate the quality of the trained SOM is to examine u-matrix representation on SOM. However it is difficult to notice a difference of two maps, trained with different parameters (Fig. 2). System "NeNet" [11] is used to create the maps. Quantitative measures allow us to estimate the quality of the SOM definitely.

3 Experimental Results

3.1 Data Set Analyzed

Three data sets are used in the experimental investigation. The glass data set was collected by a scientist, which wanted to help criminalists to recognize glass slivers found [12]. Nine-dimensional vectors $X_1, X_2, ..., X_{214}$ are formed, where $X_i = (x_{i1}, x_{i2}, ..., x_{i9})$, $i = 1, ..., 214$. Nine features are measured: x_1 is a refractive index, x_2 is sodium, x_3 is magnesium, x_4 is aluminum, x_5 is silicon, x_6 is potassium, x_7 is calcium, x_8 is barium and x_9 is iron.

The wine data set consists of the results of chemical analysis of wine grapes grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. Thirteen-dimensional vectors $X_1, X_2, ..., X_{178}$ are formed, where $X_i = (x_{i1}, x_{i2}, ..., x_{i13})$, $i = 1, ..., 178$. Thirteen features are measured: x_1 is alcohol, x_2 is malic acid, x_3 is ash, x_4 is alcalinity of ash, x_5 is magnesium, x_6 is total phenol, x_7 is flavanoid, x_8 is nonflavanoid phenol, x_9 is proanthocyanin, x_{10} is color intensity, x_{11} is hue, x_{12} is OD280/OD315 of diluted wines, x_{13} is proline.

The third data set is zoological (zoo). The data set consists of 16 boolean values. Sixteen-dimensional vectors $X_1, X_2, ..., X_{92}$ are formed, where $X_i = (x_{i1}, x_{i2}, ..., x_{i16})$, $i = 1, ..., 92$. Sixteen features are measured: x_1 is hair, x_2 is feathers, x_3 is eggs, x_4 is milk, x_5 is airborne, x_6 is aquatic, x_7 is a predator, x_8 is toothed, x_9 is a backbone, x_{10} is breathes, x_{11} is venomous, x_{12} is a fin, x_{13} is legs, x_{14} is a tail, x_{15} is domestic, and x_{16} is catsize.

3.2 Dependence of Results on the Learning Parameters

The SOM with three neighboring functions (2), (3), (7) and learning rates (4), (5), (6), (8) are implemented in Matlab. To find a tendency how much different parameters affect the SOM training results, 72 experiments are carried out (three data sets, three neighboring functions, four learning rates, and two ways of changes of learning rates

(by epochs and iterations)). The map size is set [10×10]. During the experiments, the number of epochs ranged from 10 to 100 by step 10. Each experiment is repeated 10 times with different initial values of neurons M_{ij}. The averages of quantization error (10) and their confidence intervals are calculated. In the training process, the neurons M_{ij} are recalculated, if inequality (9) is valid.

The results of the glass data set are presented in Fig. 3 and Table 1. In figures, only the numbers of epochs are presented. The number of epochs multiplied by the number of data items m corresponds to the number of iterations. For the glass data set (m=214), 10 epochs equal to 2140 iterations, 20 epochs equal 4280, etc. The worst results in the sense of quantization error are obtained, if learning rate (4) is used. In this case, the way, when the number of epochs is used in changes of the learning rate, gives better results in the sense of quantization error comparing with the results, obtained when the number of iterations is used in changes of the learning rate. When we use learning rate (5), Gaussian function (3) gives the best results, independent of the ways of changes of learning rates (by epochs or iterations). Learning rates (6) and (8) give the best results with the Gaussian function, but heuristic neighborhood function (7) with epochs also gives good results.

The confidence intervals of the averages of quantization errors are computed for each set of epochs (10, 20,…, 100). The margins of errors of the confidence intervals are averaged. Due to the limit of article length, only the averages of the margins of errors are presented in Table 1. We see that the margins of errors are small enough for all the learning rates and neighboring functions. So the averages of quantization errors, presented in Fig. 3, are informative.

Table 1. Averages of the margins of errors (confidence probability 0.95) for the glass data set. $X_i = (x_{i1}, x_{i2},...,x_{i9})$

Learning rate	By iterations			By epochs		
	Bubble	**Gaussian**	**Heuristic**	**Bubble**	**Gaussian**	**Heuristic**
(4)	0.006666	0.003382	0.005402	0.001805	0.002275	0.002394
(5)	0.001688	0.001734	0.001670	0.001457	0.001563	0.001412
(6)	0.001696	0.001531	0.001724	0.001980	0.001788	0.001830
(8)	0.001576	0.001557	0.001644	0.001516	0.001451	0.002090

The same experiments were done with the wine data set. The results are illustrated in Fig. 4. If we use learning rate (4), all the neighboring functions, when the learning rate is changed according to epochs, give the smallest quantization error. The training with learning rate (5) shows that the smallest quantization error is obtained by the Gaussian function, independent of the ways of changes of learning rates (by epochs or iterations). Just like in the case of the glass date set, the results with learning rates (6) and (8) are similar.

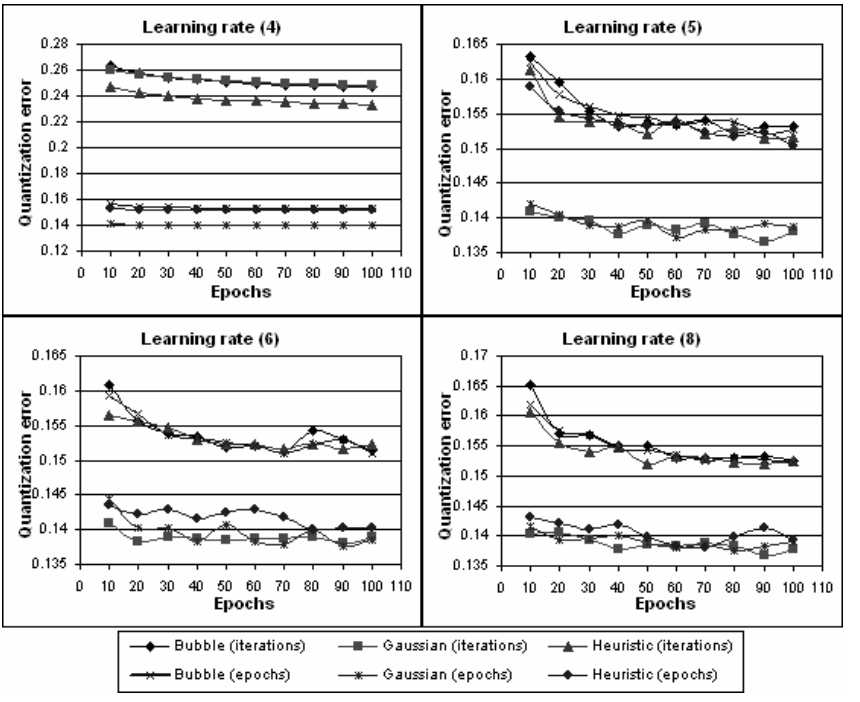


Fig. 3. The averages of quantization errors, obtained using the glass data set (learning rates, computed by formulas (4), (5), (6) and (8))

Table 2. Averages of the margins of errors (confidence probability 0.95) for the wine data set

Learning rate	By iterations			By epochs		
	Bubble	**Gaussian**	**Heuristic**	**Bubble**	**Gaussian**	**Heuristic**
(4)	0.003059	0.007191	0.002645	0.001359	0.001491	0.002169
(5)	0.001658	0.001251	0.001517	0.001736	0.001284	0.001646
(6)	0.001527	0.001333	0.001507	0.001894	0.001521	0.001428
(8)	0.001756	0.001333	0.001499	0.001919	0.001358	0.001875

The averages of the margins of errors of the confidence intervals for the wine data set are presented in Table 2. The margins of errors are small, too. We can see that the largest margins are obtained, if the Gaussian function and learning rate (4) are used, and the learning rate is changed according to iterations. The smallest margins are obtained when the Gaussian function is also used, but with learning rate (5).

The third data set analyzed is zoo. The results are presented in Fig. 5 and Table 3. If learning rate (4) is used, the same results as with other data sets are achieved (Fig. 5). All neighboring functions yield smaller quantization errors, if the learning rates are changed according to epochs. In the case of learning rate (5), the best result

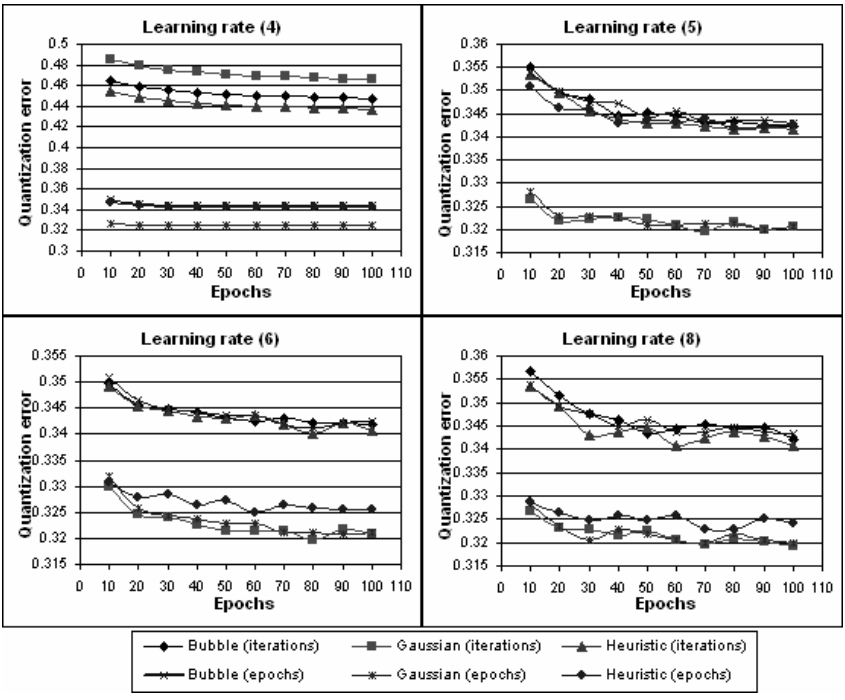


Fig. 4. The averages of quantization errors, obtained using the wine data set (learning rates computed by formulas (4), (5), (6) and (8)).

is got by Gaussian functions the same as before (according to epochs and iterations). If we use Gaussian function (iterations and epochs) and the heuristic function (epochs) with learning rate (6), we can get small quantization errors, too. In the last case (learning rate (8)), we get the best result, if the Gaussian function is used, independent of the ways of changes of learning rates.

As we see in Table 3, the margins of errors of the confidence intervals of the averages of the quantization error are small enough, too. This fact shows that the averages, computed using the observed values, differ from the mathematical expectation.

Table 3. Averages of the margins of errors (confidence probability 0.95) for the zoo data set

Learning rate	By iterations			By epochs		
	Bubble	**Gaussian**	**Heuristic**	**Bubble**	**Gaussian**	**Heuristic**
(4)	0.146629	0.039484	0.019839	0.017873	0.018802	0.021137
(5)	0.015246	0.01074	0.012951	0.018137	0.011479	0.013805
(6)	0.016417	0.013867	0.016969	0.021211	0.012662	0.014427
(8)	0.018691	0.010864	0.014641	0.016278	0.010541	0.014487

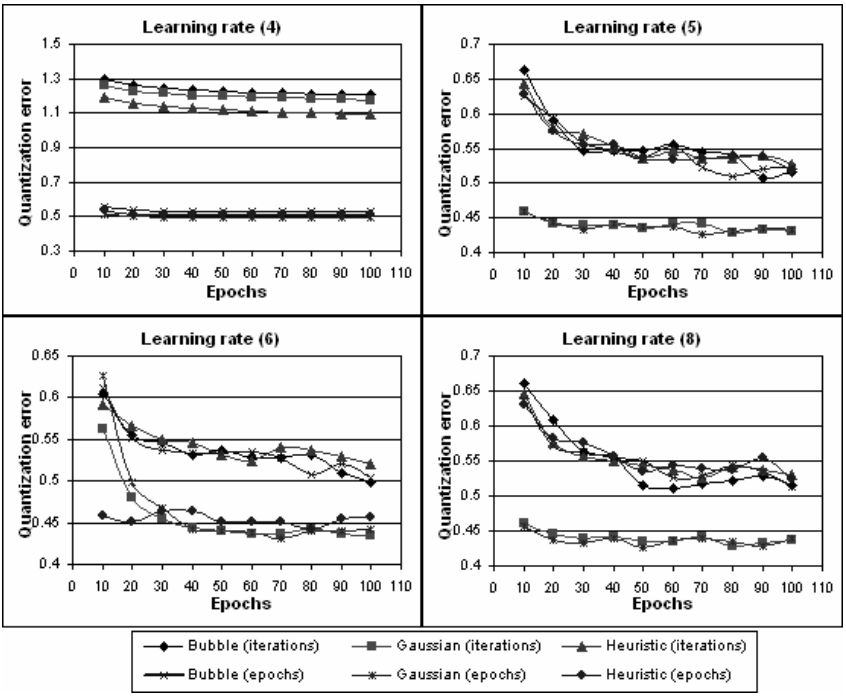


Fig. 5. The averages of quantization errors, obtained using the zoo data set (learning rates, computed by formulas (4), (5), (6) and (8))

4 Conclusions

The experimental results have showed that the smallest quantization error is obtained, if neighboring Gaussian function and nonlinear learning rates are used. The changes of learning rates according to epochs or iterations do not influence the results obtained. The bubble neighboring function yields the worst result for all the data sets analyzed. In order to get good results, the heuristic function can be helpful, too, but only if the learning rates are changed according to epochs. The smallest quantization errors are achieved with inverse-of-time, power series and heuristic learning rates. Learning rate (4) (linear) gives large quantization errors, if learning rates are changed according to iterations. It is purposeful to use another way (according to epochs) of changes of this learning rate.

The research has shown that the neighboring function, the learning rate and the way of changes of the learning rate influence the SOM results in the sense of quantization error. So it is important to choose the proper training parameters for different data sets. If we do not know which parameters to select, the best way is to choose the Gaussian function and a nonlinear learning rate.

In the future, it is purposeful to analyze how the learning parameters and neighboring functions affect the generalization capability of the SOM, i. e., how well the trained SOM describes the data not used in training.

References

1. Chen, D.R., Chang, R.F., Huang, Y.L.: Breast Cancer Diagnosis Using Self-organizing Map for Sonography. Ultrasound in Med. & Biol. 26(3), 405–411 (2000)
2. Kurasova, O., Molytė, A.: Integration of the Self-organizing Map and Neural Gas with Multidimensional Scaling. Information Technology and Control 40(1), 12–20 (2011)
3. Kurasova, O., Molytė, A.: Combination of Vector Quantization and Visualization. In: Perner, P. (ed.) MLDM 2009. LNCS (LNAI), vol. 5632, pp. 29–43. Springer, Heidelberg (2009)
4. Kurasova, O., Molytė, A.: Quality of Quantization and Visualization of Vectors Obtained by Neural Gas and Self-organizing Map. Informatica 22(1), 115–134 (2011)
5. Dzemyda, G., Kurasova, O.: Heuristic Approach for Minimizing the Projection Error in the Integrated Mapping. European Journal of Operational Research 171(3), 859–878 (2006)
6. Bernatavičienė, J., Dzemyda, G., Kurasova, O., Marcinkevičius, V.: Optimal Decisions in Combining the SOM with Nonlinear Projection Methods. European Journal of Operational Research 173(3), 729–745 (2006)
7. Dzemyda, G.: Visualization of a set of Parameters Characterized by Their Correlation Matrix. Computational Statistics and Data Analysis 36(1), 15–30 (2001)
8. Tan, H.S., George, S.E.: Investigating Learning Parameters in a Standard 2-D SOM Model to Select Good Maps and Avoid Poor Ones. In: Webb, G.I., Yu, X. (eds.) AI 2004. LNCS (LNAI), vol. 3339, pp. 425–437. Springer, Heidelberg (2004)
9. Kohonen, T.: Self-organizing Maps, 3rd edn. Springer Series in Information Sciences. Springer, Berlin (2001)
10. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: SOM Toolbox for Matlab 5 (2005), `http://www.cis.hut.fi/somtoolbox/documentation/somalg.shtml`
11. Hassinen, P., Elomaa, J., Rönkkö, J., Halme, J., Hodju, P.: Neural Networks Tool – NeNet (1999), `http://koti.mbnet.fi/~phodju/nenet/Nenet/General.html`
12. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository, University of California, School of Information and Computer, Irvine, CA (2007), `http://www.ics.uci.edu/~mlearn/MLRepository.html`

Gamma-Filter Self-Organizing Neural Networks for Time Series Analysis

Pablo A. Estévez and Rodrigo Hernández

Dept. Electrical Engineering and Advanced Mining Technology Center, University of Chile, Casilla 412-3, Santiago, Chile
pestevez@cec.uchile.cl

Abstract. In this paper, we introduce the Gamma Growing Neural Gas (γ-GNG) model for temporal sequence processing. The standard GNG is merged with a context descriptor based on a short term memory structure called Gamma memory. When using a single stage of the Gamma filter, the Merge GNG model is recovered. The γ-GNG model is compared to γ-Neural Gas, γ-SOM, and Merge Neural Gas, using the temporal quantization error as a performance measure. Simulation results on two data sets are presented: Mackey-Glass time series, and Bicup 2006 challenge time series.

1 Introduction

Several extensions of self-organizing feature maps (SOMs) [1] for dealing with processing data sequences that are temporally or spatially connected, such as words, DNA sequences, time series, etc. [2],[3],[4] have been developed. In [5] a review of recursive self-organizing network models, and their application for processing sequential and tree-structured data is presented. An early attempt to include temporal contexts in SOMs is the Temporal Kohonen Map (TKM) [6], where a neuron output depends on the current input and its context of past activities. In Recursive SOM [3],[7],the SOM algorithm is used recursively on both the current input and a copy of the map at the previous time step. In addition to a weight vector, each neuron has a context vector that represents the temporal context as the activation of the entire map in the previous time step. This kind of context is computationally expensive, since the dimension of the context vectors is equal to the number of neurons in the network.

In the Merge SOM (MSOM) [2] approach, the context is described by a linear combination of the weight and the context of the last winner neuron. This context representation is more space efficient than the one used for the Recursive SOM model, because in MSOM the dimensionality of the context is equal to the data dimensionality [4]. As the MSOM context does not depend on the lattice architecture, it has been combined with other self-organizing neural networks such as Neural Gas (NG) [8] and Growing Neural Gas (GNG) [9], yielding the Merge Neural Gas (MNG) [10] and the Merge Growing Neural Gas (MGNG) [11], respectively.

In our previous work we have added Gamma filters [12] to SOM and NG, yielding the γ-SOM [13] and γ-NG models [14], respectively. We have shown that the gamma filter variants of SOM and NG are generalizations that include as particular examples

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 151–159, 2011.

the MSOM and MNG models, when the filter order is set to one. In this paper, we add gamma filters to GNG, to produce the γ-GNG model. We compare the performance of γ-GNG with those of the γ-SOM and γ-NG models, using the temporal quantization error as a metric. Results are shown on two data sets: the Mackey-Glass time series and the Bicup 2006 time series.

2 Gamma Context Model

The Gamma filter [12,15] is defined in the time domain as

$$y(n) = \sum_{k=0}^{K} w_k c_k(n)$$

$$c_k(n) = \beta c_k(n-1) + (1-\beta) c_{k-1}(n-1) \tag{1}$$

where $c_0(n) \equiv x(n)$ is the input signal and $y(n)$ is the filter output, and $w_0, \cdots, w_K, \beta$ are the filter parameters. The $\beta \in (0,1)$ parameter provides a mechanism to decouple depth (D) and resolution (R) from filter order. Depth measures how far into the past the memory stores information, a low memory depth can hold only recent information. Resolution indicates the degree to which information concerning the individual elements of the input sequence is preserved. The mean memory depth for a Gamma memory of order-K becomes [16],[17],

$$D = \frac{K}{(1-\beta)} \tag{2}$$

and its resolution is

$$R = 1 - \beta.$$

As a consequence, depth and resolution can be adjusted in Gamma memories by simply changing β. The Gamma delay operator $G(z)$ represents the transfer function using z-transform of a single filter stage

$$G(z) = \frac{(1-\beta)}{(z-\beta)}. \tag{3}$$

Eq. (3) can be described as a leaky integrator, where β is the gain of the feedback loop. The recursive rule for context descriptor of order-K can be derived directly from the transfer function (3), as follows:

$$C_k(z) = G(z) C_{k-1}(z) = \frac{1-\beta}{z-\beta} C_{k-1}(z)$$

which can be rearranged as

$$C_k(z) = (1-\beta) z^{-1} C_{k-1}(z) + \beta z^{-1} C_k(z). \tag{4}$$

By using the inverse Z-transform, the recursive expression (1) is obtained.

Let $\mathcal{N} = \{1, \ldots, M\}$ be a set of neurons. Each neuron has associated a weight vector $w^i \in \Re^d$, for $i = 1, \ldots, M$, obtained from a vector quantization algorithm. The Gamma context model associates to each neuron a set of contexts $\mathcal{C} = \left\{c_1^i, c_2^i, \ldots, c_K^i\right\},\ c_k^i \in \Re^d,\ k = 1, \ldots, K$, where K is the Gamma filter order. Given a sequence s, the context set $\mathcal{C}$ is initialized at zero values. From eq. (2) it can be seen that by increasing the filter order, K, the Gamma context model can achieve an increasing memory depth without compromising resolution.

Given a sequence entry, $x(n)$, the best matching unit, I_n, is the neuron that minimizes the following distance criterion,

$$d_i(n) = \alpha_w \left\|x(n) - w^i\right\|^2 + \sum_{k=1}^{K} \alpha_k \left\|c_k(n) - c_k^i\right\|^2 \tag{5}$$

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 (5) every context descriptor in the different filtering stages is required. These contexts are built by using Gamma memories. Formally, the K context descriptors of the current unit are defined as:

$$c_k(n) = \beta c_k^{I_{n-1}} + (1 - \beta)\, c_{k-1}^{I_{n-1}} \quad \forall k = 1, \ldots, K \tag{6}$$

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 models is recovered. Therefore, Merge SOM and Merge NG reduce to particular examples of γ-SOM and γ-NG, respectively, when only a single Gamma filter stage is used ($K = 1$).

Because the context construction is recursive, it is recommended that $\alpha_w > \alpha_1 > \alpha_2 > \cdots > \alpha_K > 0$, otherwise errors in the early filter stages may propagate through higher-order contexts.

2.1 γ-GNG Algorithm

The γ-GNG algorithm is a merge between GNG and the Gamma context model. For the GNG model we use the implementation proposed in [11], which incorporates a node insertion criterion based on entropy maximization. In the following we extend Andreakis's GNG implementation to incorporate γ filters. Neuron ith has associated a weight vector, w^i, and a set of contexts, c_k^i, for $k = 1, \cdots, K$.

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. (6)
4. Find best matching unit (BMU), I_n, and the second closest neuron, J_n, using eq. (5)
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,

$$w^r = 0.75w^u + 0.25w^v \tag{8}$$
$$c_k{}^r = 0.75c_k{}^u + 0.25c_k{}^v$$

    (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,

$$wcount_u = (1 - \tilde{\alpha}) \times wcount_u \tag{9}$$
$$wcount_v = (1 - \tilde{\alpha}) \times wcount_v \tag{10}$$
$$wcount_r = wcount_u$$

11. Decrease winner count variables of all nodes, $j = 1, \cdots, J$ by a factor $1 - \tilde{\beta}$,

$$wcount_j = (1 - \tilde{\beta}) \times wcount_j \tag{11}$$

12. Set $n \rightarrow n + 1$
13. If $n < L$ go back to step 2, where L is the cardinality of the data set.

Typically, $\tilde{\alpha} = 0.5$ and $\tilde{\beta} = 0.0005$.

3 Experiments

Experiments were carried out with two data sets: Mackey-Glass time series and Bicup 2006 time series[1]. The parameter β was varied from 0.1 to 0.9 with 0.1 steps. The number of filter stages K was varied from 1 to 9. Training in γ-GNG is done in a single

[1] Available at Times Series Data Library
http://www.robjhyndman.com/TSDL/data/bicup2006.dat

stage, during 1 epoch for Mackey-Glass time series and 200 epochs for Bicup time series. Parameters α_i are fixed, and decayed linearly with the context order as follows:

$$\alpha_i = \frac{K+1-i}{\sum_{k=0}^{K}(k+1)},\ i = 0 \ldots K \tag{12}$$

with $\alpha_w \equiv \alpha_0$. The parameters used in (7) were set as $\epsilon_w = 0.05$, $\epsilon_c = 0.0006$. After convergence of the algorithm, each neuron will have associated a receptive field defined as the mean input sequence that triggers its selection. With the aim of measuring performance, a time window of 30 past events is defined. The size of this window does not affect the function of the model, and it is used for monitoring purposes only. The temporal quantization error (TQE) [3] is used as a performance criterion. 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.

3.1 Mackey-Glass Time Series

Figure 1 illustrates the Mackey-Glass time series, which is the solution of the differential equation $\frac{dx}{dt} = bx(t) + \frac{ax(t-d)}{1+x(t-d)^{10}}$, for $a = -0.2$, $b = -0.1$, and $d = 17$. 150,000 points were taken from this time series.

The SOM grid size was set as 10×10, with a total number of 100 neurons. Figure 2 shows the temporal quantization error for Merge GNG ($K = 1$), γ-GNG ($K = 9$), γ-NG ($K = 9$), and γ-SOM ($K = 9$). The best result obtained for each algorithm is displayed in Figure 2. It can be observed that the family of γ-filter algorithms largely outperformed Merge GNG ($K = 1$) in terms of TQE, for any value of the index of past inputs greater than 3. For very few lags, e.g. 0 lags, the best quantization is obtained when considering only the magnitude of the data samples, without taking into account temporal contexts. But for more than 3 lags, the temporal contexts help to produce a much better temporal quantization than Merge GNG. All γ-filter variants show a similar TQE for less than 15 past inputs. γ-GNG achieves a slightly better TQE for more than 15 past inputs than γ-NG and γ-SOM. Fig. 3 shows two-dimensional projections of the resulting temporal vector quantization using principal component analysis (PCA) for the Mackey-Glass time series. The dots correspond to neurons, and the links are created by running the time series and connecting the closest neurons.

Fig. 3 shows the projection obtained with a) γ-NG ($K = 9$), b) γ-GNG ($K = 9$), c) γ-SOM ($K = 9$) and d) Merge-GNG ($K = 1$). These projections illustrate clearly the cyclic nature of the Mackey-Glass time series. The Merge GNG projection is much more noisy than the projections obtained with the three γ-filter algorithms.

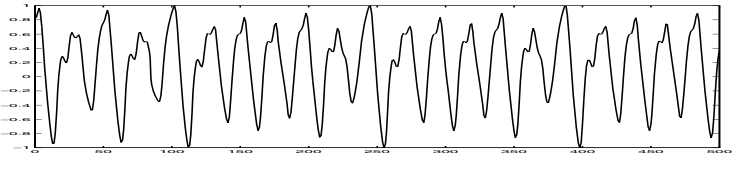


Fig. 1. Mackey-Glass time series

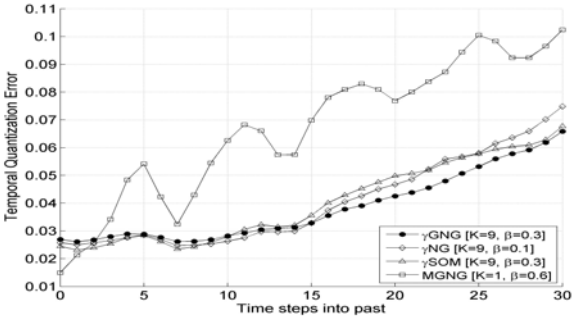
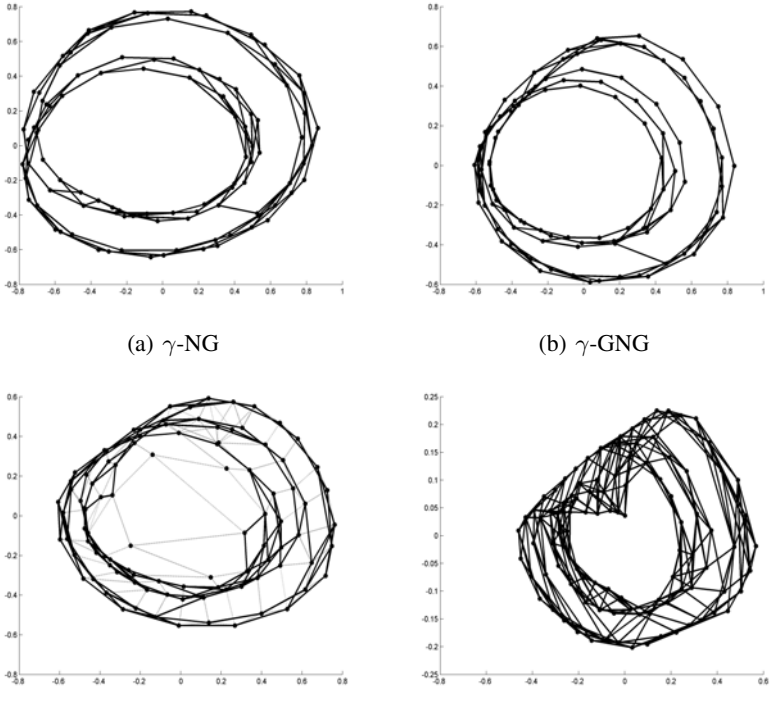


Fig. 2. TQE for Mackey-Glass time series using Merge GNG ($K = 1$), γ-GNG ($K = 9$), γ-NG ($K = 9$),and γ-SOM ($K = 9$)



(a) γ-NG (b) γ-GNG

(c) γ-SOM (d) Merge-GNG

Fig. 3. PCA projection of temporal vector quantization results for Mackey-Glass time series, using a) γ-NG ($K = 9$, $\beta = 0.1$), b) γ-GNG ($K = 9$, $\beta = 0.3$), c) γ-SOM ($K = 9$, $\beta = 0.3$), and d) Merge-GNG ($K = 1$, $\beta = 0.6$)

3.2 Bicup 2006 Time Series

The Bicup time series contains the number of passenger arrivals at a subway bus terminal in Santiago, Chile, every 15-minute interval between 6:30 hours and 22:00 hours for the period March 1-21, 2005. As the behavior of passengers is completely different during the weekends, we trimmed the data by discarding the weekends. Figure 4 shows the Bicup 2006 time series without the weekends.

The SOM grid was set to a 10×10 array, with a total number of 100 neurons. Figure 5 shows the temporal quantization error for Merge GNG ($K = 1$), γ-GNG ($K = 8$), γ-NG ($K = 9$), and γ-SOM ($K = 8$). The parameter β was varied between 0.1 and 0.9 with steps of 0.1, and the best value for each algorithm is displayed in Figure 5. The family of γ-filter algorithms largely outperformed Merge NG ($K = 1$) in terms of TQE, for any value of the index of past inputs greater than 0. All γ-filter variants display a similar TQE behavior. γ-NG achieves a slightly better TQE than γ-GNG and γ-SOM for more than 10 past inputs.

Fig. 6 shows two-dimensional projections of the resulting temporal vector quantization using PCA for the Bicup 2006 time series. The dots correspond to neurons, and the links connect the closest neurons determined by running the time series. Fig. 6 shows the projection obtained with a) γ-NG ($K = 9$), b) γ-GNG ($K = 8$), c) γ-SOM ($K = 8$) and d) Merge-GNG ($K = 1$). These projections show that there are some interesting periodicities in the Bicup 2006 time series. The Merge GNG projection is clearly more noisy than the other three γ-filter based projections.

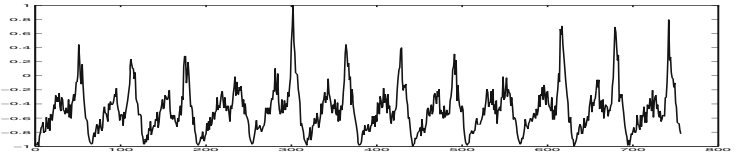


Fig. 4. Bicup 2006 time series

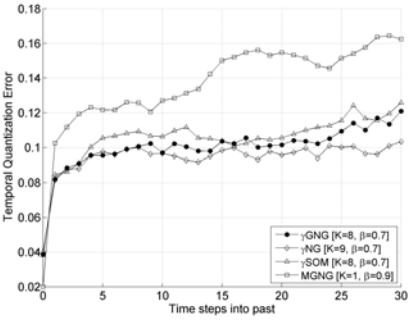
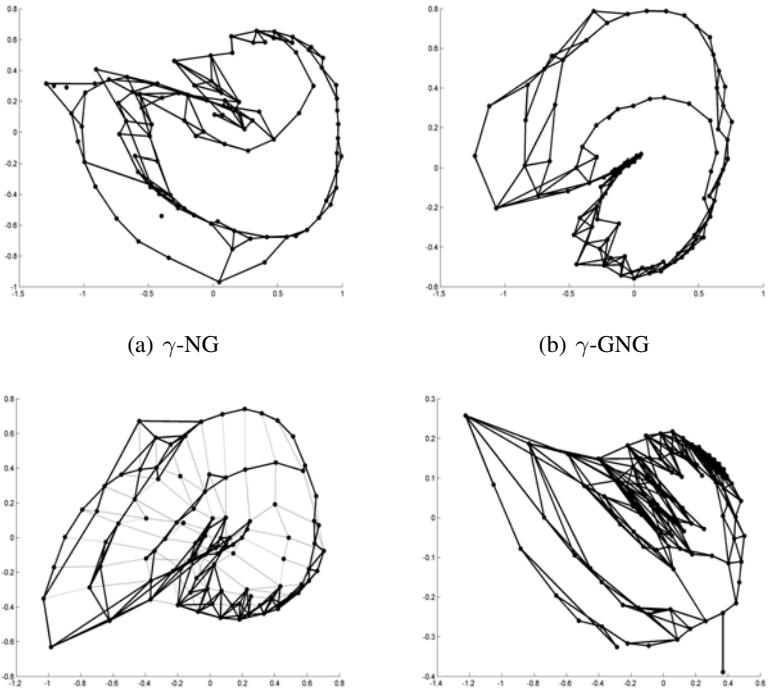


Fig. 5. TQE for Bicup 2006 time series using Merge GNG, γ-NG, γ-GNG, and γ-SOM



(a) γ-NG (b) γ-GNG

(c) γ-SOM (d) Merge-GNG

Fig. 6. PCA projection of the temporal vector quantization obtained for Bicup 2006 time series, using a) γ-NG ($K = 9$) ,b) γ-GNG ($K = 8$), c) γ-SOM ($K = 8$), and d) Merge-GNG ($K = 1$).

4 Conclusion

A γ-filter version of GNG has been implemented, and compared with γ-NG and γ-SOM models using the temporal quantization error as performance metric. The so-called γ-GNG model is a generalization of the Merge GNG model, where the latter corresponds to the particular case when a single context is used (gamma filter of order one). It has been shown empirically using two time series that by adding more contexts the temporal quantization error tends to diminish. Although the TQE performance of γ-GNG is similar to those of γ-NG, the former has the advantage of being a much faster algorithm. In addition, in γ-GNG there is no need of having two stages of training as usually done in γ-NG and γ-SOM algorithms.

Acknowledgment

This research was supported by Conicyt-Chile under grants Fondecyt 1080643 and 1101701.

References

1. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (1995)
2. Strickert, M., Hammer, B.: Merge SOM for Temporal Data. Neurocomputing 64, 39–72 (2005)
3. Voegtlin, T.: Recursive Self-Organizing Maps. Neural Networks 15, 979–991 (2002)
4. Hammer, B., Micheli, A., Neubauer, N., Sperduti, A., Strickert, M.: Self Organizing Maps for Time Series. In: Proceedings of the Workshop on Self-Organizing Maps (WSOM), Paris, pp. 115–122 (2005)
5. Hammer, B., Micheli, A., Sperduti, A., Strickert, M.: Recursive Self-Organizing Network Models. Neural Networks 17, 1061–1085 (2004)
6. Chappell, G.J., Taylor, J.G.: The Temporal Kohönen Map. Neural Networks 6, 441–445 (1993)
7. Tiňo, P., Farkas, I., van Mourik, J.: Dynamics and Topographic Organization of Recursive Self-Organizing Maps. Neural Computation 18, 2529–2567 (2006)
8. 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 4, 558–569 (1993)
9. Fritzke, B.: A Growing Neural Gas Learns Topologies. In: Tesauro, G., Touretzky, D.S., Leen, T.K. (eds.) NIPS, pp. 625–632. MIT Press, Cambridge (1995)
10. 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)
11. 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)
12. De Vries, B., Principe, J.C.: The Gamma Model- A New Neural Model for Temporal Processing. Neural Networks 5, 565–576 (1992)
13. 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)
14. Estevez, P.A., Hernandez, R., Perez, C.A., Held, C.M.: Gamma-filter Self-organizing Neural Networks for Unsupervised Sequence Processing. Electronics Letters (in press, 2011)
15. Principe, J.C., Giuliano, N.R., Lefebvre, W.C.: Neural and Adaptive Systems. John Wiley & Sons, Inc., New York (1999)
16. Principe, J.C., Kuo, J.M., Celebi, S.: An Analysis of the Gamma Memory in Dynamic Neural Networks. IEEE Trans. on Neural Networks 5, 331–337 (1994)
17. Principe, J.C., De Vries, B., De Oliveira, P.G.: The Gamma Filter – A New Class of Adaptive IIR Filters with Restricted Feedback. IEEE Transactions on Signal Processing 41, 649–656 (1993)

Self-Organizing Maps of Nutrition, Lifestyle and Health Situation in the World

Yasir Mehmood[1], Mudassar Abbas[2], Xi Chen[1], and Timo Honkela[1]

[1] Department of Information and Computer Science
[2] Department of Biomedical Engineering and Computational Science
Aalto University School of Science
ymehmood@cc.hut.fi, {xi.chen,timo.honkela}@tkk.fi,
mudassar.sabir@aalto.fi

Abstract. In this article, we present an analysis of the impact of nutrition and lifestyle on health at a global level. We have used Self-organizing Maps (SOM) algorithm as the analysis technique. SOM enables us to visualize the relative position of each country against a set of the variables related to nutrition, lifestyle and health. The positioning of the countries follows the basic understanding of their status with respect to their socioeconomic conditions. We have also studied the relationships between the variables supported by the SOM visualization. This analysis presents many obvious correlations but also some surprising findings that are worth further analyses.

1 Introduction

The overall relationship between unhealthy diet and deteriorating health is obvious and generally well understood. Establishing this relationship from data may lead us to further identify the severity of the relationship concerning different specific aspects, and to take appropriate corrective actions. Moreover, these types of analyses can be used to create social awareness regarding the strong impact of certain nutrition on the health of individuals and thus promoting overall health and wellbeing in the society. The technical report series 916 of World Health Organization explains that the rapid industrialization, in the previous decade, has affected the health and nutrition especially in the developing countries, which has resulted in "inappropriate dietary patterns, decreased physical activities and increased tobacco use, and a corresponding increase in diet-related chronic diseases, especially among poor people" [1]. The report also states that existing scientific evidence has helped in identifying the role of diet in controlling various diseases. However, the evidence is contradicting at times [1].

A possible way to study correlations of different elements of nutrition and lifestyle with the diseases could be to focus on the eating and drinking trends in different parts of the world. The patterns of nutrition intake vary to a great extent in different regions of the world. As this is also true in the case of the prevalence of various diseases. The connections between the nutrition intake and health can be further examined (see e.g. [2]). Volkert [2] studies the nutrition and

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 160–167, 2011.

lifestyle of elderly people in Europe and finds out that these vary widely even within Europe. Volkert continues to state that the elderly in the south consume more vegetables, grains, fruit, lean meat and olive oil whereas relatively more milk products are consumed in the northern European countries. These kind of findings are particularly interesting because a deeper analysis of world-wide eating trends and prevalence of diseases may enable us to identify the regions where some improvements in terms of changing eating habits are essential to promote wellbeing. In our research, we have carried out a similar investigation where we can clearly identify large differences in nutrition intake at the global level. Thus, we can group different regions of the world depending on nutrition intake profiles of each region. We have also explored the links between the prevalence of certain diseases in various countries and citizens′ diet in those countries. The dataset, that we have used for our analysis, contains more than one hundred nutrition, lifestyle and health indicators in 86 countries. The dataset has been obtained from Canibais e Reis [3], with FAO (Food and Agriculture Organization) [4], WHO (World Health Organization) [5] and British Heart Foundation [6]. as the main sources of the data.

In order to inspect the aforementioned aspects of our research, sophisticated analysis techniques are required since the dimensionality of data is large owing to a lot of health, lifestyle and nutrition related features. For this reason, we have used Self-Organizing Maps (SOM) algorithm, a well-known data analysis and visualization technique, to mine interesting correlations. SOM is a suitable means to create an ordered representation of high-dimensional data. The method reduces the complexity of the data and reveals meaningful relationships by mapping the data into an intuitive two-dimensional space. This also helps in understanding dependencies between variables in the data. In our analysis, we have used SOM for a countrywide grouping of the data or data subsets. For instance, SOM enables us to visually perceive the groups of countries based on the spread of different diseases or the intake of certain nutrition elements. SOM also helps in visualizing the relationships between different food items and diseases.

Section 2 provides a brief introduction of SOM and the dataset. Section 3 presents the results of experimentations on the data and finally Section 4 concludes the paper.

2 Method and Data

This section first describes in brevity different features of the dataset and then sheds some light on the technical and mathematical details of SOM.

2.1 Nutrition, Lifestyle and Health Database

The dataset under investigation is comprised of statistics that can be divided into three categories namely health, diet and lifestyle. The first category contains information such as obesity prevalence, incidence of tuberculosis, mortality rates and related variables in different countries. The dietary information includes the

consumption of proteins, sugar and milk products, and various other components of nutrition (see a subset shown in Fig. 1). The lifestyle category provides information related to the drinking and smoking habits etc. This categorization has helped us, as shown in section 3, to group different countries based on the similarity of food consumption and the spread of diseases.

Nutrition	Argentina	China	Ethiopia	Finland	...	USA
Protein (g/day)	94	82	54	102	...	114
Fats (g/day)	100	90	20	127	...	156
Carbohydrates (g/day)	477.5	450.5	366	399.75	...	426
Animal Products (kcal/day)	823	644	96	1164	...	1045
Animal Fats (kcal/day)	72	46	13	131	...	116
Bovine Meat (kcal/day)	342	27	29	90	...	115
Butter, Ghee (kcal/day)	28	1	5	78	...	40
Cheese (kcal/day)	90	1	0	164	...	149
Eggs (kcal/day)	24	74	1	32	...	55
Fats, Animals, Raw (kcal/day)	42	44	7	17	...	75
Fish, Seafood (kcal/day)	10	35	0	59	...	28
Freshwater Fish (kcal/day)	1	20	0	14	...	5
Honey (kcal/day)	1	1	4	4	...	4
Meat (kcal/day)	475	440	44	497	...	451
Milk - Excluding Butter (kcal/day)	222	30	33	438	...	390
Milk, Whole (kcal/day)	127	28	27	218	...	199
Mutton & Goat Meat (kcal/day)	8	15	6	2	...	3
Offals, Edible (kcal/day)	17	10	3	4	...	3
Pelagic Fish (kcal/day)	1	0	0	34	...	7
Pigmeat (kcal/day)	34	343	0	348	...	132
Poultry Meat (kcal/day)	83	51	2	53	...	197
Vegetal Products (kcal/day)	2135	2296	1761	1978	...	2708
Alcoholic Beverages (kcal/day)	67	150	12	183	...	102
...	...	...	...	...	...	...

Fig. 1. A selection of nutrition-related data

2.2 Self-Organizing Maps

Self-organizing Maps (SOM) is an unsupervised learning algorithm, often refered to as an artificial neural network, that models input patterns in a dataset as an ordered collection of model vectors [7]. The self-organizing map is famous as an effective visualization methods, with which high-dimensional input data can be reduced to 2-dimensional data space that can be interpreted and perceived more easily. SOM has been used in many real world problems ranging from analysis of complex data sets to monitoring of large industrial processes [8,9].

In Self-organizing Maps, the output layer of neurons is usually arranged in a 2-dimensional lattice structure. Each input vector in the input space is connected to each neuron in the output space through a synaptic weight connection which is initialized arbitrarily and learned or updated during the learning process.

The result of the learning process is a particular topological arrangement of the neurons in the output layer. The learning mechanism of SOM consists of three main processes, namely competitive process, co-operative process and adaptive process. These three processes combined can be called the learning algorithm of SOM [7].

The experiments in this work have been conducted using the SOM Toolbox Matlab package [10].

3 Analysis and Results

Our analysis on the data is twofold. In the first phase, we visualize the distribution of countries on the 2-dimensional space of SOM, based on the consumption of different nutrients. This explains how countries group with each other regarding the intake of nutrition. The second phase of the analysis shows relationships between the nutrition variables as well as between the nutrition variables and diseases.

3.1 Nutrition Analysis

For the world-wide nutrition analysis, 50 nutrition variables were selected as an input for the SOM algorithm. The groups shown on the SOM map (see Fig. 2) are based on the commonality of consumption of these nutrients in all 86 countries.

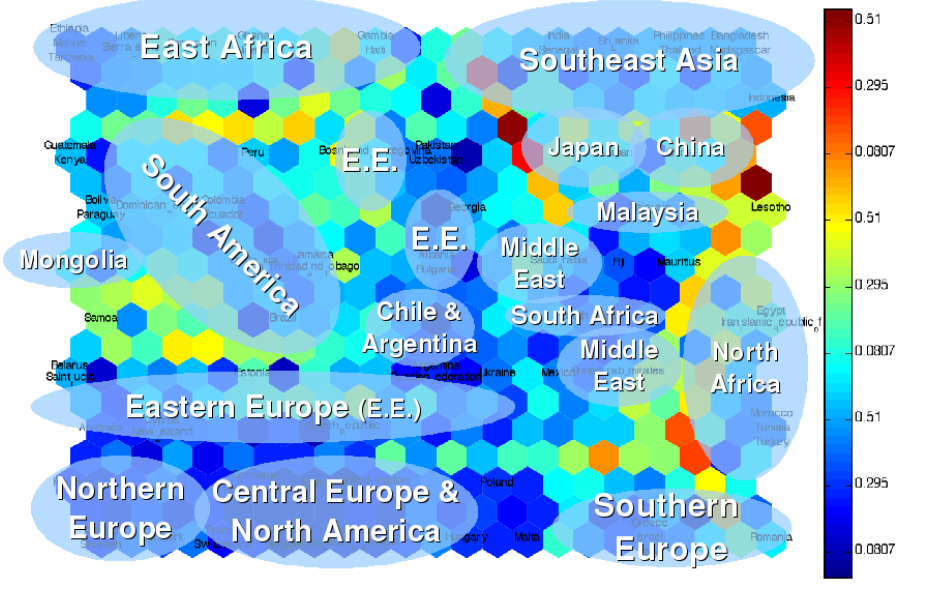


Fig. 2. A map of nutrition variables with an illustration of division of larger geographical groups

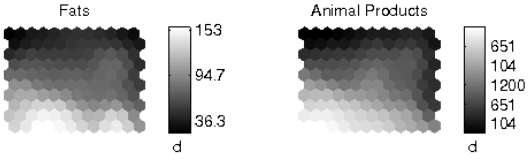


Fig. 3. Component distributions of fats and animal products

In Fig. 2, the structure of clusters on the map is illuminated by the visualization of distances in the original high-dimensional space. The shades of red color indicate large distances in the original space whereas shades of blue refers to relatively shorter distances. The visualization is interesting since the groups of countries on the SOM map appear to have a close resemblance with the region-wise grouping of countries on the globe. This possibly explains the fact that people living in different regions of the world have common preferences for certain foods when compared with others. We have annotated the SOM map by making virtual group boundaries. However, this demarcation is not fully correct since some countries fall in the group of those countries that are not their geographical neighbors. Interesting exceptions include Egypt, Iran and Turkey appearing in the group of North African countries. Moreover, Israel appears in the group of South European countries, and Jamaica in South America. These kind of exceptions are because of the resemblance in eating trends.

We have also performed similar analyses by combining various disease, lifestyle and nutrition variables. The maps of disease and lifestyle variables (disease map is shown in section 3.3) show various similar groups. For example, groups of various South Asian and European countries can be identified (see section 3.3).

3.2 Correlation Analysis

In this section, we present the correlation analysis of variables or components of data in the form of different component maps. The component maps are shown

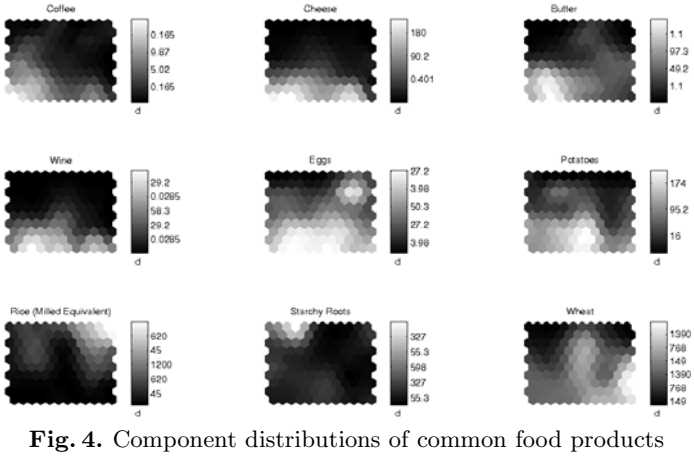


Fig. 4. Component distributions of common food products

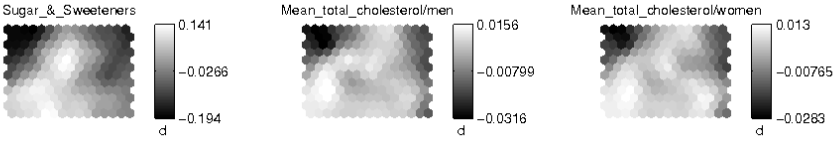


Fig. 5. Relationship between the intake of sugar and sweeteners (kcal/day) and the mean total cholesterol value (mg/dl) in men and women

on the gray scale where the dark shades represent low values and lighter shades represent high values of variables.

The correlation between fats and animal products (see Fig. 3) is significant since it shows strong correlations on both high and low values. This explains the fact that high consumption of fats in European countries is strongly correlated with high consumption of animal products. Similarly, low consumption of fats in some East African and South Asian countries is strongly correlated with low consumption of animal products.

Another component map of nine commonly used food constituents is shown in Fig 4. The map clearly shows strong correlations on relatively low values/ consumption of coffee, cheese, butter, wine, eggs, potatoes, starch and wheat products in various parts of the world.

Disease-Food correlation. Another important aspect of our research was to explore the connection between food and health. For this reason, we studied the

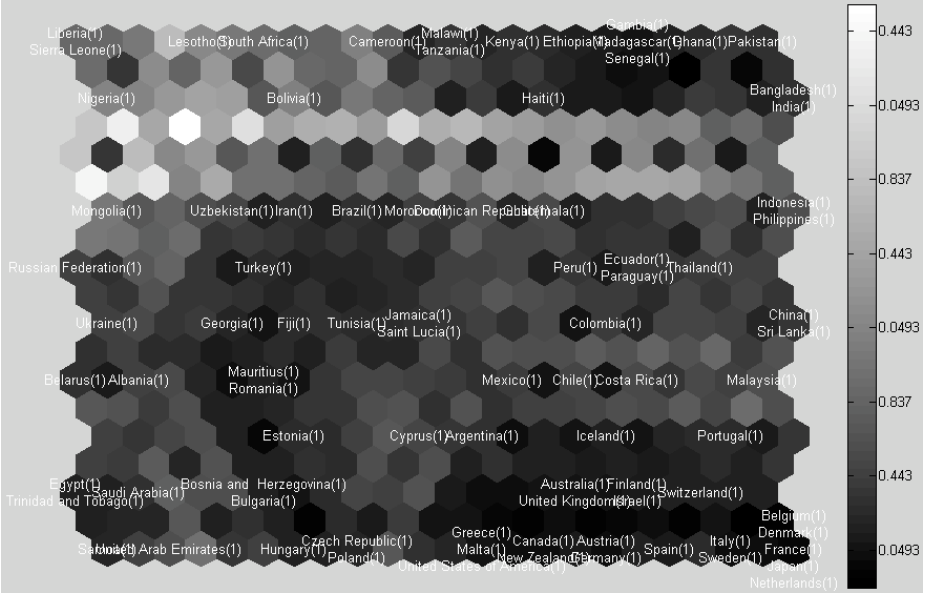


Fig. 6. SOM analysis of disease variables

relationships between some disease-related components and food constituents. An interesting finding, shown in figure 5, indicated a clear correlation between high consumption of sugar or sweeteners and a prevalence of cholesterol in men and women. This result appears to be an important finding from the health point of view (see also [11,12]).

3.3 Disease Analysis

In addition to the nutrition-based analysis, we have also analyzed the prevalence of different diseases in all 86 countries. The disease dataset includes prevalence of heart diseases in men and women, obesity in both genders, cholesterol and many others. The analysis leads us to conclude that countries, like in nutrition-based analysis in section 3.1, tend to fall in the same cluster in which their geographical neighbors reside. The SOM map of disease dataset (see Fig. 6) clearly shows some South Asian countries in the top right corner whereas much of the Europe is clustered in the bottom right.

4 Conclusion

The results obtained using SOM analysis provide a good understanding of the data by not only showing the underlying correlations within different food components but also between food components and diseases.

The study also shows the significance of machine learning techniques in order to infer useful information from different eating and drinking trends in a population. Moreover, deeper analyses performed on richer datasets may bring forth information that can be used for the societal and individual wellbeing.

Acknowledgments: We are grateful to EIT-ICT labs for funding Well-being Innovation Camp, which has greatly helped in conceiving the idea of this paper. We are also thankful to the organizers of the Camp as well as the participants.

References

1. WHO. Technical report series 916 (2003), http://whqlibdoc.who.int/trs/
2. Volkert, D.: Nutrition and lifestyle of the elderly in europe. Journal of Public Health 13, 56–61 (2005)
3. Canibais e Reis. Integrated nutrition, lifestyle and health database: epidemiological information for an improved understanding of diseases of civilization (2009), http://www.canibaisereis.com/2009/03/21/nutrition-and-health-database/
4. FAO: Statistical yearbook 2005-2006 - consumption, http://www.fao.org/economic/ess/en/
5. WHO. Global health atlas, http://apps.who.int/globalatlas/dataQuery/
6. British heart foundation statistics website, http://www.heartstats.org/atozindex.asp?id=8
7. Kohonen, T.: Self-organizing maps. Springer Series in Information Sciences (2001)

8. Díaz, I., Domínguez, M., Cuadrado, A.A., Fuertes, J.J.: A new approach to exploratory analysis of system dynamics using SOM. Applications to industrial processes. Expert Systems with Applications 34(4), 2953–2965 (2008)
9. Abonyi, J., Nemeth, S., Vincze, C., Arva, P.: Process analysis and product quality estimation by self-organizing maps with an application to polyethylene production. Computers in Industry 52(3), 221–234 (2003)
10. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: Self-organizing map in matlab: the som toolbox. In: Proceedings of the Matlab DSP Conference 1999, Espoo, Finland, November 1999, pp. 35–40 (1999)
11. Welsh, J.A., Sharma, A., Abramson, J.L., Vaccarino, V., Gillespie, C., Vos, M.B.: Caloric sweetener consumption and dyslipidemia among us adults. Journal of American Medical Association 303, 1490–1497 (2010)
12. Welsh, J.A., Sharma, A., Cunningham, S.A., Vos, M.B.: Consumption of added sugars and indicators of cardiovascular disease risk among us adolescents. Circulation 123, 249–257 (2011)

Mining the City Data: Making Sense of Cities with Self-Organizing Maps

Omar Neme[1], JRG Pulido[2], and Antonio Neme[3,4]

[1] School of Economics, National Polytechnic Institute, Plan de agua Prieta 66, Mexico City, MEX
oneme@ipn.mx
[2] Faculty of Telematics, University of Colima
Av Universidad 333, Colima, MEX
jrgp@ucol.mx
[3] Adaptive Informatics Research Centre,
Aalto University, Konemiehentie 2, Espoo, Fin
[4] Complex systems group, Autonomous University of Mexico City
San Lorenzo 290, Mexico City, MEX
aneme@james.hut.fi

Abstract. Cities are instances of complex structures. They present several conflicting dynamics, emergence of unexpected patterns of mobility and behavior, as well as some degree of adaptation. To make sense of several aspects of cities, such as traffic flow, mobility, social welfare, social exclusion, and commodities, data mining may be an appropriate technique. Here, we analyze 72 neighborhoods in Mexico City in terms of economic, demographic, mobility, air quality and several other variables in years 2000 and in 2010. The visual information obtained by self-organizing map shows interesting and previously unseen patterns. For city planners, it is important to know how neighborhoods are distributed accordingly to demographic and economic variables. Also, it is important to observe how neighbors geographically close are distributed in terms of the mentioned variables. Self-organizing maps are a tool suitable for planners to seek for those correlations, as we show in our results.

Keywords: Self-organizing maps, urbanism, data mining, urban analysis.

1 Introduction

Cities are complex structures defined by several processes and variables [1]. Urban planers, as well as policy and decision makers, are urged to understand those processes and variables in order to seek for patterns that allow them to proper planing. Several methodologies have been applied, such as those derived from statistics [2], and others derived from the mathematical and computational modeling [3]. A third tool is that of data mining, in which algorithms are presented with (possibly) high-dimensional data and structures and patterns are identified.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 168–177, 2011.

In an effort to make sense of those variables and phenomena, data visualization may be a relevant task.

Almost 80% of the population of developed countries live in urban areas, whereas almost 50% of the population is urban in some developing countries [2]. In the cities, several dynamics are encountered, such as traffic flows and jams, pollution, crime, innovation processes, cultural activities and several others. Also, the distribution of the population over the city is not neither random nor completely regular. The attributes of the population, such as age distribution and educational level, are important for city planners in order to make correct decisions and future modifications, such as new schools, hospitals and streets.

Regions in a city may be quantified accordingly to several variables. For specialists, it is relevant to identify those regions that presents similar descriptions, and also, to identify the heterogeneity of certain geographic areas. Visualization of those high-dimensional regions is a basic step for planers in order to make sense of the dynamics and attribute distribution [4].

Data visualization in the urban context has widely applied, as in, for example [5], where authors apply self-organizing maps to predict new areas in which cities will be expanded to. Also, in [6], self-organizing maps are applied to generate urban meshes. Data visualization in the urban modeling and planing contexts is relevant for better decisions and understanding of the underlying dynamics. Traditional tools such as principal component analysis, multidimensional scaling, and others lack of some of the major attributes of certain non-linear mappings. Thus, in this contribution, we base our analysis and data visualization describing regions of a city on the self-organizing map.

Our contribution continues as follows. In section 2 we describe the general aspects of urban modelling and urban data analysis. In section 3 we present the specific case of study and briefly describe the self-organizing map, emphasizing its features for data visualization. We also present the results of applying the self-organizing map to visualize urban data, and in section 4 we present some conclusions.

2 Urbanism and Urban Data

Cities and its constituent regions may be quantified in terms of relevant variables. Each region is associated to a feature vector, and from there, data visualization techniques may be applied. We define in this section the fundamental variables that define city areas.

For city planners and policy makers, it is important to distinguish between two major attributes of cities: physical aspects and human-related aspects [7]. The first group refers to variables that take into account the street topology and the distribution of different services as well as of pollution indicators. Examples of these variables are the number of streets, avenues, turnovers, traffic lights, number of schools in a determined region, pedestrian bridges, the number of offices and industries in the area, the number of communicating streets to major avenues, and the average concentration of pollutant in the last month.

We define a neighborhood as a region of blocks that share the same administrative instance. This is of course an artificial division, but is defined as the basic structure within cities. Each neighborhood is defined by an attribute vector. The variables defining the high-dimensional space are now specified. This first group of variables describes not only the distribution of public services and facilities, but also the street network topology, which is fundamental for a proper traffic flow over the city. Two of the most frequent variables are the number of nodes and edges of a street graph. Edges are defined as sub-sections of streets that connect two nodes, and nodes are defined as squares, intersections between streets and dead ends. These variables characterize the possible traffic flow patterns within the neighborhood studied [8]. Also in this group of variables are the pollution conditions. The main pollutant considered in health analysis are CO, O_3, NO, NO_2, SO_2, among others. These pollutant are defined in terms of the average concentration during a given period of time, and the number of measures that exceeded the maximum safe level.

The second group of variables accounts for the description of inhabitants in a determined region. Examples of these variables are the average income, educational level, average number of individuals living in the same household, number of trips generated in a traffic district (generally weighted by the population in the neighborhood), number of attracted trips to a given region, the percentage of houses with proper services, and the proportion of individuals with social security (see Fig. 1-b). The age distribution is also relevant, as well as the gender distribution. Hereafter, we refer to the urban space as the space generated by all these variables (groups one and two).

Cities are dynamic in several ways. Citizens get older, new ones are born, pollution may be controlled, new streets are to be built, new schools may be constructed, or old houses may be demolished in order for new buildings to be constructed. All these processes are difficult to be observed at once by traditional tools, but are very relevant for urban planners.

Megalopolis, defined as cities with more than 10 million inhabitants, are the result of several growing processes [1,2]. Although there is not a formal theory of city growth and development, there are some general patterns more or less accepted by the majority of specialists. In the first process, growing occurs from several isolated urban spots. Small towns and settlements tend to growth by diffusion-related mechanisms and eventually, the free areas will be covered. These small towns are mainly ancient settlements that have suffered from demographic stress and migration from other areas. Also, these small areas may be industrial complexes or administrative facilities that tend to attract edifications towards their vicinity.

The second process for city growth is the planification of completely new settlements, more or less nearby to already existing settlements. These settlements are in general planed as to be self-inclusive, that is, to include schools and other facilities as to minimize the necessity of traveling outside its limits, other for displacement to work areas. The third process is related to the formation of poor or very poor areas, the so called satellite cities or slums [9]. These settlements

are formed by people attracted to the city that can not afford to live in already established neighborhoods. These settlements tend to be isolated from the major streets, with poor or non-existing facilities.

In the next section, we describe the project of analyzing urban data from several neighborhoods in a Megalopolis. The neighborhoods are described by the already defined variables.

3 The Study Case

In this project we analyze urban data of Mexico City, from three major databases [10,11,12]. We have two major interests in studying that data. First, we seek to identify both, neighborhoods with similar urban features and the urban heterogeneity of certain regions. Second, we intend to visualize the evolution of certain neighborhoods from the urban point of view. We compared 72 neighborhoods in two different stages: years 2000 and 2010. We selected those neighborhoods because they are within a distance of up to 1km from a major public transport system established in 2004, so we may, as a side effect, study the impact of it over its surroundings. The mentioned public transport system is an instance of the so-called Bus Rapid Transit (BRT), which consists of a high-capacity bus and an exclusive lane, running over a major avenue that crosses the city from south to north for more than 24km. In year 2000, the BRT was not yet implemented, so we have data of the previous situation in the surroundings neighborhoods.

The neighborhoods we considered in this study vary in size from just a few blocks to up to one hundred, and from 950 inhabitants to up to 25,000. However, there are other variables that are relevant, besides of the obvious ones that relate to size, the already defined variables that constitute the urban space (see Fig. 1-b). The total population of the 72 neighborhoods is 554,590 in 2000 and 551,390 in 2010.

Based on the seminal work by Kaski and Kohonen [13], in which self-organizing maps were applied to study the world poverty distribution accordingly to several variables, we propose the use of SOM to study not only the distribution of instances (neighborhoods here after), but also their evolution on the high-dimensional space. By evolution, we refer to how each neighborhood has modified its own variables in two different times (2000 and 2010).

The self-organizing map is a non-linear projection with the capability of shown in a low-dimensional discrete space the data distribution found in the high-dimensional feature space [14]. One of the main properties of the SOM is the ability to preserve in the output map those topographical relations present in the input data [14]. This attribute is achieved through the transformation of an incoming analogical signal of arbitrary dimension into a discrete low-dimensional map and by adaptively transforming data in a topologically ordered fashion [15]. Each input data is mapped to an unit in the lattice, to the one with the closest weight vector to the input vector, or best matching unit (BMU). The SOM preserves neighborhood relationships during training through the learning

equation (1), which establishes the effect that each BMU has over any other neuron. The two-dimensional SOM is a map from $\Re^d \rightarrow N^2$.

The SOM structure consists of a two-dimensional lattice of units referred to as the map space. Each unit n maintains a dynamic weight vector w_n which is the basic structure for the algorithm to lead to map formation. The dimension of the input space is considered in the SOM by allowing weight vectors to have as many components as features in the input space. Variables defining the input space, and thus, the weight space, are continuous. Weight vectors are adapted accordingly to:

$$w_n(t+1) = w_n(t) + \alpha(t)h_n(g,t)(x_i - w_n(t)) \tag{1}$$

where $\alpha(t)$ is the learning rate at epoch t, $h_n(g,t)$ is the neighborhood function from BMU g to unit n at epoch t and x_i is the input vector. The neighborhood function is to be decreasing with time and distance for proper map unfolding.

With the aim to visualize how neighborhoods are distributed in the high-dimensional space, SOMs were constructed. We made use of software SOM-PACK, available at www.cis.hut.fi/research/som_lvq_pak.shtml. Fig. 2 shows the U-matrix [16]. It is indicated the id of the neighborhood followed by the label that identifies the year (00 or 10 for 2000 and 2010). The gray levels represent the similitude of the surrounding cells, and are useful to identify the clusters formed by the SOM.

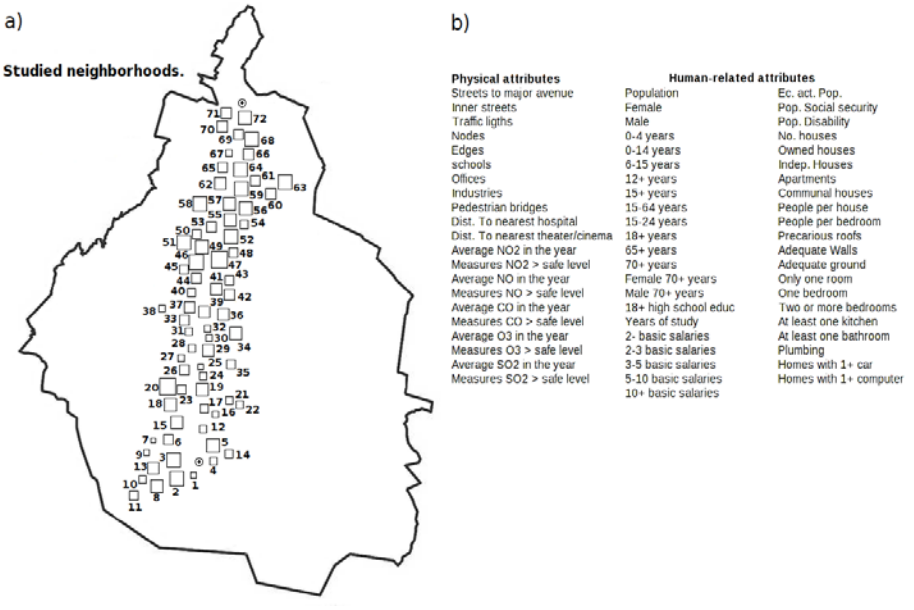


Fig. 1. The 72 neighborhoods contemplated in this study (a). It is shown the general geographic location. The area of polygons representing neighborhoods is proportional to the square of the population in the neighborhood. The variables included in the analysis (b).

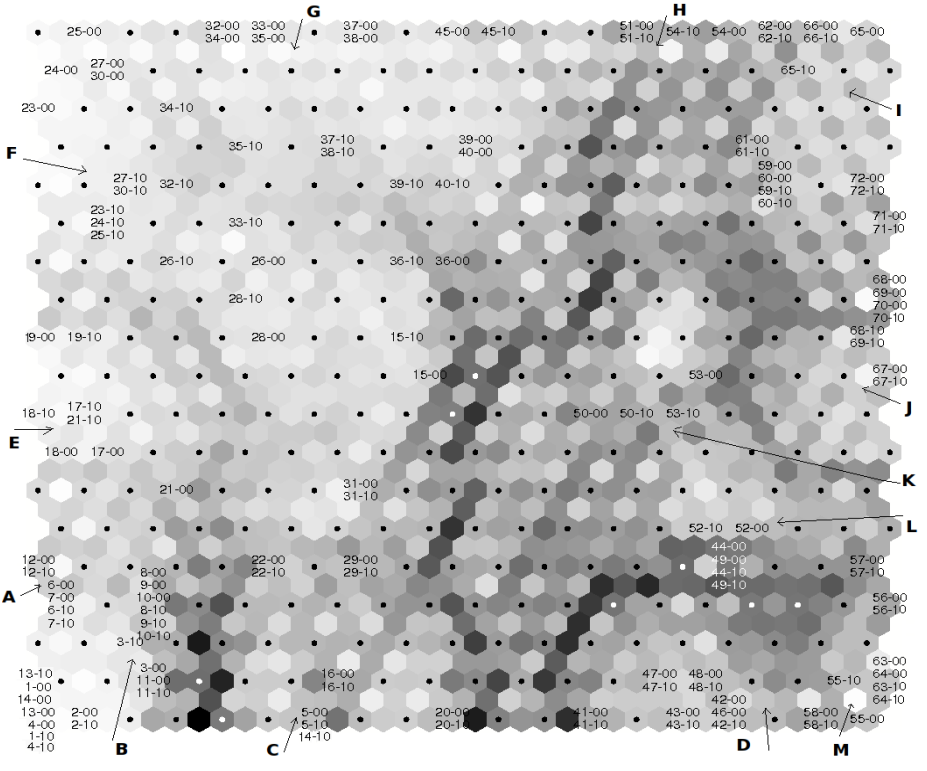


Fig. 2. U-matrix for SOM. The 72 neighborhoods are shown for years 2000 (00) and 2010 (10).

There are several clusters identified by the SOM and highlighted by the U-matrix method. They are labeled A - M. Neighborhoods in cluster A are very poor areas, slums, with poor urban services and poor inhabitants. They are located in the southern part of the studied region. Streets in those neighborhoods are only a few and mainly insufficient to even be represented by a connected graph. Area B is geographically close to neighbors in area A, but have a different description. They are modern neighborhoods, with less than 25 years and there is a high concentration of physicians and researchers working in the nearby hospitals and at the university. Area C is related to wealthy neighbors, with high living standards. The nature of neighbors in this region is that of original settlements (5, 31) and of recent urban constructions (14, 16, 22, 29). Areas B and C are formed by neighborhoods in the south part of the city.

Cluster D is also formed by wealthy neighborhoods, but it includes some from the middle-south and middle-north part of the city. They are mainly modern neighborhoods with adequate urban planning, which includes wide avenues and streets, and several facilities are found in the neighborhood or in the surroundings, with good public transport in the surroundings, including metro stations. Clusters E- G consists of relatively new neighbors, with several apartment build-

ings and close to hospitals and several schools and the university in the surroundings. They are geographically located to the middle-south area of the city.

Clusters H and I includes neighborhoods in the central area and in the northern part of the city. They are similar not only in that they are wealthy neighbors, but also in that the street topology, traffic conditions and air pollution are more or less equivalent. CO levels are high in neighbors within these clusters. Cluster J includes residential neighborhoods within the vicinity of industrial complexes, with medium and working classes, although with several facilities. Finally, clusters K and L are new urban settlements, with good street planning and in general, are among the neighborhoods with the highest standards in Mexico City.

In order to interpret the results, it is important to keep in mind that two factors are being considered simultaneously. The first one is that the studied neighborhoods are situated along a main avenue that crosses Mexico City from south to north for almost $30km$, served by a major public transport system. The analysis of the SOM of these neighborhoods is important as it shed light about similar features and patterns among the city. The first important aspect we observe is that neighbors geographically close to each other tend to have more or less similar positions in the urban space (see also Fig. 1). However, this is not always the case. Neighborhood 5, an original town from where city had growth in the last century, is a wealthy region. The neighborhoods around it (12, 6, 7) are considered poor regions. Also, the differences are not only in the economic sense: the street topology of both regions is different (see Fig. 3).

The second aspect is that each neighborhood is represented by two instances: its description, accordingly to the relevant urban variables, in year 2000 and in year 2010. So, in the same map we found two instance of each neighborhood. The distribution of these two instances is a visualization of how different a neighborhood is in 2010 compared to how it was in year 2000. That is, we can visualize the evolution of each neighborhood in two periods at the time that we visualize how each neighborhood is placed in the urban space with respect to other neighborhoods. In general, ten years is a short period of time to observe significant changes in a city.

In general, it is observed that neighborhoods tend to stay more or less the same, at least in a ten-year period. There are, however, some interesting counterexamples. Neighborhood 14, a residential area with several apartment buildings, has shifted its position towards a cluster of neighborhoods with higher standards. Also, no neighborhood seems to be attracted to clusters defining poor neighborhoods. However, several instances have not abandoned its poverty condition (neighborhoods 1, 3, 6, 7, 10, ...).

In Fig. 3 it is shown the planes for some of the considered variables. A plane indicates in gray levels over each unit the average value of a certain variable of the vectors mapped to that unit. In the first plane (starting at top left), it is shown the distribution for the percentage of households with at least one car. Also, plane 6 shows the percentage of population of each neighborhood that earns 10 or more basic salaries. It is observed a similitude between these

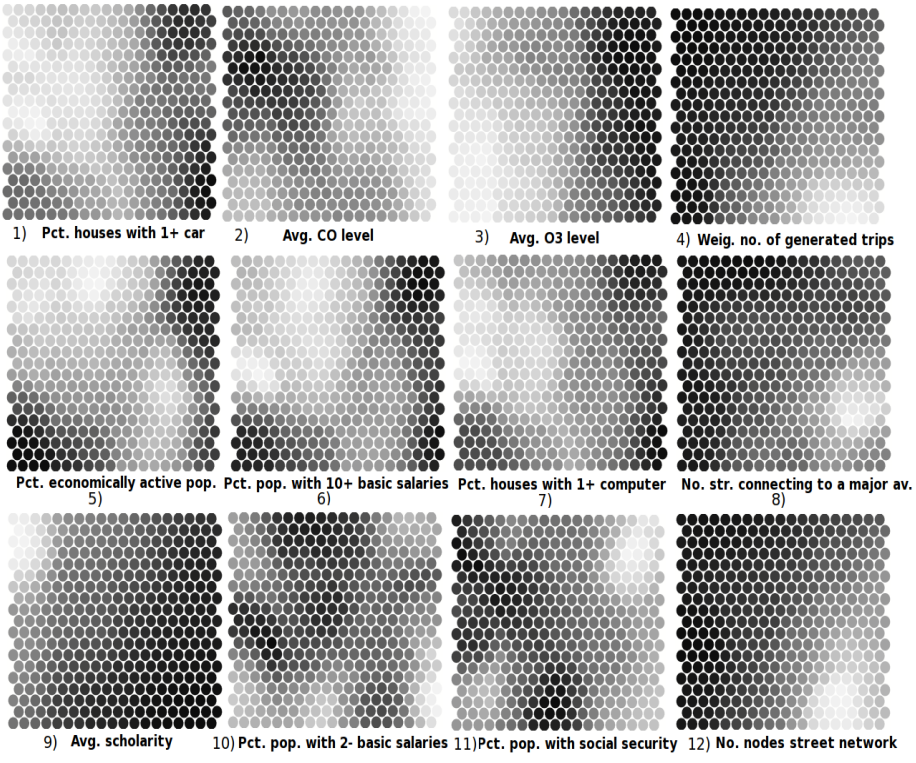


Fig. 3. Planes for some of the considered variables. Gray level indicates the corresponding value of the plane. Light tones indicate higher values.

two planes, which clearly indicates that people with good salaries can afford to buy cars.

Another interesting and previously undetected fact is that neighborhoods with the highest percentage of educational level are not the neighborhoods whose inhabitants earnt the highest salaries, which is in contradiction with what is observed in, for example, the United States [17] (see planes 6 and 9 in Fig 3).

On the side of urban traffic and pollution issues, a few facts are also of interest for urbanists, and were previously undetected. Plane 8 shows the number of street connecting the neighborhood to a major avenue. By comparing this plane with plane 1, it is observed that neighborhoods whose inhabitants have more cars are less connected that neighborhoods whose inhabitants have fewer cars. This, of course, is a cause of traffic jams. This finding express the fact that neighborhoods with several cars will present higher traffic problems not only because there are more cars but also because there are less streets to leave (enter) the neighborhood.

Plane 1 and plane 12 show that neighborhoods with more cars tend to be less connected, that is, less street are connecting the neighborhood with itself. This

is confirmed by the fact that many of the neighborhoods with households with more cars tend to present more traffic problems [18].

Plane 4 shows the number of generated trips from a neighborhood to another one. It is included in this variable both trips to work and trips for social reasons. This variable is important for urbanist and traffic oficials to propose new or additional routes. By comparing this plane with plane 9, it is observed that in those neighborhoods with lower level of education people tend to travel more outside their own neighborhood. From here, it may be inferred that people with higher levels of education tend to live near their jobs.

4 Discussion and Conclusions

In order to make sense of data from different fronts related to urban settlements and inner processes and phenomena, we applied the self-organizing maps. The visualizing capabilities of the self-organizing map were very helpful in identifying some patterns and correlations. We analyzed several neighborhoods within the vicinity of a bus rapid transit, based on several dozens of demographic, economic, environmental and topological variables, and we were able to find some clusters. Those clusters group neighbors with similar descriptions so interesting patterns may be identified.

The ability to visualize simultaneously clusters and their correlation as in the U-matrix and planes, is important to seek for relevant patterns. In the case of urban data, it may lead to the discover of relevant information. In this project, we identified some hidden patterns, previously undetected. Also, we have been able to detect clusters of similar neighborhoods and also.

Visualization of high-dimensional data is a first step for urban planners to make sense of the city and its inner displacements and processes. Self-organizing maps are a good alternative at least in the data visualization and data inspection tasks.

Acknowledgments

This research is derived from a project supported by Instituto de Ciencia y Tecnología del Distrito Federal, Mex. (ICyTDF), under contract PICCT08-55.

References

1. Batty, M.: Cities and Complexity. MIT Press, Cambridge (2005)
2. Bettencourt, L., West, J.: A unified theory of urban living. Nature 467, 912–913 (2010)
3. Batty, M.: Rank clocks. Nature 444, 592–596 (2006)
4. Batty, M., Steadman, P., Xie, Y.: Visualization in spatial modeling. In: Working Paper from Center for Advanced Spatial Analysis (2004)

5. Lobo, V., Cabral, P., Bacão, F.: Self organizing maps for urban modelling. In: Proc. 9th Int. Conf. on Geocomputation (2007)
6. Castro, A., Gómez, N.: Self-organizing map and cellular automata combined technique for advanced mesh generation in urban and architectural design. Int. J. Information Technologies and Knowledge 2, 354–360 (2008)
7. Batty, M.: Urban modelling. Cambridge University Press, Cambridge (1976)
8. Buhl, J., Gautrais, J., Reeves, N., Solé, R., Valverde, S., Kuntz, P., Theraulaz, G.: Topological patterns in street networks of self-organized urban settlements. The European Physical Journal B 49, 513–522 (2006)
9. Milgram, S.: The experience of living in cities. Science 167, 1461–1468 (1970)
10. Instituto Nacional de Estadistica, Geografia e Informatica (National Institute for Statistics, Geography, and Informatics), Mex, http://www.inegi.org.mx
11. Encuesta Origen Destino (Origin - destination survey), http://igecem.edomex.gob.mx/descargas/estadistica/ENCUESTADEORIGEN/
12. Sistema de Información de Desarrollo Social, http://www.sideso.df.gob.mx
13. Kaski, S., Kohonen, T.: Exploratory data analysis by the self-organizing map: structures of welfare and poverty in the world In Apostolos-Paul et al. In: Neural Networks in Financial Engineering, pp. 498–507 (1996)
14. Kohonen, T.: Self-Organizing maps, 3rd edn. Springer, Heidelberg (2000)
15. Hujun, Y.: The self-organizing maps: Background, theories, extensions and applications. In: Computational Intelligence: A Compendium, pp. 715–762 (2008)
16. Ultsch, A.: Self organized feature maps for monitoring and knowledge aquisition of a chemical process. In: Proc. of the Int. Conf. on Artificial Neural Networks, pp. 864–867 (1993)
17. United States Department of Labor. Bureau of labor statistics, http://www.bls.gov/emp/ep_chart_001.htm
18. Graizbord, B.: Geografía del transporte en el área metropolitana de la Ciudad de México. Colmex (2008)

A Discussion on Visual Interactive Data Exploration Using Self-Organizing Maps

Julia Moehrmann, Andre Burkovski, Evgeny Baranovskiy,
Geoffrey-Alexeij Heinze, Andrej Rapoport, and Gunther Heidemann

Intelligent Systems Group,
University of Stuttgart,
Universitaetsstr. 38, 70569 Stuttgart, Germany
firstname.lastname@vis.uni-stuttgart.de,
firstname.lastname@vismail.informatik.uni-stuttgart.de

Abstract. In recent years, a variety of visualization techniques for visual data exploration based on self-organizing maps (SOMs) have been developed. To support users in data exploration tasks, a series of software tools emerged which integrate various visualizations. However, the focus of most research was the development of visualizations which improve the support in cluster identification. In order to provide real insight into the data set it is crucial that users have the possibility of interactively investigating the data set. This work provides an overview of state-of-the-art software tools for SOM-based visual data exploration. We discuss the functionality of software for specialized data sets, as well as for arbitrary data sets with a focus on interactive data exploration.

1 Introduction

Self-organizing maps (SOMs) [1] have been widely employed for data exploration tasks in the last decade. Their popularity is especially due to the ability of creating low-dimensional, topology preserving representations of high-dimensional data. Visualizations of these representations help the user to understand the distribution of data elements in the feature space. In recent years, a variety of visualization techniques have been developed which support users in the identification of clusters and correlated data. Most of these visualizations have been included in software tools which provide different means to analyze and explore the data set.

The active research area of visual analytics early identified the need to interact with data visualizations. Interactive data exploration is relevant in every domain where users want to gain insight into the data. Interaction techniques allow users to mentally connect visual elements with actual data. The application of such techniques may result in a deeper understanding of the visualization and allow a goal-oriented analysis of the data.

Most current SOM-based visualizations focus on cluster formation, contribution of variables (features) to these clusters, and homogeneity of clusters. However, to gain insight into the data, interaction with the clustered data is crucial.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 178–187, 2011.

The selection of feature vectors and investigation of data linked to these feature vectors can improve the visual data exploration task. Additionally, more information may be transported through such interactive visualizations than single static visualization are capable of.

This paper presents a short overview of SOM-based visualization techniques and evaluates existing SOM-based software tools regarding their application to visual interactive data exploration. We summarize the results in order to provide a reference for interactive tools. We conclude with the discussion of the current state-of-the-art software tools.

2 SOM-Based Visualization Techniques

Most SOM-based visualization techniques focus on cluster identification. In the following, we give an overview of such visualizations for SOMs.

Ultsch proposed the U-Matrix [2], which can be displayed by color coding the distances between neighboring map units. The resulting representation allows an effective cluster identification. Ultsch also propose several extensions to the U-Matrix visualization. The P-Matrix [3] visualizes data densities and distances for map units. The U*-Matrix [4] combines the P-Matrix and the U-Matrix. This results in an advanced highlighting of clusters. Merkl et al. [5] developed a cluster connection visualization where connections between neighboring units are drawn inside clusters only and small distances are highlighted. Smoothed data histograms [6] use contours to display areas of equal data density. Neighborhood graphs [7] connect map units if the data is close in feature space, as well as in the resulting map. Poelzlbauer et al. [8,9] created vector field visualizations. An arrow which points in the direction of the cluster center is displayed for each map unit, resulting in a smooth vector field which can be well interpreted for cluster identification. The orthogonal representation creates a borderline representation, thereby displaying cluster boundaries. Tasdemir et al. [10] extended the usual SOM visualization with a connectivity matrix. Map units which are the first and second best matching unit (BMU) for any feature vector are connected. The width of the connecting line is increased with the number of feature vectors for which this applies. This visualization not only highlights homogeneous clusters with high BMU connectivity and dense data areas but also reveals distortions and topological errors in the map. This visualization is mentioned to provide a complete overview, it is, however, not implemented in any of the discussed tools. Latif et al. [11] proposed a sky-metaphor visualization. In contrast to other visualization techniques not only the map units are displayed. Instead, individual feature vectors are displayed and their position is adapted according to their similarity with neighboring units. This visualization is beneficial since the mapping to BMUs is often coarse and fails to distinguish the differences of individual feature vectors mapped onto the same units. Vesanto [12] described several SOM-based visualizations, like data histograms, adapting the position of map units relative to each other according to the accuracy of map units with data samples, and component planes. Data histograms visualize the number of

feature vectors which are projected onto a map unit (as BMU), either utilizing the size of the unit, color coding, or a three dimensional plot where the height of each bar indicates the number of feature vectors. Although these visualizations focus on the identification of clusters, Vesanto additionally discusses visualization techniques which focus on data analysis. Response surfaces, for example, display the relative goodness of all units for the data set (using color coding) and thereby provide information about the quality of the map. Component planes allow the investigation of individual variables of codebook vectors by color coding their influence on the U-Matrix visualization.

Mayer et al. [13] developed visualizations for coloring map units based on category labels of the feature vectors (class visualization). The category labels are provided by the ground truth data. Rauber et al. [14] developed a visualization which also displays labels for clusters, however, no ground truth data is required. Instead, the feature variable which is primarily responsible for the assignment of a feature vector to a specific map unit is used as the label. This visualization aims to provide users with a better understanding of the map topology. We will summarize such techniques as *class histograms* in the remainder of this work. Neumayer et al. [15] introduced the metro map metaphor which extends component planes by connecting the lowest and highest values in each plane. The resulting lines are combined in one diagram. This visualization is especially useful for investigations of high-dimensional data sets where the analysis of individual component planes becomes unfeasible.

3 Criteria

Dzemyda et al. [16] conducted a comparative analysis of six software tools for SOM-based visual data exploration, with a focus on the interpretability of provided visualizations. Their conclusion was that insight might best be gained by using several such tools. Although this recommendation seems unfeasible, it is not surprising, since the analysis focused only on different visualizations and accordingly found advantages and disadvantages for all of them.

In contrast, we evaluate SOM-based software tools for the use in visual interactive data exploration tasks. Our criteria focus on interaction, since it is a key component and allows users to explore data, as well as their correlations and properties, in detail. We will base our discussion of tools on Keim's visual analytics mantra [17]: *analyze first – show the important – zoom, filter and analyze further – details on demand.* In other words, visualizations should provide users with interaction techniques that allow selection of data subsets, zooming, filtering, and displaying detailed information, as well as appropriate diagnostics and statistics.

We propose the following criteria for the assessment of the software tools discussed in Section 4:

– *Data preprocessing.* The common input for data exploration tools are feature vectors in a specific data format. The software should provide methods to process the input data before SOM training. Data preprocessing routines should at least provide functionality for data centering and data normalization.

- *Visualizations.* Section 2 gives an overview of SOM visualization techniques. Different visualizations allow different interpretations and as such are important for users to gain insight. The software should provide several state-of-the-art visualizations. In other words, does the software provide sufficient methods for analyzing and displaying important aspects?
- *Visualization of data.* Feature vectors are connected to some arbitrary original data. In case of images, texts or motion trajectories it is favorable to have the original data displayed. Visualizations of original (or raw) data provide a good overview of the data distribution and quality of the resulting map.
- *Interaction with the map.* The topological structure of the SOM is a fundamental detail of the algorithm. Interaction techniques for low-dimensional map topologies are beneficial for the understanding of high-dimensional feature space projections. Map interaction should provide basic zoom and filtering methods, like showing the data distribution of a specific category or with a specific data property.
- *Interaction with the data.* Although the input for the SOM learning algorithm are feature vectors, they are connected to specific data. Showing details on demand not only refers to displaying feature vectors, but also to reveal the actual data. Interaction with the data refers to the possibility of displaying features or original data, as well as to the possibility of modifying or rearranging the data.
- *Interaction with visualization.* To leverage visualizations users should be able interact with them. The basic form is the interactive adaption of visualization parameters, e.g. switching between U-Matrix and component plane visualizations or adapting thresholds.
- *Labeling data.* The tool should provide the possibility of labeling data. Labeling data is a critical criteria for users who want use the SOM for annotation purposes.

All investigated tools behave differently and provide different levels of interaction and visualization. To provide a basis for our discussion we assign the following values in the evaluation (see Table 1). If a tool does not provide the given functionality we assign the *none* (-) label for that tool, if it provides basic methods that partly fulfill the criteria, we assign the *basic* (∘) label. If a tool implements one or more advanced solutions for a criteria, we assign the *advanced* (+) label.

4 Tools for SOM-Based Interactive Data Exploration

This section gives an overview of existing software for SOM-based visual data exploration. To provide a clear structure we divided the software tools into two categories: tools developed for the investigation of specialized data sets and tools for the exploration of arbitrary data.

4.1 Special Purpose Tools

A variety of SOM-based systems were developed for the exploration of special data sets, e.g. image data or gene expression data. The following list of (academic) tools provides an overview of such systems and gives a description of their

purpose and functionality. A summarization of the visualization techniques provided by each tool and an assessment of their interactivity is given in Table 1.

VALT, Heidemann et al. [18,19]. This system was developed for the purpose of labeling image data in an augmented reality (AR) scenario. The image data is clustered with a SOM and the map is displayed to the user with one representative image per unit. Users may select and assign labels to map units or individual images. The U-Matrix can be displayed in the background of the map.

Moehrmann et al. [20]. Similar to VALT but optimized for use on desktop computers. Image data is clustered with a SOM using arbitrary features. A zoomable user interface (UI) is used to display the SOM in two levels of detail (map units and all images for a BMU). Class labels can be assigned to map units or individual images.

Schreck et al. [21]. SOM-based visualization of time-series data. The system is optimized for the application to time-dependent data and integrates techniques to automatically identify interesting candidate views. Selection of map units is possible, as well as displaying the component planes for them. The system includes a U-Matrix visualization, color coding of quantization errors and nearest neighbor connections. Users may merge or expand units, as well as edit, create or delete them. The layout of the map may also be rearranged to better suit the users' expectations. The time-dependent data is displayed as a two-dimensional trajectory.

Torkkola et al. [22]. They propose the use of SOM-based visualizations for mining gene expression data. The map can be color coded according to the value of individual components (i.e. one component plane). It is, however, not obvious whether the selection of a component can be performed interactively. Users may set thresholds to identify clusters in the SOM. A basic degree of interactivity with the map is given with the possibility of zooming into areas of interest. Users may thereby investigate the data in detail (given as line plots).

Kanaya et al. [23,24]. Software tool developed for exploratory analysis of gene expression data. No data preprocessing is included in the software. It provides data histograms, component planes, as well as a comparison map which highlights differences between two component planes. This visualization is of interest for this special data set since components refer to individual experiments. The applicability of this visualization to other data sets is unknown. Feature vectors can be investigated in detail by selecting a unit in the visualization.

4.2 Analytical Purpose Tools

In contrast to specialized tools, a variety of systems exist which were developed for visual data exploration on arbitrary data sets. The following list summarizes these tools and gives an overview of their functionality and interaction techniques:

SOM toolbox for Matlab [25]. Extensive toolbox for Matlab with focus on SOM calculation. Provides data preprocessing, U-Matrix, component planes, and class

histograms. The interaction is mainly command-line based. There are a few dialogs which are useful for beginners, one of which allows the adjustment of visualization parameters. However, no direct interaction with the visualization is possible, since the visualization is recalculated and displayed in a new window. The usual interaction with maps is provided, i.e. zooming and rotating. Selection of map units or interaction with the data is not possible. Another tool that could be discussed here is R-project with its packages for SOM visualizations. However, regarding its functionality and its command-line interface it is on one level with the Matlab SOM toolbox. It provides only static visualizations and no degree of interactivity except via command-line. It will therefore not be included in the detailed discussion.

SOMVis [26]. Extends the SOM toolbox for Matlab with basic interactions like zooming, rotating and adding labels and arrows to the map. Labels are attached to the map only, not to units. Selection of map units is possible, but the only information that can be displayed for selected units is their position. There are a lot of options for configuring the visualizations. However, visualizations are not interactively adapted but recalculated.

Java SOMToolbox [27]. Provides basic interactions with the map, like zooming and rotating. The selection of map units is possible. Labels can be manually added to the visualization but cannot be attached to units. It is possible to display cluster labels which show details about the feature vectors inside this cluster. However, actual interaction with the data is not possible. The visualization may be adapted by choosing different color schemes, displaying/hiding cluster boundaries, or selecting units according to their value ranges on component planes. This toolbox utilizes most of the visualization introduced in Section 2.

Peltarion Synapse [28]. This software is not focused on SOMs but provides some standard SOM visualizations. A dialog is available for configuring the visualizations with options like show U-matrix, show component planes, or map size. Selection of units or clusters is possible. Different views are linked, thereby enabling users to identify selected regions in other visualizations. As units are selected, additional information, like number of units, number of data vectors, as well as minimum, maximum, and average values per component are displayed. Viewing and editing feature vectors is possible. Assigning labels to units is possible by manually editing the class component of a feature vector. Regarding the visualizations, configuration of parameters is possible via dialogs.

Viscovery SOMine [29]. This workflow-based software is available in a basic edition and in an expert edition. It includes extensive data preprocessing abilities, like the extraction of new features, data transformation, and outlier removal. Zooming the map is restricted to five zoom levels and only available for component planes. Component planes may be rearranged or hidden. Selection of individual units or clusters is possible, as well as selecting units according to value ranges of components. The actual data for selected clusters can be investigated in a table like arrangement. Viewing data for one map unit is possible by manually modifying the clustering. A basic labeling functionality is available

Table 1. Summarization of the visualization techniques and interaction possibilities provided by all data exploration tools. The interaction criteria are judged as -(not available), ∘(basic functionality), and +(advanced functionality). Visualizations are not judged by their quality. The *basic* sign (∘) indicates the availability of a visualization.

	Java SOMToolbox	Viscovery SOMine	Synapse	Matlab SOMToolbox	SOMvis	VisiSOM	Heidemann et al.	Moehrmann et al.	Schreck et al.	Torkkola et al.	Kanaya et al.
Data preprocessing	∘	+	∘	∘	∘	∘	-	-	-	-	-
Interaction with map	+	+	+	-	∘	∘	∘	∘	∘	∘	∘
Interaction with data	-	∘	∘	-	∘	∘	-	-	-	∘	-
Interaction with visualization	+	∘	∘	-	∘	∘	∘	∘	∘	-	-
Label assignment	-	∘	∘	-	-	-	+	+	-	-	-
Data visualization							∘	∘	∘		
Data histograms	∘	∘	∘	∘	∘	∘					∘
Class histogram	∘	∘		∘	∘		∘	∘			
Cluster connections	∘										
Clustering	∘	∘	∘	∘	∘					∘	
Component planes	∘	∘	∘	∘	∘	∘			∘	∘	∘
Metro map	∘				∘						
Neighborhood graph	∘				∘						
U-Matrix	∘		∘	∘	∘	∘	∘	∘	∘		
U*-matrix	∘				∘						
P-Matrix	∘				∘						
Response surfaces	∘				∘				∘		
Sky-metaphor visualization	∘										
Smoothed data histograms	∘				∘						
Vector fields	∘				∘						

for codebook vectors only. The whole software is focused on cluster investigation and therefore does not provide many visualizations. The expert edition, which we did not evaluate, seems to provide additional functionality which allows the context-sensitive display of data vectors for selected units.

VisiSOM [30]. A commercial software for SOM-based data exploration which provides only basic (2D and 3D) visualizations. Codebook vector data (complete, or individual component values) can be displayed on an additional axis for selected map units. Additionally, feature vector data is displayed in a list below the visualization and is updated if the selection is modified.

5 Discussion

As seen in the previous section, the tools can be divided in two classes: special purpose tools and analytical purpose tools. Special purpose tools like Heidemann et al., Moehrmann et al., or Schreck et al. provide advanced interaction techniques which are optimized for the exploration of specific data sets. However, only few visualizations are available. This results from the fact that both, image and time-series data, can be displayed well and is easy to grasp for users in its natural form. Additional advanced cluster visualizations would not provide much benefit. In contrast, gene expression data is much more generic due to its abstract nature. These tools apply a lot of advanced visualizations but lack in interactivity, although the investigation of different experiments could probably be performed more efficiently with appropriate techniques.

Analytical purpose tools, like Java SOMToolbox and the SOMVis extension for Matlab, provide various visualizations for standardized input data. Additionally, visualizations may be adapted to better suit the users' purpose through the user interface or via command line. Viscovery SOMine and Peltarion Synapse provide linked views which allow users to investigate several visualizations at once, thereby supporting the identification of correlations. Both systems provide good interaction techniques for the map and the visualizations, but only very basic interaction with the data. Although SOMine allows the assignment of labels to codebook vectors, it is not possible to identify incorrectly projected feature vectors. In contrast, Synapse allows the manual editing of class attributes in the feature vector but does not support the assignment of labels to whole clusters. It is difficult for analytical purpose tools to display the actual underlying data. Such visualizations require special models and optimizations and one can hardly implement visualizations for all possible applications. However, interactive or embedded data visualizations are essential for extensive data exploration tasks and special tasks, like image or text labeling.

Most tools allow basic interaction with the maps, but only few provide additional information depending on the level of detail. For example, labels are displayed without consideration of the current zoom level which leads to visual clutter. Instead, labels for whole clusters could be displayed in a low zoom level and as the zoom is increased labels could be displayed more detailed, for codebook vectors or, in a very high zoom level, for individual data vectors. A major advantage for visual data exploration would be the possibility to augment the visualization with customizable information. Instead of displaying class labels, individual components could be of interest. It is possible to display such components in many tools but not depending on the level of detail.

In our opinion the focus of research in the area of visual SOM-based data exploration has to shift from the development of slightly advanced visualizations for cluster identification to the development of interactive user interfaces with the aim to support users in their exploration task. Although it could be argued that the visual analytics research area is responsible for such developments we believe that the development has already begun to head for the direction of visual analytics with the realization of sophisticated visualizations. The evolution of

SOMs to an interactive data exploration tool will have a positive influence on other areas like human computer interaction, since SOMs are rather intuitive. Given adequate interaction techniques this fact could be exploited to greatly simplify the learning phase and reduce the initial barrier for non-expert users.

6 Conclusion

Recent developments in the area of visual data exploration with SOMs has focused on the improvement of visualizations for cluster detection or detection of correlations. However, special purpose tools have been developed which provide sophisticated interaction techniques which were optimized for specific tasks, like labeling images. Although various software tools exist which allow visual exploration of arbitrary data sets, the interaction techniques are in a very basic stadium. We believe that the focus of future research has to shift from the development of further cluster visualizations to the development of sophisticated interaction techniques for arbitrary data. SOM-based visual data exploration can be performed intuitively and is therefore especially of interest for non-expert users from other domains.

References

1. Kohonen, T.: The Self-Organizing Map. Proceedings of the IEEE 78(9), 1464–1480 (1990)
2. Ultsch, A., Siemon, H.P.: Kohonen's Self Organizing Feature Maps for Exploratory Data Analysis. In: International Neural Networks Conference, pp. 305–308. Kluwer Academic Press, Paris (1990)
3. Ultsch, A.: Maps for the Visualization of High-Dimensional Data Spaces. In: Workshop on Self-Organizing Maps, pp. 225–230 (2003)
4. Ultsch, A.: U*-Matrix: A Tool to Visualize Clusters in High Dimensional Data. Technical Report 36, Dept. of Mathematics and Computer Science, University of Marburg, Germany (2003)
5. Merkl, D., Rauber, A.: Alternative Ways for Cluster Visualization in Self-Organizing Maps. In: Workshop on Self-Organizing Maps, pp. 106–111 (1997)
6. Pampalk, E., Rauber, A., Merkl, D.: Using Smoothed Data Histograms for Cluster Visualization in Self-Organizing Maps. In: Dorronsoro, J.R. (ed.) ICANN 2002. LNCS, vol. 2415, pp. 871–876. Springer, Heidelberg (2002)
7. Pölzlbauer, G., Rauber, A., Dittenbach, M.: Advanced visualization techniques for self-organizing maps with graph-based methods. In: Wang, J., Liao, X.-F., Yi, Z. (eds.) ISNN 2005. LNCS, vol. 3497, pp. 75–80. Springer, Heidelberg (2005)
8. Poelzlbauer, G., Dittenbach, M., Rauber, A.: A Visualization Technique for Self-Organizing Maps with Vector Fields to Obtain the Cluster Structure at Desired Levels of Detail. IEEE International Joint Conference on Neural Networks 3, 1558–1563 (2005)
9. Poelzlbauer, G., Dittenbach, M., Rauber, A.: Advanced Visualization of Self-Organizing Maps with Vector Fields. Neural Networks 19(6-7), 911–922 (2006)
10. Tasdemir, K., Merenyi, E.: Exploiting Data Topology in Visualization and Clustering of Self-Organizing Maps. IEEE Transactions on Neural Networks 20(4), 549–562 (2009)

11. Latif, K., Mayer, R.: Sky-Metaphor Visualisation for Self-Organising Maps. J. Universal Computer Science (7th International Conference on Knowledge Management), 400–407 (2007)
12. Vesanto, J.: SOM-based Data Visualization Methods. Intelligent Data Analysis 3, 111–126 (1999)
13. Mayer, R., Aziz, T.A., Rauber, A.: Visualising Class Distribution on Self-Organising Maps. In: de Sá, J.M., Alexandre, L.A., Duch, W., Mandic, D.P. (eds.) ICANN 2007. LNCS, vol. 4669, pp. 359–368. Springer, Heidelberg (2007)
14. Rauber, A.: LabelSOM: on the Labeling of Self-Organizing Maps. In: International Joint Conference on Neural Networks, vol. 5, pp. 3527–3532 (1999)
15. Neumayer, R., Mayer, R., Polzlbauer, G., Rauber, A.: The Metro Visualisation of Component Planes for Self-Organising Maps. In: International Joint Conference on Neural Networks, pp. 2788–2793 (2007)
16. Dzemyda, G., Kurasova, O.: Comparative Analysis of the Graphical Result Presentation in the SOM Software. Informatica 13(3), 275–286 (2002)
17. Keim, D., Mansmann, F., Schneidewind, J., Ziegler, H.: Challenges in Visual Data Analysis. In: 10th International Conference on Information Visualization, pp. 9–16 (2006)
18. Heidemann, G., Saalbach, A., Ritter, H.: Semi-Automatic Acquisition and Labelling of Image Data Using SOMs. In: European Symposium on Artificial Neural Networks, pp. 503–508 (2003)
19. Bekel, H., Heidemann, G., Ritter, H.: Interactive image data labeling using self-organizing maps in an augmented reality scenario. Neural Networks 18(5-6), 566–574 (2005)
20. Moehrmann, J., Bernstein, S., Schlegel, T., Werner, G., Heidemann, G.: Optimizing the Usability of Interfaces for the Interactive Semi-Automatic Labeling of Large Image Data Sets. In: HCI International. LNCS, Springer, Heidelberg (to appear, 2011)
21. Schreck, T., Bernard, J., von Landesberger, T., Kohlhammer, J.: Visual Cluster Analysis of Trajectory Data with Interactive Kohonen Maps. Information Visualization 8, 14–29 (2009)
22. Torkkola, K., Gardner, R.M., Kaysser-Kranich, T., Ma, C.: Self-Organizing Maps in Mining Gene Expression Data. Information Sciences 139(1-2), 79–96 (2001)
23. Kanaya, S., Kinouchi, M., Abe, T., Kudo, Y., Yamada, Y., Nishi, T., Mori, H., Ikemura, T.: Analysis of Codon Usage Diversity of Bacterial Genes with a Self-Organizing Map (SOM): Characterization of Horizontally Transferred Genes with Emphasis on the E. Coli O157 Genome. Gene 276(1-2), 89–99 (2001)
24. Simple-BL SOM Website, http://kanaya.naist.jp/SOM/
25. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: Self-Organizing Map in Matlab: the SOM toolbox. In: Matlab DSP Conference, pp. 35–40 (1999)
26. SOMVis, TU Wien, http://www.ifs.tuwien.ac.at/dm/somvis-matlab/index.html
27. Java SOMToolbox, TU Wien, http://www.ifs.tuwien.ac.at/dm/somtoolbox/
28. Peltarion Synapse, http://www.peltarion.com/products/synapse/
29. Viscovery SOMine, http://www.viscovery.net/somine/
30. VisiSOM, http://www.visipoint.fi/visisom.php

Design of a Structured 3D SOM as a Music Archive

Arnulfo Azcarraga and Sean Manalili

De La Salle University
2401 Taft Avenue, Manila, Philippines
arnie.azcarraga@delasalle.ph
scmanalili@msn.com

Abstract. A structured 3D SOM is an extension of a Self-Organizing Map from 2D to 3D where a structure has been built into the design of the 3D map. The 3D SOM is a 3x3x3 cube, with a distinct *core cube* in the center, and 26 exterior cubes around the center. The structured SOM mainly uses the 8 *corner cubes* among the 26 exterior cubes. Used to build a music archive, the SOM learning algorithm is modified to include a four-step learning and labeling phase. The first phase is meant only to position the music files in their general locations within the core cube. The second phase is meant to position the music files in their respective corner cubes. The third phase is meant to do a fine adjustment of the weight vectors in the core cube. The fourth phase is the labeling of the map and the association of music files to specific nodes in the map. Through the embedded structure of the 3D SOM, a precise measure is developed to measure the quality of the resulting trained SOM (in this case, the music archive), as well as the quality of the different categories/genres of music albums based on a novel measure of the *attraction index* and the *fidelity* of music files to their respective music genres.

Keywords: Self-Organizing Map, 3D map, music archive, SOM quality, fidelity measure.

1 Introduction

Self-Organizing Maps [1] are artificial neural networks that translate high dimensional input data into a low dimensional representation, in usually a 2D planar map. The benefit of using 2D visualization is that it is easy to visualize the relationship among the cells of the map [2]. By employing a technique such as U-Matrix [3], the clusters formed can be spotted and inspected visually.

The 2D SOM has been used in various types of applications, one of which is the use of the SOM as a music archive. For example, [4] uses a 2D SOM to organize a music collection. In visualizing the map, a metaphor of a geographic map is employed wherein islands represent musical genres. Islands signify the presence of songs while water signifies the absence of songs. To interact with the system, the system would need to click around the map. Another application using a 2D SOM is described in [5], where various hardware interfaces were used to navigate the 2D SOM. Among the hardware interfaces, included are an eye-tracker, wii-mote, iphone and desktop controllers.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 188–197, 2011.

A research which utilized a 2D SOM but provided a 3D visualization of the map is described in [6]. Similar to the work described in [4], a metaphor of a geographic map was used to visualize the organized map. However, a 3D virtual landscape was employed wherein the height of an island signals the number of songs associated to the cluster. To navigate through the map, a gamepad interface is used. Indeed, a SOM can in fact be rendered in 3D but the difficulty is that that viewing the 3D SOM would require rotating, zooming or moving 3D models to actually determine the relationship among the map units.

In this paper, we present a variant of SOM which is a *structured SOM* that organizes high-dimensional data into a 3D map. The 3D SOMhas an embedded structure that then paves the way for measuring the quality of the SOM organization. Section 2 discusses the design of a structured SOM while Section 3 illustrates a practicaluse of the structured SOM as a music archive. Section 4 discusses the concept of an *attraction index* and *fidelity measure* which are used to gauge the quality of the 3D SOM. Section 5 gives the conclusion.

2 Design of a Structured SOM

A structured SOM is a 3D map that uses a rectangular lattice and is represented as a 3x3x3 cube, much like a Rubik's cube. The 3D SOM has 27 sub-cubes of the same size. In our experiments, each sub-cube is a 3D map with 9x9x9 nodes. In structured SOMs, we assign categories to each of the exterior cubes to be specialized maps for each of the categories. The *core cube* at the center takes care of combining together in one 3D map all the input files from all possible categories. For the experiments that we will discuss in this paper, we only used the 8 exterior cubes at the corners which we refer to as *corner cubes*. Figure 1 shows their location in the 3D map. The 8 *corner cube* positions were assigned to specific categories, based on an initial training of the core cube.

Figure 2 illustrates the four-step supervised training and labeling of a structured SOM. The supervised training of the *core* and *corner* cubes pre-supposes that there is a labeled training set, where each input element has an accompanying category label. The first phase of training involves the supervised training of the *core cube*, followed

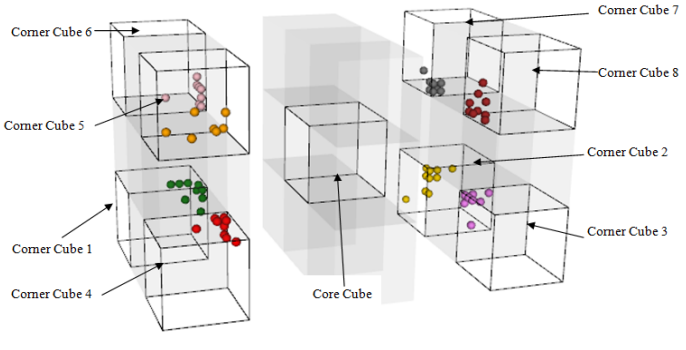


Fig. 1. Corner cube positions in a structured SOM

by the training of the *corner cubes*, then again the training of the *core cube*, then finally the labeling of the *corner* and *core cubes*. The initial supervised training of the *core cube* assigns the general positions of the input files in the map, which would be refined later. The training of *corner cubes* then positions the input files in their respective *corner cubes*, according to their category labels. The second training of the *core cube* (third phase) is designed to fine-tune the weights of the node vectors in the *core cube*, since the initial training phase was just meant for a rough representation by the *core cube* node vectors of the various interrelationships of the items in the input dataset.

The training parameters used for the three training phases are quite different, with learning rates being high during phase 1 ($\alpha = 0.75$) and being much lower during phase 3 ($\alpha = 0.10$), while the number of training cycles is only 10,000 in phase 1, but 50,000 in phase 3. The supervised training of *core* and *corner* cubes is essentially like the usual unsupervised training of regular SOM, except that the best-matching-unit for a given input element is constrained to be from among the pre-assigned nodes that correspond to the accompanying category label. As for the adaption of the weights of node vectors, the learning rule and the Gaussian function for the diminishing effect of the learning rate within the neighborhood of the best matching unit follow the usual learning mechanism of self-organizing maps.

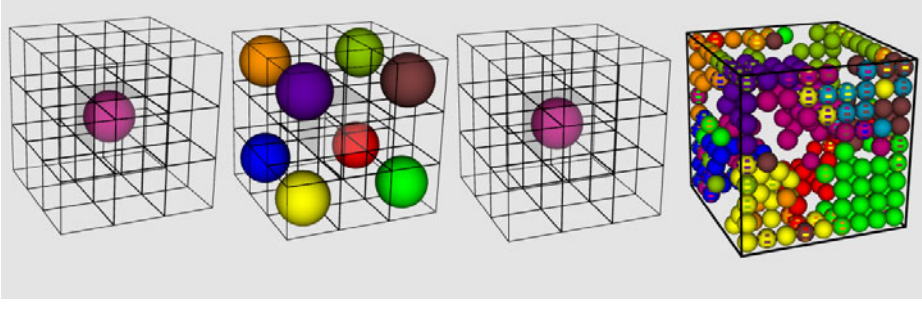


Fig. 2. Training of a structured SOM

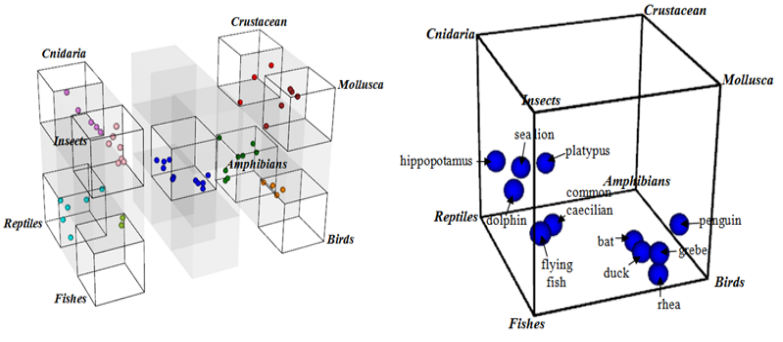
Because of the nature of the learning mechanism of self organizing maps where all other nodes in the map are updated (but at decreasing learning rates as the distance of the node to the best-matching unit increases), the *core cube*also gets trained during supervised training of the *corner (exterior) cubes*. Consequently, by the collective influences of all the *corner cubes*, the *core cube* gets to represent the inter-relationships of the various data elements of the entire data set.

We tested the use of structured SOMs by using an artificial dataset of animal data, each represented by binary features. In the animal data set, there are 10 animal categories, namely *amphibians, birds, fish, reptiles, cnideria, crustaceans, mollusks,* and *insects. Mammals* and *special animals* (e.g. *bats* as mammals that fly, *platypus* that has characteristics of mammals and birds, *ducks* as birds that swim, *whales* as mammals that swim in the ocean) were used to do experiments on the *core cube*.

Following the usual implementation of self-organizing maps, the unknown category of certain animals can be identified by computing the distance of the input

vector to each of the node vectors of the *core/corner cube*. The label of the nearest node (*best matching unit*) would be the identified class/category of the unknown animal.

With the structured SOM set-up, there are nodes in the *core cube* that straddle along the boundaries between two or more classes or categories, somewhere mid-way between two or three corners. These nodes would correspond to input items that have multiple categories, such as animals that have traits that make them look both *mammal* and *bird*, or both *fish* and *mammal*, or perhaps songs that are partly *hip-hop*, partly *reggae*, and maybe partly *rock*.



a) b)

Fig. 3. a) training of structured SOM using an animal dataset b) distribution of "special animals" in the core cube that has been trained with an animal dataset of 8 different classes/categories of animals

Figure 3a shows a screenshot of the structured SOM that has been trained using an animal dataset. We had only loaded the trained *core cube* with the set of *special animals*. Figure 3b is a zoomed-in display of the *core cube* of Figure 3a, with labels on the spheres to identify the various *special animals* and how they are positioned vis à vis the different categories assigned to the eight *corner cubes*. *Duck, rheas* and *grebes* are pulled towards the *fish* corner which is why these *birds* are positioned a little away from the *birds* corner. As for the *bat*, it is positioned near the *birds* corner. With regard to the *flying fish*, it is positioned between *fishes* and *insects*. Since most of the *insects* in the data set have the ability to fly, the *flying fish* is being pulled by the *insects* corner (less by the *bird* corner) since the *flying fish* has more features that are shared with *insects* than with *birds* as far as the features we used were concerned.

3 Interactive 3D Music Organizer

We used the structured SOM as a music archive and trained the 8 *corner cubes* with music files from 8 music categories or genres, namely*blues, classical, country, disco, hip-hop, jazz, reggae* and *rock*. One genre is assigned to one corner cube. As explained earlier, the corner cube's node vectors get updated (along with the node

vectors of nearby nodes in the *core cube*) only when a music file from that particular genre is selected during training. The labeled training set has music files from all the 8 genres and the sequence of music files for training is completely random.

Interactive 3D Music Organizer (i3DMO) is one instance of the structured SOM that performs content-based organization of a music collection following the training and labeling scheme of structured SOMs. It provides a 3D visualization of the map, with zoom-in and zoom out functions, rotations, probes to the interior of the cube, etc.

The application provides a media player functionality that enables playback of songs and managing of playlists. The music data set used by the application is composed of songs with extracted music features. The music features were retrieved by performing a specialized extraction process that relies on applications using digital signal processing to extract a song's content.

The feature extraction and selection method is a fairly lengthy and elaborate process, using various statistical techniques to select pertinent features. In particular, each music file (song) was divided into 10-second segments and the first and last segments were discarded. The features were computed for each of the segments and then the mean and standard deviation of each feature was computed. For a given feature, all segments whose extracted feature value deviates from the mean by more than 1 standard deviation were discarded. After further segment filtering, the remaining valid segments were used to compute for the average feature value for each feature and for each song. Once all these were computed, Weka [8] was used to filter out the redundant features (i.e. features that were highly correlated). From an initial list of about 692 possible features culled from the literature, specifically *MusicMiner* [9] and *jAudio* [10], the list was finally trimmed down to less than 70. This was the basis for building a corpus of 1,000 songs, with 10 genres, of 10 albums per genre and 10 songs per album.

The map is visualized as 3x3x3 cube, with a light grey border to establish the boundaries of the sub-cubes. Inside each sub-cube (whether a *corner cube* or a *core cube*), a node is represented by a sphere if there is a song associated to the node. The color of the sphere depends on the assigned color of the song's genre. If the songs assigned to a given node belong to different genres, there will be rectangular horizontal strips of different colors on the surface of the sphere to denote the other genres. Figure 4 displays a visualization of the music archive as a 3x3x3 cube. Note the core cube in the middle with all the songs of all genres, and the corner cubes loaded with just the songs of their assigned genres (as distinguished by the color).

In Figure 4, there are 10 genres in the music collection. Eight (8) out of the 10 genres are assigned to the corner cubes of the structured SOM. The 2 genres which are not assigned to a corner cube are *metal* and *pop music* genres, which were used for various experiments on the *core cube*. It must be stressed here that the genres of the songs included in the dataset were based on the genre tags from well-known, authoritative websites that classify songs according to genres.

In the i3DMO application, placing the mouse cursor over a sphere displays a pop-up window which contains the song information. In addition, a song from the list of songs associated to the node/sphere is randomly selected and a short playback is heard to allow the user to determine the kind of music associated to the sphere. An external hardware interface was taken into account in the design of the structured SOM. The hardware interface should allow a user to position his fingers in his personal space to

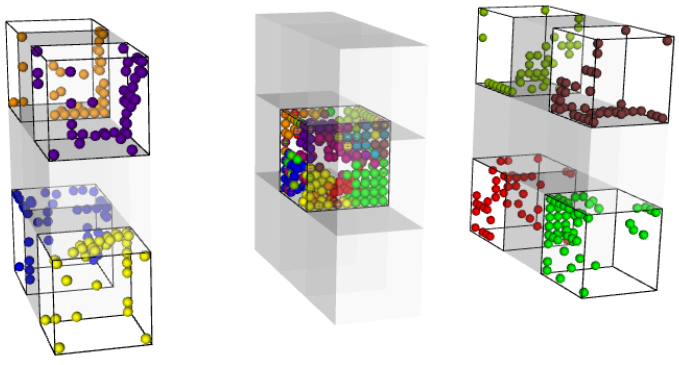


Fig. 4. Visualization of the music collection

interact with the SOM. By pointing at a specific location in the 3D space around the user, a corresponding node or cluster can be selected. Then, commands can be issued by performing hand gestures. The primary objective of this interface is to enable a visually impaired individual to interact with the 3D SOM [11]. With this kind of interface, a visually impaired user need not "see" the map, but can just interact with the space around it, guided by audio cues, a huge part of which would be the samplings or short playbacks that the user hears when pointing at different positions (corresponding to spheres) in the 3D space around.

4 Measuring the Quality of the 3D Map

Self-Organizing Maps have been traditionally used as an unsupervised learning model, and since the input vectors do not have assigned labels, and the membership to clusters of nodes on the map had not been as crisp, very little work have been carried out to measure the quality of the trained SOM other than [12] and [13] – compared to other supervised learning models of other types of neural networks. In other words, if we use the trained SOM as a music archive, it is not straightforward to figure out whether songs that are associated with specific nodes in the map should in fact be associated to those nodes and not to other nodes in the map. Is there a notion of "quality of the trained SOM" that we can use to evaluate a given SOM, vis à vis a given data set?

The design of the structured SOM, as proposed here, lends itself naturally to a novel way of evaluating the quality of the resulting SOM organization. We measure the quality of the SOM through what we refer to here as the *attraction index* of a given music filetowards a specific category. In our current music archive application, we measure the *attraction index* of the music files of all songs per music genre, and we compare the average *attraction index* of each of the genres.

For the whole music archive, with all 8 genres combined, we can also compute for the *fidelity measure* that evaluates the degree by which music files that are of a given identified genre have been in fact assigned to their proper music genres or categories.

The *fidelity measure**f**(k)* of category *k* is the *average attraction index* to category *k* of all music files that belong to category *k*.

More formally, the attraction index, denoted by *att-index (i, k)* of music file *i* to the *core cube* corner assigned to category *k* is defined as follows:

$$1.0 \; - \; \frac{\textbf{\textit{dist}}\,(bmu\,(\mathrm{i}),\, corner\,(\mathrm{k}))}{\mathrm{r}\,\sqrt{3}} \qquad (1)$$

where ***dist*** is the Euclidean distance between two points in 3D Cartesian space, *bmu(i)* is the x,y,z position of the best matching unit to input *i* in the map, *corner (k)* is the x,y,z position of the corner assigned to category *k*, and *r* is the number of nodes in each of the x, y, or z dimension of the *core cube*.

Figure 5 depicts the *attraction index* of a specific music file to each of the eight (8) corners of the *core cube*. Unless a music file really sounds as if it is a blend of all types of genres, we expect music files to be positioned relatively nearer to a specific corner, or perhaps somewhere in between two corners. In figure 5, the corner with a red oblong depicts the pre-assigned corner for the genre of the music file.

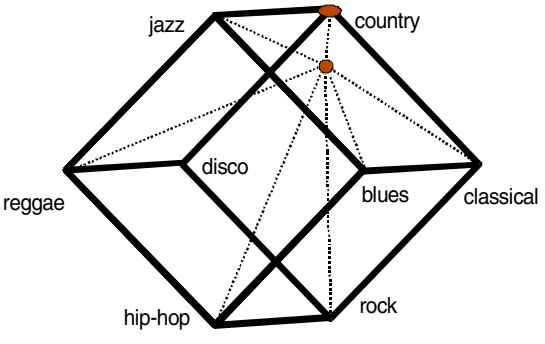


Fig. 5. The *attraction index* of a music file vis à vis the corners of the *core cube*

As defined above, the attraction index *att-index (i,k)* ranges in value from 0 to 1, with a value of 1 signifying that the input element *i* is associated with the very corner node that is assigned to category *k*, and a value 0 signifying that the input element is associated to the corner directly opposite the correct or desired corner. Table 1 lists the average *attraction indices* by music genre. The average attraction indices by music genre were based on 10 independent runs. For each run, 50% of the music files were randomly chosen to train the music archive, while the remaining 50% were used to compute for the attraction indices after loading them into the *core cube*.

In the ideal scenario, the average *attraction index* for a given category *k* must be highest for the corner associated with the same category *k*. These are the average *attraction indices* along the right diagonal of Table 1. These average attraction indices along the diagonal are what we refer to as the *fidelity measure*. Note from Table 1 that for all music genres, the highest average attraction indices per row (or for a given music genre) are precisely those on the diagonal of Table 1. We have made various

other experiments and it is clear that this is not always the case – and would depend on the type of training employed, the training parameters used, as well as the over-all "quality" or sometimes "complexity" of the input dataset.

Table 1. Average attraction index per music genre relative to specific music genres

	Average Attraction Indices							
Genre	Blues	Classical	Country	Disco	Hip-hop	Jazz	Reggae	Rock
Blues	**0.8271**	0.5311	0.3576	0.2298	0.5347	0.5270	0.3657	0.3643
Classical	0.5300	**0.8138**	0.5184	0.3563	0.3564	0.3710	0.2299	0.5257
Country	0.4375	0.5583	**0.7494**	0.4887	0.2810	0.5184	0.3564	0.3847
Disco	0.2464	0.3645	0.4795	**0.7929**	0.3922	0.3391	0.5191	0.5400
Hip-hop	0.5188	0.3618	0.2217	0.3632	**0.8423**	0.3475	0.5306	0.5427
Jazz	0.5067	0.3836	0.5400	0.3866	0.3535	**0.7806**	0.5086	0.2632
Reggae	0.4128	0.2823	0.3841	0.5234	0.5401	0.5280	**0.7371**	0.3861
Rock	0.3437	0.5107	0.3724	0.5486	0.5079	0.2370	0.3654	**0.8056**

The *fidelity measure* ***f****(k)* is the average *attraction index* of all music files belonging to category *k* vis à vis the corner associated with category *k*. Table 2 gives the mean *m* and standard deviation *s* of the *attraction indices* for all the music files in the collection. Table 2 further shows the *standard scores*, or *z-values*, for each of the genres. The *z-values* are computed as the difference between the fidelity measure of a given genre *g* and the mean *m* of all attraction indices to *g* for all music files, divided by the standard deviation *s*. The *standard score* measures the positive or negative deviation from the mean in terms of number of standard deviations from the mean.

Table 2. Raw and standard *fidelity measure* of each music genre

Genre	Fidelity Measure	*m*	*s*	z-value
Blues	0.8271	0.4788	0.2000	1.74
Classical	0.8138	0.4767	0.1911	1.76
Country	0.7494	0.4526	0.1800	1.65
Disco	0.7929	0.4604	0.2012	1.65
Hip-hop	0.8423	0.4769	0.1941	1.88
Jazz	0.7806	0.4561	0.1936	1.68
Reggae	0.7371	0.4512	0.1780	1.61
Rock	0.8056	0.4762	0.1887	1.74

Music genres with high *z-values* for the fidelity measures have music files that are closest to the corner of the core cube assigned to their genre, as compared to music files belonging to other genres. The *z-value* of the fidelity measure is a more accurate measure of the true "fidelity" of the music files belonging to a given genre. In Table 1, we can see that the *disco* genre has a slightly higher raw fidelity score than *jazz*, but the *jazz* music genre has in fact a higher *z-value* ("standardized" fidelity measure).Of all the music genres, *hip-hop* yielded the highest *fidelity measure*, while *reggae* had the lowest *fidelity measure*.

5 Conclusion

A SOM is usually a regular 2D rectangular or hexagonal lattice where nodes that are spatially close in the 2D map are associated with input elements that are similar in the input environment. It is, however, feasible to design a SOM as a 3D map, and the learning algorithm remains virtually the same.

We presented a 3D SOM that is used as a music archive. More important than just extending the SOM from 2D to 3D, we have a built-in structure in the design of the 3D map in such a way that we distinguish between eight (8) *corner cubes* and the *core cube* in the center and that each *corner cube*has an assigned music genre. We have had to alter the learning algorithm by having a three-step learning phase followed by a labeling and music loading phase. The training phases are supervised, and target both the *corner cubes* and the *core cube*.

Through the embedded structure of the 3D SOM, we also presented a novel way of measuring the quality of the resulting trained SOM (in this case, the music archive), as well as the quality of the different categories/genres of music albums based on a measure of the *attraction index* and the *fidelity measure* of music files vis à vis their respective music genres.

References

1. Kohonen, T.: Self-Organizing Maps. Springer, Berlin (2001)
2. Lagus, K., Honkela, T., Kaski, S., Kohonen, T.: Websom for Textual Data Mining. Artificial Intelligence Review 13(5-6), 345–364 (1999)
3. Ultsch, A., Siemon, H.: Kohonen's self organizing feature maps for exploratory data analysis. In: Proceedings of the International Neural Network Conference, Dordrecht, Netherlands, pp. 305–308 (1990)
4. Pampalk, E., Rauber, A., Merkl, D.: Content-based organization and visualization of music archives. In: Proceedings of the tenth ACM international conference on Multimedia, Juan-les-Pins, France, pp. 570–579 (2002)
5. Tzanetakis, G., Benning, M., Ness, S., Minifie, D., Livingston, N.: Assistive music browsing using self-organizing maps. In: Proceedings of the 2nd international conference on Pervasive Technologies Related to Assistive Environments. Corfu, Greece (2009)
6. Knees, P., Schedl, M., Pohle, T., Widmer, G.: An innovative three-dimensional user-interface for exploring music collections enriched. In: Proceedings of the 14th annual ACM international conference on Multimedia, Santa Barbara, CA, USA, pp. 17–24 (2006)

7. Manalili, S.: i3DMO: an Interactive 3D Music Organizer, MS Thesis, College of Computer Studies, De La Salle University, Manila, Philippines (2010)
8. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations 11(1) (2009)
9. Moerchen, F., Ultsch, A., Thies, M., Loehken, I., Noecker, M., Stamm, C., Efthymiou, N.: Kuemmerer. M.: MusicMiner: Visualizing perceptual distances of music as topographical maps, Technical Report Dept. of Mathematics and Computer Science, University of Marburg, Germany (2005)
10. McKay, C.: Automatic music classification with jMIR. Ph.D. Thesis. McGill University, Canada (2010)
11. Villmann, T., Der, R., Herrmann, M., Martinetz, T.M.: Topology preservation in self-organizing feature maps – exact definition and measurement. IEEE Trans. Neural Networks 8(2), 256–266 (1997)
12. 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)
13. Azcarraga, A., Caw, A.: Enhancing SOM digital music archives using Scatter-Gather. In: Proc. Intl. Joint Conference on Neural Networks, Hong Kong, pp. 1833–1839 (2008)

A Novel Bioinformatics Strategy to Predict Directional Changes of Influenza A Virus Genome Sequences

Yuki Iwasaki, Kennosuke Wada, Masae Itoh, Toshimichi Ikemura, and Takashi Abe*

Department of Bioscience, Nagahama Institute of Bio-Science and Technology,
Nagahama, Shiga, Japan
takaabe@nagahama-i-bio.ac.jp

Abstract. Influenza A viruses cause a significant threat to public health as highlighted by the recent introduction of the swine-derived H1N1 virus (pandemic H1N1/09) into human populations. Pandemics were primarily initiated by introduction from animal sources and successive adaptation among humans through human-to-human transmission. We established a sequence alignment-free clustering method "BLSOM", which can analyze and compare all influenza A virus genome sequences on one map. Separation according to host animal, subtype and epidemic year could be efficiently visualized. Notably, H1N1/09 strains have oligonucleotide and codon compositions clearly distinct from those of seasonal human flu strains. This enabled us to make inferences about directional changes of H1N1/09 sequences in the near future and to list codons and oligonucleotides with the potential of reduction in H1N1/09 sequences. The strong visualization power of BLSOM also provides surveillance strategies for efficiently detecting potential precursors to pandemic viruses.

Keywords: influenza A virus, self-organizing map, oligonucleotide frequency, codon usage.

1 Introduction

One of the most important issues for informatics studies of virus genomes, particularly of influenza viruses, is the prediction of genome sequence changes that will be hazardous. We have developed a novel informatics strategy to predict a portion of sequence changes of influenza A virus genomes, by focusing on the pandemic H1N1/09 strains [1] as a model case. The phylogenetic analysis based on sequence homology searches is a well-established and an irreplaceably important method for studying genomic sequences. However, it inevitably depends on alignments of sequences, which is potentially error-prone and troublesome especially for distantly related sequences. This difficulty becomes increasingly evident as the number of sequences obtained from a wide range of species, including novel species, increases dramatically because of the remarkable progress of the high-throughput

* Corresponding author.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 198–206, 2011.

DNA sequencing methods. To address the difficulty and complement the sequence homology searches, we here report an alignment-free clustering method that could analyze far more than 100,000 sequences simultaneously, on the basis of Self-Organizing Map (SOM) [2,3]. SOM is known to be a powerful clustering method, which provides an efficient interpretation of complex data using visualization on one plane. We have previously developed a modified SOM (batch-learning SOM: BLSOM), which depends on neither the data-input order nor the initial conditions, for studying oligonucleotide frequencies in vast numbers of genomic sequences [4,5]. When we constructed BLSOMs for oligonucleotide frequencies in fragment sequences (e.g., 10 kb) from wide varieties of species, sequences were self-organized according to species; BLSOMs could recognize and visualize species-specific characteristics of oligonucleotide composition. Here, we have analyzed influenza A viruses, including those of the pandemic H1N1/09 [1,6,7], and developed a widely applicable strategy for predicting directional sequence changes of zoonotic virus genomes.

2 Methods

A total of 43,831 virus sequences analyzed in Fig. 1A were obtained from the DDBJ GIB-V web site (http://gib-v.genes.nig.ac.jp/), and a total of 42,800 sequences derived from 5,350 influenza A virus strains were obtained from the NCBI Influenza Virus Resource (http://www.ncbi.nlm.nih.gov/genomes/FLU/).

BLSOM program was obtained from UNTROD Inc. (y_wada@nagahama-i-bio.ac.jp), witch developed the program under collaboration with our group.

3 Results and Discussion

3.1 BLSOMs Constructed with almost all Available Virus Sequence

To test the clustering powers of BLSOM for large numbers of sequences from a wide variety of virus genomes, we initially constructed BLSOM with tetranucleotide frequencies in all 1-kb fragment sequences (199,067 sequences) from 43,828 virus genomes available from GIB-V[8]. Lattice points that contained sequences from one phylogenetic family were indicated in color, and those that included sequences from more than one family were indicated in black (Fig. 1a). A major portion was colored, showing a major portion of sequences to be self-organized according to phylotype; 86% of lattice points had sequences from one phylotype. Tri-BLSOM, as well as Tri- and Tetra-BLSOMs with 0.5-kb sequences (ca. 400,000 sequences), also showed a clear clustering according to phylotype with a slight reduction (approximately 5% reduction) of the separation (data not shown). Notably, no information in regard to phylotype was given during the BLSOM calculation and half a million sequences could be analyzed.

3.2 BLSOMs Constructed with Oligonucleotide Composition in Influenza a Virus Sequences

We next analyzed influenza A virus sequences available from NCBI[9]. Viruses are inevitably dependent on many host factors for their growth, but have to escape from

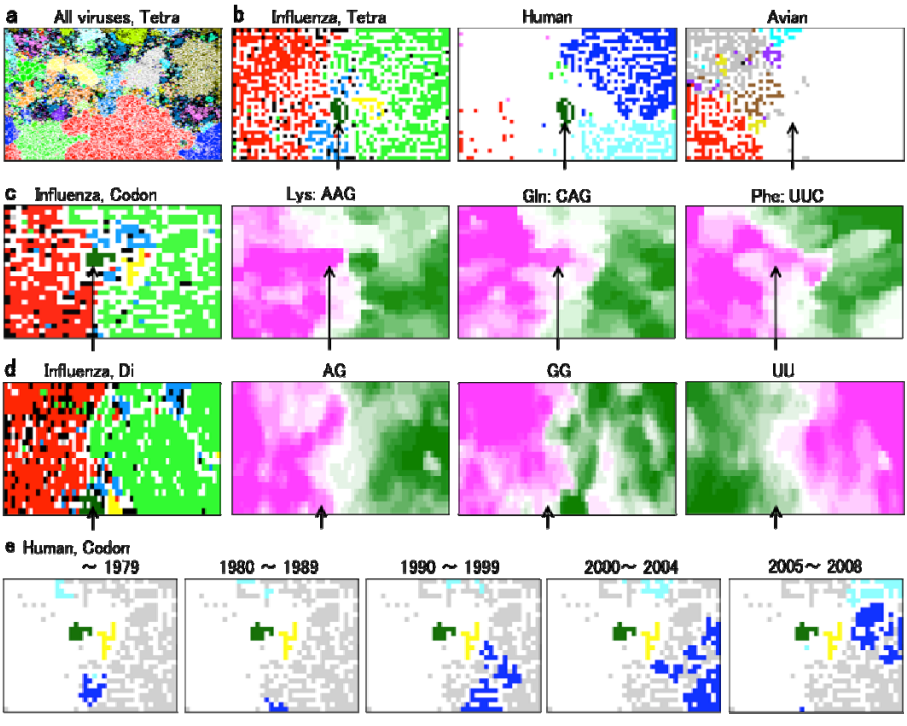


Fig. 1. BLSOMs for virus genome sequences. **(a)** Tetranucleotide BLSOM (Tetra) for 1-kb sequences (199,067 sequences) from 43,828 virus genomes. Lattice points containing sequences from a single phylotype were indicated in a color representing the phylotype. A major portion of the map was colored, showing a major portion of sequences was self-organized according to phylotypes. (**b**) Influenza, Tetra. Lattice points containing sequences from one host were indicated in a color representing the host: **avian strains** (1948 strains), **human H1N1/09 strains** (167 strains), **other human strains** (2788 strains), **swine strains** (249 strains), **equine strains** (68 strains), and **strains from other hosts** (130 strains). Human; human subtype was specified in a color representing the subtype (**H1N1**, **H1N1/09**, **H3N2**, **H5N1**, **others**). Avian; avian subtype was specified in a color representing the subtype (**H5N1**, **H5N2**, **H6N1**, **H7N2**, **H9N2**, **others**). A minor human territory representing **H1N1/09** was indicated with an arrow. **(c)** Influenza, Codon; BLSOM was constructed for synonymous codon usage in 34,376 genes from 4297 strains, and lattice points were specified as described in **b**. Occurrence levels of three codons were indicated with different levels of two colors, **pink (high)** and **green (low)**; intermediate in achromo. **(d)** Influenza, Di; BLSOM was constructed for dinucleotide composition, and lattice points were specified as described in **b**. **(e)** Retrospective time-series changes for human subtype strains on Codon-BLSOM. **H1N1** and **H3N2** strains in the specified time period were shown by these colors, and other human strains were in gray. A zone for **H1N1/09** or **equine** was additionally marked to help to recognize the position in Codon-BLSOM in **c**.

For color picture, refer to http://trna.nagahama-i-bio.ac.jp/WSOM2011/WSOM2011_Fig1.pdf

antiviral host mechanisms such as immunity, interferon and RNA interference[10-12]. To search for possible host-dependent characteristics of virus sequences, di-, tri- or tetranucleotide frequencies within the whole sequence in eight genome segments of 5350 influenza A strains were summed up for each strain, and BLSOM was constructed with the summed frequency for each strain (Tetra-BLSOM in Fig. 1b). Lattice points containing strains isolated from one host species were indicated in a color representing the host and those including strains isolated from more than one host were in black. Without information of host, strains isolated from **avians (red)** or **humans (green)** were clustered (self-organized), forming a large continuous territory. This shows existence of host-specific characteristics of oligonucleotide composition. A **minor human territory** (dark green, arrowed in Fig. 1) appeared, which was separated from the **major human territory** and surrounded by **avian** and **swine (sky blue)** territories. This minor territory arrowed was composed of the pandemic H1N1/09 strains, which have resulted from genetic reassortment between the recently circulating swine H1 viruses in North America and the avian-like swine viruses in Europe[6,7].

To investigate virus subtypes, lattice points that contained human viruses of one subtype on the Tetra-BLSOM were specified with one color representing the subtype ("Human" panel in Fig. 1b). Seasonal human **H1N1 (cyan)** and **H3N2 (blue)** strains were clearly separated from each other. In contrast to the compact minor territory of **H1N1/09 (dark green)**, human **H5N1 (red)** strains were rather scattered within the avian territory (achromic in the "Human" panel). This reflects that most of human H5N1 strains were the result of direct infections from avians to humans[11], and therefore, they should have characteristics of avian viruses. We next marked lattice points containing sequences from one avian subtype by one color ("Avian" panel in Fig. 1b). Some avian strains (e.g., H5N1 isolated in Russia, H5N2 in Minnesota, H6N1 in Taiwan, H7N2 in New York, and H9N2 in Israel) were in a closer proximity to swine and H1N1/09 territories than human H5N1 strains already known (data not shown). These strains had oligonucleotide compositions more similar to those of human and/or swine viruses than the known human H5N1 strains. This similarity at a strain level is of particular interest with regard to infection powers in swine and human populations, if these strains invade swine and/or human populations. A very minor portion of avian strains were located within swine territories, representing strains resulting from direct infection from swine. BLSOM could effectively visualize these strains directly introduced from other hosts, even analyzing large numbers of strains.

3.3 BLSOMs Constructed with Codon Composition

Synonymous codon choice sensitively reflects constraints imposed on genome sequences and thus provides a sensitive probe for searching for molecular mechanisms responsible for the constraints; e.g., codon third position G+C% and tRNA composition in microorganisms[13-16]. We previously found that BLSOM efficiently detects species-specific codon-choice patterns of microorganisms, resulting in self-organization of genes according to species[17]. Furthermore, in genes horizontally transferred relatively recently, codon choice reflected primarily that of the donor, but not the recipient genome. We constructed BLSOM for codon usage in

influenza A virus genes (Fig. 1c). **Human (green)** and **avian (red)** territories were clearly separated from each other. Notably, **human H1N1/09 strains** (dark green, arrowed) were again separated from the major human territory and surrounded by **avian** and **swine (sky blue)** territories. The Codon-choice pattern of newly invading viruses should be close to that of the original host viruses, at least for a period immediately after the invasion. Because viruses depend on many cellular factors, codon choice will most likely shift during infection cycles among humans towards the pattern of seasonal human viruses. If so, the direction of sequence changes of H1N1/09 over time especially in the near future is predictable, so far as judged from codon usage and possibly from oligonucleotide frequency.

3.4 Codons and Oligonucleotides Diagnostic for Host-Specific Separation

BLSOM provides a powerful ability for visualizing diagnostic codons or oligonucleotides that contribute to self-organization of sequences according to host. The frequency of each codon at each lattice point was calculated, sorted according to the frequency, and represented at different levels in colors[17]. Transitions between the **high (pink)** and **low (dark green)** levels often coincided with host territory borders, and examples of codons diagnostic for host separation were presented (Fig. 1c). When we focus on all diagnostic cases, which were listed in Table 1, one tendency was observed; G- or C-ending codons were more favorable in the avian territory than the human. The G+C% effect was most apparent in two-codon boxes, witch are composed of two synonymous codons. This was also observed for many codons in four- or six-codon boxes, but there were exceptional cases, such as GCA and UUG (Table 1), indicating the presence of other constrains.

Table 1. Preferred codons and oligonucleotides in avian or human viruses

	Preferred in avian viruses	Preferred in human viruses
Codon	AAG, CAG, CUC, CUG, GCA, GCG, GTG, UCG, UUC	AAA, ACU, AGA, CAA, CCU, GUU, UCA, UUA, UUG, UUU
Di	AG, CG, CU, GA, GG	AA, UU
Tri	ACG, AGG, CAG, CCA, CGU, GAG, GCA, GCG, GGA, GUG, UCC, UCU	AAA, AUU, UAA, UCA, UUA, UUU
Tetra	ACGC, ACGG, AGAG, AGCG, CCAC, CGAG, CGGA, CGGC, CUUC, GACU, GAGC, GAGG, GCAG, GGAG, UCUU, UGUG, UUCG	AAAA, AAAU, AAGU, AUUA, AUUU, CAAA, CCAU, CUUU, GGCC, GGGG, UGUA, UGUU, UUAA, UUAU, UUCA, UUGU, UUUC, UUUG, UUUU

Notably, for many diagnostic codons, human H1N1/09 had the avian-type preference (Fig. 1c and Table 1). In Table 1, to specify the codon preference in H1N1/09, codons preferred in H1N1/09 by comparison with seasonal human viruses were indicated in red within the column for codons preferred in avian viruses, and codons not preferred in H1N1/09 were specified in green within the column for codons preferred in seasonal human viruses. Adaptation of codon choice to cellular factors and environments (e.g., host body temperature) may be a process for invading viruses to establish continuous infection cycles among humans and to increase viral fitness. We hypothesize the codon choice in H1N1/09 will change towards the pattern commonly found in seasonal human viruses. Removal of unfavorable codons can be attained by not only synonymous, but also nonsynonymous changes, and the rate of

nonsynonymous and thus amino-acid substitution of viral genes is known to be often accelerated around antigenic sites[11,18]. If the unfavorable codons in seasonal viruses witch were judged by codon BLSOM and specified in red in Table 1, are clustered in certain regions in the H1N1/09 genome, probability of sequence change in the regions is thought to be elevated; and this can be tested in the near future because of the high mutation rate of influenza A viruses[18].

In Fig. 1d, dinucleotide BLSOM was presented along with three examples of diagnostic dinucleotides for host-specific separation, and all diagnostic di-, tri- and tetranucleotides for the host-specific separation were summarized in Table 1. Two tendencies were found. 1) G- and C-rich oligonucleotides were more favorable in avians than in humans: G+C% effect. 2) AG and GA dinucleotides were more favorable in avian than in human. Most of diagnostic di- and trinucleotides could be explained by these rules, but there were various exceptional cases for tetranucleotides (e.g., CGCG, GGCC, GGGG preferred in human), indicating the presence of other effects than the two rules. Notably, H1N1/09 strains have characteristics of avian, rather than of human, strains. Oligonucleotides that were preferred in avian and H1N1/09 but not in seasonal human strains and thus have the potential for reduction in H1N1/09 in the near future were specified in bold letters (Table 1). Searches for unfavorable codons and oligonucleotides within antigen binding regions, antiviral drug-binding sites and sites affecting virus virulence will provide valuable information to predict possible H1N1/09 descendant sequences presenting potential hazards.

3.5 Retrospective Time-Series Changes Visualized for Human Viruses

Invader viruses will change their sequences on balance between a stochastic process of mutations and selection pressure derived from various constraints, including those from host. To examine the feasibility of predicting directional sequence changes of invader viruses, retrospective time-series changes in all influenza A virus sequences available may provide useful information, and thus were visualized on Codon-BLSOM (Fig. 1e). **H1N1 (cyan)** or **H3N2 (blue)** strains that were isolated before 1980 were located around the border between the human and avian territories, and no pandemic descendants invaded the avian territory (achromic). If human viruses had changed their sequences solely as stochastic processes, pandemic descendants of some years might deeply invade the avian territory by crossing over the initial pandemic zone. Absence of such invasion indicated a directional pressure during the course of establishment of seasonal strains at least soon after a new pandemic.

3.6 Segments Separately Analyzed

At the onset of a new pandemic, reassortment of virus genome segments in a certain host (e.g., swine) and successive invasion of the new reassortant into the human population were often essential [6,7,11,18]. Therefore, we next analyzed sequences derived from the eight virological segments separately. The length of the shortest segment (Segment 8) is approximately 0.8kb, and therefore, enough clustering power can be expected based on the result in Fig. 1a. Tetra-BLSOMs for eight segments were presented in Fig. 2a. Clear clustering according to host was observed for all segments,

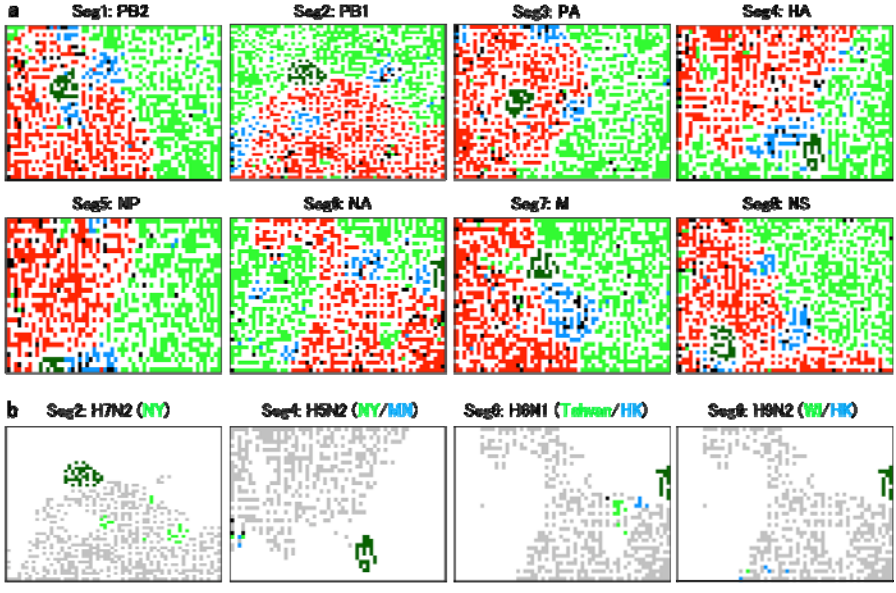


Fig. 2. Tetranucleotide BLSOMs for eight genome segments. **(a)** Gene product name was listed along with the segment number. Lattice points were marked as described in Fig. 1b, and thus a zone for **human H1N1/09** was in **dark green**. **(b)** Examples of subtype of avian virus segments that were in close proximity to human and/or swine territories were marked in **green** or **blue** for specifying the geographical areas where the avian strains were isolated: H7N2 in the segment 2 and H5N2 in the segment 4 isolated in **New York** (NY), H5N2 in the segment 4 isolated in **Minnesota** (MN), H6N1 in the segment 6 isolated in **Taiwan** and **Hon Kong** (HK), and H9N2 in the segment 6 isolated in **Wisconsin** (WI) and **Hon Kong** (HK). Other avian sequences were in gray and sequences from other hosts were in achromo, but a zone for **human H1N1/09** was additionally marked to help to recognize the position in **a**.

For color picture, refer to http://trna.nagahama-i-bio.ac.jp/WSOM2011/WSOM2011_Fig2.pdf

and this was true also for Di-, Tri- and Codon-BLSOMs (data not shown). Segment 2 of **H1N1/09 (dark green)** was in close proximity to the **human (green)** territory, but some other segments (e.g., segments 1 and 3) were within the **avian (red)** territory. This similarity of the oligonucleotide composition of H1N1/09 with that of human, swine or avian viruses was consistent with that found with conventional phylogenetic studies[6,7]. Importantly, more than 5000 sequences could be characterized and visualized on one map, supporting efficient knowledge discovery.

In Fig. 2b, we noted avian strains that were in close proximity to human and/or swine territories along with the geographical information of places where the strains were isolated. Identification of avian- or swine-virus segments whose oligonucleotide and codon compositions were closely related to those of humans should be valuable for predicting candidate strains that may cause pandemics. By summarizing potentially hazardous segments, we can specify avian strains that will come to resemble human or swine strains with reassortment of only a few segments. This type of information should be valuable for gaining new perspectives on systematic

surveillance of viruses presenting potential hazards. BLSOM can efficiently discern such hazardous strains and segments utilizing its strong visualization power. Characterization of host-specific codon- and oligonucleotide-composition may also provide novel information to aid the design of vaccine candidate strains or of strains with high growth rates for supporting high yield vaccine production.

Acknowledgements

This work was supported by the Integrated Database Project and Grant-in-Aid for Scientific Research (C) and for Young Scientists (B) from the Ministry of Education, Culture, Sports, Science and Technology of Japan. The computation was done in part with the Earth Simulator of Japan Agency for Marine-Earth Science and Technology. We wish to thank Dr Kimihito Ito (the Research Center for Zoonosis Control, Hokkaido University) for valuable suggestions and discussions.

References

1. Centers for Disease Control and Prevention: Swine influenza A (H1N1) infection in two children-South California, March- April 2009. Morb Mortal Wkly Rep. 58, 400–402 (2009)
2. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biol. Cybern. 43, 59–69 (1982)
3. Kohonen, T., Oja, E., Simula, O., Visa, A., Kangas, J.: Engineering applications of the self-organizing map. Proc. IEEE 84, 1358–1384 (1996)
4. Abe, T., et al.: Informatics for unveiling hidden genome signatures. Genome Res. 13, 693–702 (2003)
5. Abe, T., Sugawara, H., Kinouchi, M., Kanaya, S., Ikemura, T.: Novel phylogenetic studies of genomic sequence fragments derived from uncultured microbe mixtures in environmental and clinical samples. DNA Res. 12, 281–290 (2005)
6. Smith, G.J., et al.: Origins and evolutionary genomics of the 2009 swine-origin H1N1 influenza A epidemic. Nature 459, 1122–1125 (2009)
7. Garten, R.J., et al.: Antigenic and genetic characteristics of swine-origin 2009 A(H1N1) influenza viruses circulating in humans. Science 325, 197–201 (2009)
8. Hirahata, M., et al.: Genome Information Broker for Viruses. Nucl. Acids Res. 35(Database issue), D339–D342 (2006)
9. Bao, Y.: The influenza virus resource at the National Center for Biotechnology Information. J. Virol. 82, 596–601 (2008)
10. García-Sastre, A.: Inhibition of interferon-mediated antiviral responses by influenza A viruses and other negative-strand RNA viruses. Virology 279, 375–384 (2001)
11. Nelson, M.I., Holmes, E.C.: The evolution of epidemic influenza. Nat. Rev. Genet. 8, 196–205 (2007)
12. Alexey, A., Moelling, K.: Dicer is involved in protection against influenza A virus infection. J. Gen. Virol. 88, 2627–2635 (2007)
13. Ikemura, T.: Correlation between the abundance of Escherichia coli transfer RNAs and the occurrence of the respective codons in its protein genes. J. Mol. Biol. 146, 1–21 (1981)
14. Ikemura, T.: Codon usage and transfer RNA content in unicellular and multicellular organisms. Mol. Biol. Evol. 2, 13–34 (1985)
15. Sharp, P.M., Matassi, G.: Codon usage and genome evolution. Curr. Opin. Gen. Dev. 4, 851–860 (1994)

16. Sueoka, N.: Intrastrand parity rules of DNA base composition and usage biases of synonymous codons. J. Mol. Evol. 40, 318–325 (1995)
17. Kanaya, S., et al.: Analysis of codon usage diversity of bacterial genes with a self-organizing map (SOM): characterization of horizontally transferred genes with emphasis on the *E. coli* O157 genome. Gene 276, 89–99 (2001)
18. Domingo, E., Holland, J.J.: RNA virus mutations and fitness for survival. Annu. Rev. Microbiol. 51, 151–178 (1997)

Impairment and Rehabilitation in Bilingual Aphasia: A SOM-Based Model

Uli Grasemann[1], Chaleece Sandberg[2], Swathi Kiran[2], and Risto Miikkulainen[1]

[1] Department of Computer Science
The University of Texas at Austin, Austin, TX 78712, USA
{uli,risto}@cs.utexas.edu
[2] Department of Speech, Language & Hearing Sciences
Boston University, Boston, MA 02215, USA
challysand@gmail.com, kirans@bu.edu

Abstract. Bilingual aphasia is of increasing interest because a large and growing proportion of the world's population is bilingual. Current clinical research on this topic cannot provide specific recommendations on which language treatment should focus in a bilingual aphasic individual and to what extent cross-language transfer occurs during or after rehabilitation. This paper describes a SOM-based model of the bilingual lexicon, and reports on simulations of impairment and rehabilitation in bilingual aphasia. The goal is to create computational methods that can complement clinical research in developing a better understanding of mechanisms underlying recovery, and that could be used in the future to predict the most beneficial treatment for individual patients.

Keywords: Bilingualism, Aphasia, Lexicon, Neural network, SOM.

1 Introduction

Aphasia is the partial or complete loss of language function due to brain damage, most commonly following a stroke. In bilinguals, aphasia can affect one or both languages, and during rehabilitation and recovery, the two languages can interact in complex ways. Current research on bilingual aphasia has only begun to inform us about these interactions. At the same time, a better understanding of language recovery in bilinguals is badly needed to inform treatment strategies. Decisions like the choice of a target language for treatment affect the outcome in ways that are currently unpredictable, and the optimal treatment strategy is thought to depend on many factors, including how late the second language was learned and the degree of impairment in either language [26].

The problem of choosing the right treatment approach is of considerable practical importance: Over half the world's population today is bi- or multilingual [1,6], making bilingual aphasia at least as common as its monolingual counterpart. Moreover, treatment is most effective during a limited time window, and resources available for treatment are often limited. As the proportion of bilinguals in the world increases, so will the potential benefits of more targeted and effective treatment.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 207–217, 2011.

Current clinical research faces considerable difficulties in providing the necessary insight. Too many factors contribute to the outcome of rehabilitation, including which first and second languages (L1 and L2) the patient speaks, the second-language age of acquisition (AoA), the relative pre-stroke competencies, and the relative impairments in both languages. The large number of possible combinations of these factors, and thus of possible treatment scenarios, makes it impractical to examine treatment effects clinically in a systematic way.

In this situation, computational modeling can be a useful tool to complement and guide clinical research. Neural network-based models of impairment and recovery can be used systematically to simulate treatment scenarios and to predict outcomes. These predictions can then inform clinical research, which in turn provides data to validate the model.

This paper reports on recent progress in work that follows this approach. A model of the bilingual human lexicon based on self-organizing maps is trained and then lesioned in order to model lexical access in bilinguals before and after the onset of aphasia. The model is matched to, and compared with, human subject data collected from a group of aphasic patients. Additionally, a simulation of language-based treatment is developed, and is used to investigate a range of treatment scenarios. The treatment simulation makes testable predictions, and could ultimately be used to simulate treatment for individual patients, and to predict the most beneficial treatment strategy in each specific case.

2 Lexical Access and Bilingual Aphasia

The mental lexicon, i.e. the storage of word forms and their associated meanings, is a major component of language processing. Lexical access is frequently disturbed in aphasia, and naming impairments are especially common, where patients have trouble recalling words or naming objects (anomic aphasia). The mental lexicon of bilinguals is considerably more complex than that of monolinguals, and the way in which multiple language representations can develop, coexist, and interact in the human brain has an important bearing on our understanding of naming impairment in bilingual aphasia.

Current theoretical models of the bilingual lexicon generally agree that bilingual individuals have a shared semantic (or conceptual) system and that there are separate lexical representations of the two languages. However, the models differ on how the lexica interact with the semantic system and with each other.

The concept-mediation model [25] (Fig. 1a), proposes that both the first (L1) and the second-language lexica directly access concepts. In contrast, the word-association model assumes that second-language words (L2) gain access to concepts only through first-language mediation (Fig. 1b). Empirical evidence [18] suggests that the word association model is appropriate for low-proficiency bilinguals and concept mediation model for high-proficiency bilinguals. As an explanation, De Groot [7] proposed the mixed model (Fig. 1c), where the lexica of a bilingual individual are directly connected to each other as well as indirectly (by way of a shared semantic representation). This model was further revised

with asymmetry by Kroll & Stewart [19] (Fig. 1d). The associations from L2 to L1 are assumed to be stronger than those from L1 to L2, and the links between the semantic system and L1 are assumed to be stronger than those between the semantic system and L2.

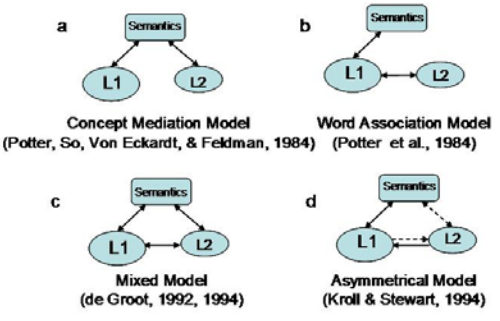


Fig. 1. Theoretical models of the bilingual lexicon. All four theories assume a shared semantic system with language specific representations in L1 and L2. The most recent theory (d) includes connections between all maps, with connections of varying strength depending on the relative dominance of the two languages. This theory is used as the starting point for the computational model.

A second important issue is whether activation of the semantic system spreads to both lexica or only within that of the language being used. The prevailing theory suggests that lexical access is target-language nonspecific [4], although this view is controversial [5]. A third issue is the extent to which proficiency in the two languages and the age at which they are acquired (AoA) affect lexical access. There is evidence that language proficiency, and not AoA, primarily determines the nature of semantic processing [8].

These issues are of central importance to our understanding of the mechanisms underlying the benefits of language-based treatment in aphasia. Specifically, a kind of treatment that can improve word-finding skills in anomic aphasia is naming treatment, where patients are asked to identify objects or activities shown in pictures [2]. In bilinguals, this treatment can occur in either language, and if lexical access in one language indeed co-activates the other language as well, then that language may be able to recover even if it is not actively used in treatment. The literature on such cross-language transfer is sparse, but several case studies suggest that is does occur [15,27], although under what circumstances is currently not known. The computational model described in the next chapter can simulate cross-language transfer, and can potentially help shed light on this and other issues relevant to treatment and recovery.

3 Modeling Approach

Although the physiological structure and location of the lexicon in the brain are still open to some debate, converging evidence from imaging, psycholinguistic,

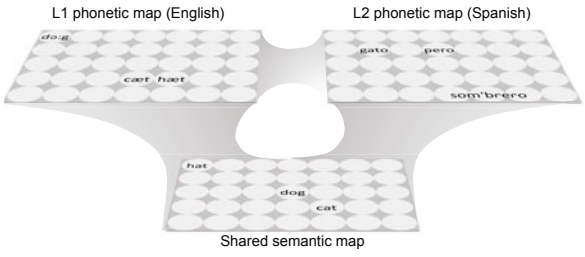


Fig. 2. The DISLEX model is structured after the theoretical model in Fig. 1d. Three SOMs, one each for semantics, L1, and L2, are linked by associations that enable the model to translate between semantic and phonetic symbols, simulating lexical access in bilingual humans.

computational, and lesion studies suggests that the lexicon is laid out as one or several topographic maps, where concepts are organized according to some measure of similarity [10,3,28].

Self-organizing maps (SOMs; [16,17]) model such topographical structures, and are therefore a natural tool to build simulations of the lexicon. SOM models have been developed to understand e.g. how ambiguity is processed by the lexicon [22], how lexical processing breaks down in dyslexia [23], and how the lexicon is acquired during development [21].

The foundation for the bilingual model used in the present work is DISLEX, a computational neural network model initially designed to understand how naming and word recognition take place in a single language [22,23], and later extended to study first language learning [21]. For the purpose of this study, DISLEX was extended to include a second language [24]. The resulting computational model, shown in Fig. 2, reflects the revised model by Kroll and Stewart (1994; Fig. 1d): Its three main components are SOMs, one for word meanings, and one each for the corresponding phonetic symbols in L1 and L2. Each pair of maps is linked by directional associative connections that enable network activation to flow between maps, allowing the model to translate between alternative semantic and phonetic representations of a word.

The organization of the three maps and the associations between them are learned simultaneously. Input symbols are presented to two of the maps at the same time, resulting in activations on both maps. Each individual map adapts to the new input using standard SOM training with a Gaussian neighborhood. Additionally, associative connections between the maps are adapted based on Hebbian learning, i.e. by strengthening those connections that link active units, and normalizing all connections of each unit:

$$a'_{ij} = \frac{a_{ij} + \alpha\, \theta_i \eta_i\, \theta_j \eta_j}{\sum_k (a_{ik} + \alpha\, \theta_i \eta_i\, \theta_k \eta_k)},$$

where a_{ij} is the weight of the associative connection from unit i in one map to unit j in the other map, and η_i is the activation of unit i. The neighborhood function θ_i is the same as for SOM training. As a result of this learning process,

when a word is presented to the semantic map, the resulting activation is propagated via the associative connections to the phonetic maps, and vice versa. In this way, DISLEX can model both comprehension and production in both L1 and L2.

Note that the L1 and L2 maps have direct connections between them as well, which creates a possible alternative path for the flow of activation between the semantic map (S) and either phonetic map. For example, activation may flow S→L1 directly, but also S→L2→L1.

Importantly, such indirect flow of activation between maps can potentially simulate and explain how treatment in one language can benefit the other. For example, if the lexicon is presented with input symbols for S and L1, those maps and the connections between them can be adapted using the method described above. However, in addition, the L2 map is activated indirectly, and that activation can be used to train its associative connections as well. How beneficial this "indirect training" is for L2 may depend on several factors, including the strength and quality of the connections between L1 and L2. The computational experiments reported below will examine this model of cross-language transfer in detail.

The input data used for training the model is based on a list of 300 English nouns gathered for previous studies of naming treatment in aphasia (e.g. [9,14]). The words were translated into Spanish by a native speaker. Semantic representations are vectors of 261 binary semantic features such as "is a container", or "can be used as a weapon". These features were encoded by hand, and the resulting numerical representations were then used to train the semantic map.

Phonetic representations are based on phonetic transcriptions of English and Spanish words, which were split into spoken syllables, and padded such that the primary stress lined up for all words. The individual phonemes comprising each syllable were represented as a set of phonetic features like height and front-ness for vowels, and place, manner, etc. for consonants [20], similar to the method used in previous work based on DISLEX [23]. Phonetic representations consisted of 120 real-valued features for English and 168 for Spanish.

The semantic and phonetic maps of all models were a grid of 30x40 neurons. All learning rates, both for maps and associations, were set to 0.25. The variance of the Gaussian neighborhood was initially 5, and decreased exponentially with a halflife of 150 training epochs. Training always lasted 1000 epochs; the number of randomly selected English and Spanish words trained during each epoch was controlled by two "exposure" parameters.

Second-language AoA was simulated by starting training for the L2 phonetic map and its associative connections as soon as the neigborhood size fell below a specific threshold. For example, a thresholds of 0.7 resulted in training beginning at epoch 425, and a treshold of 5.0 meant it started at epoch 1.

The resulting models generally have well-organized semantic and phonetic maps. Their naming performance, measured as the percentage of semantic symbols that are translated correctly into their phonetic equivalents, is close to 100% (98% for English, 97% for Spanish) for a wide range of combinations of AoA and

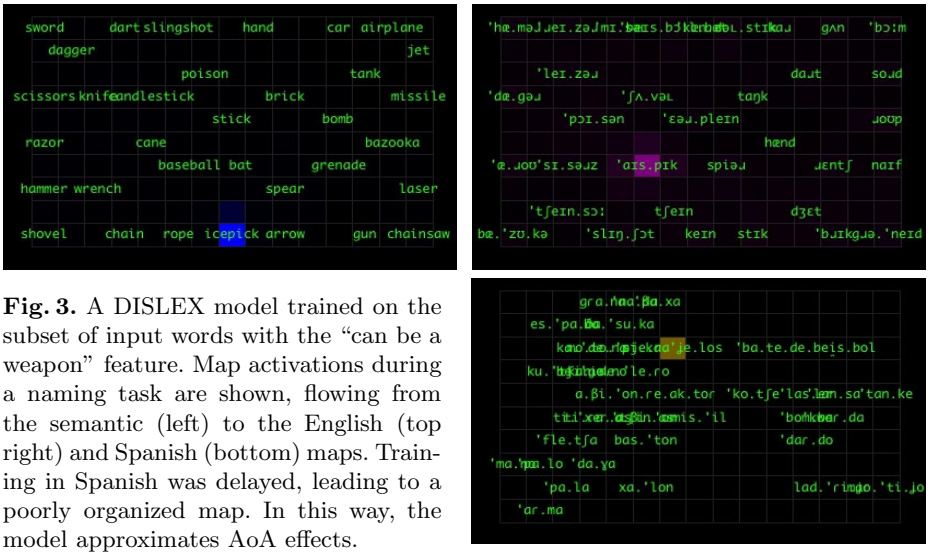


Fig. 3. A DISLEX model trained on the subset of input words with the "can be a weapon" feature. Map activations during a naming task are shown, flowing from the semantic (left) to the English (top right) and Spanish (bottom) maps. Training in Spanish was delayed, leading to a poorly organized map. In this way, the model approximates AoA effects.

exposure. However, as expected, for very low exposure and/or very late AoA, the performance decreases. This is consistent with human language learning, where performance on second-language naming tests tends to be very good, unless the AoA is very late, or exposure to L2 is very limited [11]. As an example, Fig. 3 shows a DISLEX system that was trained on a subset of the input data to make the maps easier to visualize. Semantic and L1 maps reflect semantic and phonetic similarity well. In contrast, the L2 map is poorly organized due to the effect of very late acquisition.

4 Computational Experiments

The method to simulate the effects of different levels of exposure and AoAs outlined above was then used to create a number of DISLEX models that were individually tailored to match a group of bilingual patients suffering from aphasia following a stroke. The first step in creating these models was to train DISLEX to match the patients' premorbid state, including naming performance in both Spanish and English, AoA, and exposure data.

Eighteen of the patients were native Spanish speakers, with English AoA varying from from birth to 35 years. One was a native English speaker (AoA 20 years). Premorbid levels of naming performance, AoA, and relative exposure to Spanish vs. English were collected from all patients, and were used to determine the way in which each patient model was trained. The available patient data on language exposure only specified relative exposure (e.g. 30% Spanish, 70% English); the absolute amount of exposure was therefore adjusted (retaining the correct ratio) such that the resulting model fit naming performance best.

Fig. 4 shows the language performance of the resulting best-fit models for each patient. Bars show the model's performance; small triangles are the target data, i.e. the human pre-stroke English and Spanish performance. In most cases

(∼80%), the model is able to match the premorbid language performance (in addition to AoA and relative exposure) of patients well. Why DISLEX sometimes did not achieve a good fit is not clear in all cases. Interestingly, however, at least in one case (#19), the model identified irregular patient data in this way.

Excluding the patients without a matching DISLEX model, the remaining 16 premorbid models were then used to simulate damage to the lexicon leading to bilingual aphasia. In order to simulate the brain lesion caused by a stroke, the models were damaged by adding varying levels of Gaussian noise to the associative connections between the semantic and phonetic maps.

This model of stroke damage was motivated by several known constraints on the mechanisms by which strokes cause aphasia; for example, word comprehension is often relatively spared in aphasia, which could not be simulated in DISLEX using damage to semantic or phonetic maps. Additionally, recent evidence points to white matter damage in anomic aphasia [12,13].

Fig. 5 shows how increasing levels of noise damage affect the naming performance of the patient models. The bars on the left side of each plot show the same data as in Fig. 4, i.e. the performance of the undamaged model. Moving from left to right in each plot, the damage increases. Red and green lines show the resulting naming performance in English and Spanish respectively. The vertical position of the triangles pointing left show the patients *post-stroke* naming performance in English and Spanish, i.e. the performance the damaged models need to match in each case.

By adjusting the amount of damage for English and Spanish separately, each patient's post-stroke naming performance can always be matched, as shown in

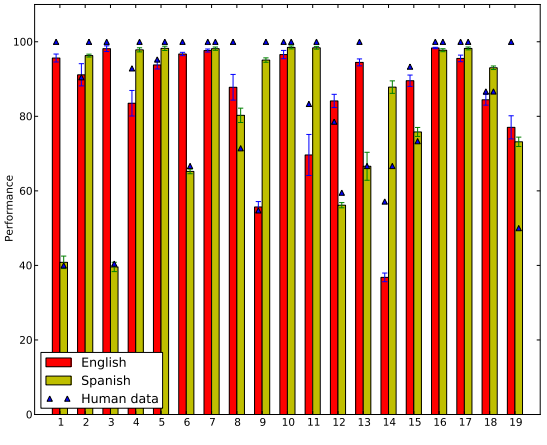


Fig. 4. Modeling the pre-stroke state of bilingual aphasics. The performance of best-fit DISLEX models in English (red/dark bars) and Spanish (yellow/light bars) is compared to patient data (small triangles). The models were trained with the same relative exposure as patients to both languages; AoAs were simulated by variably delaying L2 training. The models are generally able to match the patient data well, providing a basis for simulations of stroke damage.

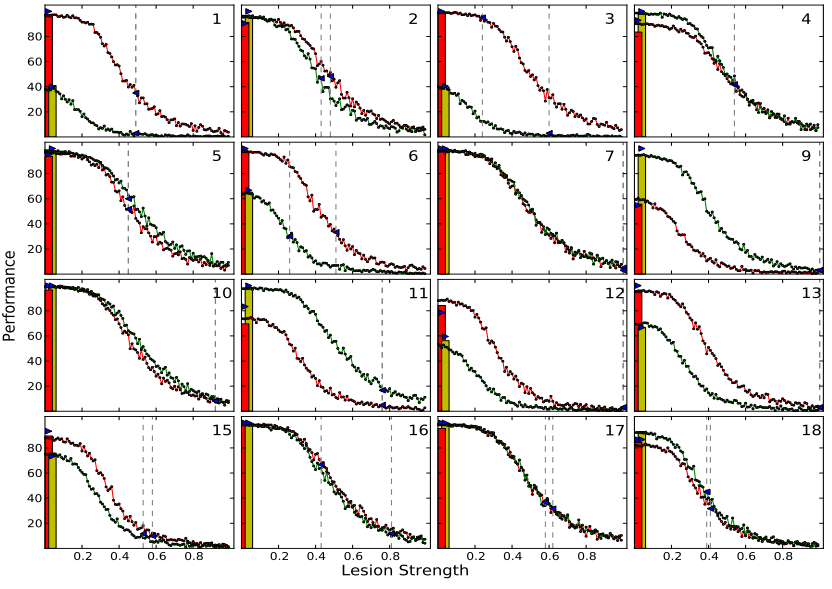


Fig. 5. The effect of damage to the associative connections on naming performance. Patients are numbered as in Fig 4; bars on the left show the same pre-stroke (undamaged) models shown in Figure 4. Moving from left to right in each plot, lesion damage increases, and performance in both languages drops. Triangles pointing left indicate human post-stroke (aphasic) performance, which can in most cases be matched using approximately equal damage to both languages (dashed vertical lines), suggesting that stroke damage is modeled realistically.

the figure. Interestingly, however, in all but three cases (81%), the patient's post-stroke performance can be simulated by damaging English and Spanish connections equally. This is consistent with the type of impairment seen in aphasia patients, which usually, but not always, affects both languages equally. An interesting prediction of the model is that less proficient languages are more vulnerable to damage than highly proficient ones. This is clearly visible e.g. in models #1, 3, and 12.

In the future, these individual models will be used to investigate and predict treatment effects in human patients. As a first step towards this goal, DISLEX simulations for a range of 64 different treatment scenarios were created, which differed in L1 (English/Spanish), AoA (early/late), exposure to L1 and L2 (low/high), damage to L1 and L2 (low/high), and treatment language (English/Spanish). Treatment was simulated by retraining a subset of the original input words in the treatment language. Associative connections of the untreated language were also trained, using indirect activation in the way described in Section 3.

Fig. 6 illustrates the clearest prediction of this model of treatment: If one language is damaged more than the other, training the less damaged language benefits the more damaged language, but not vice versa. Surprisingly, all other

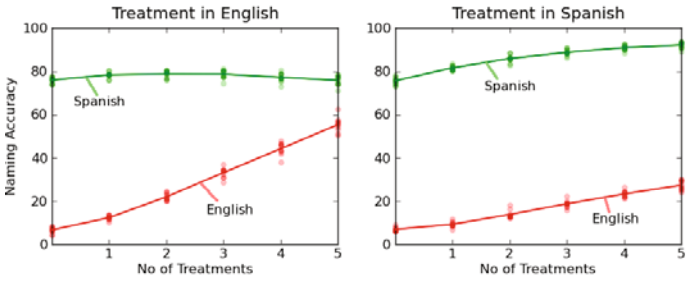


Fig. 6. Effects of treatment language on outcome in the model. In the scenario shown, English is L2 (early AoA), exposure to both languages is low, and English is damaged more than Spanish. The model predicts that treating the less damaged language (in this case Spanish) benefits the more damaged language, but not vice versa.

factors, including relative proficiency and AoA, have little or no effect on cross-language transfer in the model. Moreover, the current model predicts that treating one language should benefit the other in the majority of training scenarios independent of treatment language. However, damage in the model was only applied to semantic→phonetic connections, and damage to other connections, which may be common in humans, may prevent this in many cases. Future work will investigate such additional damage, which will lead to further testable predictions.

5 Discussion and Future Work

In this paper, a bilingual version of DISLEX, a SOM-based model of the human lexicon, was used to simulate impaired lexical access and the effects of language-based treatment in patients with bilingual aphasia. The model was trained and then lesioned to simulate lexical access before and after the onset of aphasia. Human subject data collected from real aphasic patients were used to demonstrate that the model can account for AoA and exposure effects, and that brain damage underlying aphasia can be simulated with a simple noise lesion. Additionally, a simulation of language-based treatment was applied to a range of possible treatment scenarios, and used to predict how the choice of treatment language affects the outcome.

The model makes sevaral testable predictions. First, the effects of noise damage on naming accuracy suggest that in most cases, the brain damage underlying aphasia, not just the impairment, is close to equal for both languages. Second, it predicts that the weaker language, whether it is the first or second, is less resistent to damage than the stronger one. Finally, and most interestingly, the treatment model predicts that using the less damaged language for treatment benefits the more damaged one, but not vice versa.

In future work, the treatment simulation will be applied to the individual patient models, and the results will be compared with the real treatment outcomes.

If validated in this way, the model could be used to meaningfully predict the benefits of different treatment approaches, and could ultimately contribute to the development of optimized treatment strategies tailored to individual patients.

References

1. Bhatia, T.K., Ritchie, W.C. (eds.): The Handbook of Bilingualism. Blackwell Publishing, Malden (2005)
2. Boyle, M., Coelho, C.: Application of semantic feature analysis as a treatment for aphasic dysnomia. American Journal of Speech-Language Pathology, 94–98 (1995)
3. Caramazza, A., Hillis, A., Leek, E., Miozzo, M.: The organization of lexical knowledge in the brain: Evidence from category- and modality-specific deficits. In: Hirschfeld, L., Gelman, S. (eds.) Mapping the Mind, Cambridge University Press, Cambridge (1994)
4. Costa, A., Heij, W., Navarette, E.: The dynamics of bilingual lexical access. Bilingualism: Language and Cognition 9, 137–151 (2006)
5. Costa, A., Miozzo, M., Caramazza, A.: Lexical selection in bilinguals: Do words in the bilingual's two lexicons compete for selection? Journal of Memory and Language 43, 365–397 (1999)
6. Crystal, D.: English as a global language. Cambridge University Press, Cambridge (1997)
7. de Groot, A.: Determinants of word translation. Journal of Experimental Psychology: Learning, Memory and Cognition 18, 1001–1018 (1992)
8. Edmonds, L., Kiran, S.: Lexical selection in bilinguals: Do words in the bilingual's two lexicons compete for selection. Aphasiology 18, 567–579 (2004)
9. Edmonds, L.A., Kiran, S.: Effect of semantic naming treatment on crosslinguistic generalization in bilingual aphasia. Journal of Speech Language and Hearing Research 49(4), 729–748 (2006)
10. Farah, M., Wallace, M.: Semantically bounded anomia: Implications for the neural implementation of naming. Neuropsychologia 30, 609–621 (1992)
11. Hernandez, A., Li, P.: Age of acquisition: Its neural and computational mechanisms. Psychological Bulletin 133, 638–650 (2007)
12. Fridriksson, J., et al.: Impaired speech repetition and left parietal lobe damage. The Journal of Neuroscience 30(33), 11057–11061 (2010)
13. Anderson, J.M., et al.: Conduction aphasia and the arcuate fasciculus: A reexamination of the wernickegeschwind model. Brain and Language 70, 1–12 (1999)
14. Kiran, S.: Typicality of inanimate category exemplars in aphasia treatment: Further evidence for semantic complexity. Journal of Speech Language and Hearing Research 51(6), 1550–1568 (2008)
15. Kohnert, K.: Cognitive and cognate-based treatments for bilingual aphasia: a case study. Brain and Language 91(3), 294–302 (2004)
16. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics 43, 59–69 (1982)
17. Kohonen, T.: Self-Organizing Maps. Springer, Berlin (2001)
18. Kroll, J., Curley, J.: Lexical memory in novice bilinguals: The role of concepts in retrieving second language words. In: Practical Aspects of Memory, vol. 2, Wiley, New York (1988)
19. Kroll, J.F., Stewart, E.: Category interference in translation and picture naming: Evidence for asymmetric connections between bilingual memory representations. Journal of Memory and Language 33, 149–174 (1994)

20. Ladefoged, P.: Vowels and consonants: An introduction to the sounds of languages. Blackwells, Oxford (2001)
21. Li, P., Zhao, X., MacWhinney, B.: Dynamic self-organization and early lexical development in children. Cognitive Science 31, 581–612 (2007)
22. Miikkulainen, R.: Subsymbolic Natural Language Processing: An Integrated Model of Scripts, Lexicon, and Memory. MIT Press, Cambridge (1993)
23. Miikkulainen, R.: Dyslexic and category-specific impairments in a self-organizing feature map model of the lexicon. Brain and Language 59, 334–366 (1997)
24. Miikkulainen, R., Kiran, S.: Modeling the bilingual lexicon of an individual subject. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 191–199. Springer, Heidelberg (2009)
25. Potter, M., So, K., von Eckardt, B., Feldman, L.: Lexical and conceptual representation in beginning and proficient bilinguals. Journal of Verbal Learning and Verbal Behavior 23, 23–38 (1984)
26. Roberts, P., Kiran, S.: Assessment and treatment of bilingual aphasia and bilingual anomia. In: Ramos, A. (ed.) Speech and Language Disorders in Bilinguals, pp. 109–131. Nova Science Publishers, New York (2007)
27. Sasanuma, S., Suk Park, H.: Patterns of language deficits in two korean-japanese bilingual aphasic patients - a clinical report. In: Paradis, M. (ed.) Aspects of bilingual aphasia, pp. 111–122. Pergamon, Oxford (1995)
28. Spitzer, M., Kischka, U., Gückel, F., Bellemann, M.E., Kammer, T., Seyyedi, S., Weisbrod, M., Schwartz, A., Brix, G.: Functional magnetic resonance imaging of category-specific cortical activation: Evidence for semantic maps. Cognitive Brain Research 6, 309–319 (1998)

Gaussian Selection Using Self-Organizing Map for Automatic Speech Recognition

Yujun Wang and Hugo Van hamme

ESAT Department, Katholieke Universiteit Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
{yujun.wang,hugo.vanhamme}@esat.kuleuven.be

Abstract. The Self-Organizing Map (SOM) is widely applied for data clustering and visualization. In this paper, it is used to cluster Gaussians within the Hidden Markov Model (HMM) of the acoustic model for automatic speech recognition. The distance metric, neuron updating and map initialization of the SOM are adapted for the clustering of Gaussians. The neurons in the resulting map act as Gaussian clusters, which are used for Gaussian selection in the recognition phase to speed up the recognizer. Experimental results show that the recognition accuracy is kept while the decoding time can be reduced by 70%.

Keywords: SOM, Speech recognition, Gaussian clustering, Gaussian selection.

1 Introduction

The Self-Organizing Map (SOM) is a widely applied neural model for data analysis and especially for clustering. Its algorithms are comprehensively formulated in [1]. The SOM already attracted interests of the researcher in speech recognition as a vector quantization method to classify speech features, e.g. [20] and [21]. Research that used the SOM to cluster the Hidden Markov Models (HMM) in speech recognition is described in [15]. The author directly treated the parameters of the HMMs as the input features to the SOM. In this paper, SOM is used for clustering Gaussians of the acoustic model for automatic speech recognition.

In a HMM based state-of-the-art Large Vocabulary Continuous Speech Recognition (LVCSR) system [6], there are typically over twenty thousand Gaussians. During the decoding phase, i.e. at recognition time, all the Gaussians need to be evaluated given a 39 dimensional observation vector, which is renewed every 10 milliseconds. Hence evaluation of Gaussians is one of the most time-consuming tasks for the recognizer. However, given the observed feature vector, only a small subset of Gaussians dominate the likelihood of the states of the HMM, while the rest are unlikely. To speed up the decoding, vector quantization based Gaussian selection ([7] [10] [11]) was proposed to exclude unlikely Gaussians from evaluation. Here, cluster Gaussians are computed and assigned likelihoods by the decoder. Only the member Gaussians belonging to those likely clusters are evaluated. The clustering method in the previous research is hard K-Means. SOM, as a soft clustering technique, is closely related to

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 218–227, 2011.

K-Means [3]. It is formed of neurons located on a regular, usually low dimensional grid. The neurons are connected to adjacent neurons usually by a Gaussian neighborhood function preserving the topological properties of the input space. The weights of each input training vector (in our case an input Gaussian) for updating the neurons or codebook is determined by the neighborhood function and the map topology. The output neurons then define cluster Gaussians which can be assigned likelihoods, hence indicating if their cluster members are likely to score well in their turn and if spending computer time on their evaluation is useful or not.

Figure 1 shows an example of the unified distance matrix [4] of a 2-dimensional hexagonal SOM of Gaussians. The matrix visualizes the distance between adjacent neurons. A dark coloring corresponds to a large distance. The distance metric between neurons will be explained in the next section. The input Gaussian and the output neurons are expressed in the spectral domain. The connected gray and black cells form the boundaries of distinct regions on the map. Sample spectra of the vowel /ɑ:/, /i:/, /u:/, /ɔ:/, /æ/ in IPA are plotted on the map. While the mapping serves the primary goal of clustering of Gaussians in this paper, it also has a diagnostic value in engineering acoustic models. Furthermore, research such as [19] claims that the SOM provides an interaction to assist speaker to optimize their pronunciation.

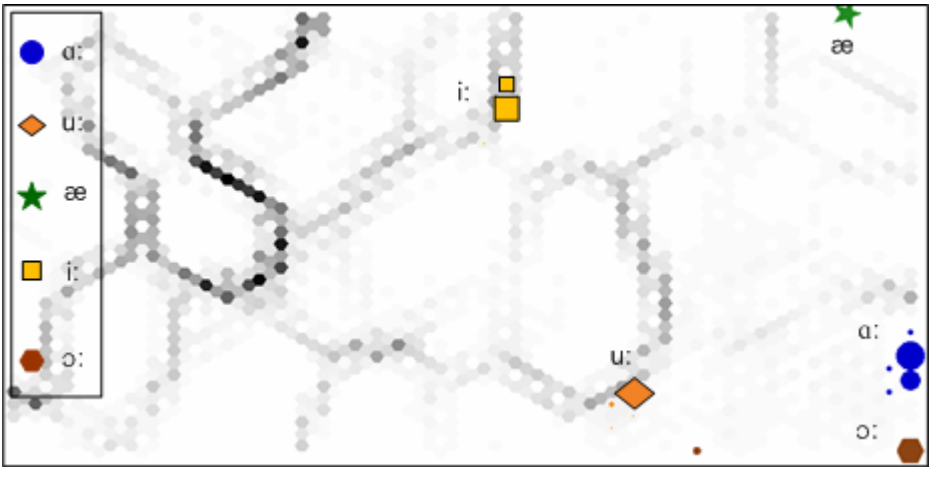


Fig. 1. Unified distance matrix [4] of a SOM plotted by SOM Toolbox [18]. The input Gaussians and output neurons are expressed in the spectral domain. Sample spectra of five vowels are plotted on the map: Given a frame of spectral features, the likelihood of every neuron is calculated and the one with highest likelihood is marked as the best matching unit of the feature vector. The size of the markers indicates the hit rate of the spectral features on a particular neuron.

The rest of the paper is organized as follows: section 2 introduces the SOM training procedure and the adaptations to the algorithms to order Gaussians; section 3 describes our scheme of Gaussian selection using the SOM. Section 4 shows the experimental settings and results. In the last section, conclusions are drawn and future work is proposed.

2 Training a SOM of Gaussians

The SOM in our work has a two-dimensional hexagonal topology and is trained in a batch mode, i.e. updating all the neurons or clusters after presenting all training data. Our training data are the *N* Gaussians of the acoustic model of the speech recognizer, henceforth called *member Gaussians*. Further suppose there are *M* neurons. The process for the batch ordering is:

```
1     Initialize the map to a principal linear sub-
      manifold.
2     For training episode t from 1 to T
        For the n^th member Gaussian, n from 1 to N
2.a       Find its best matching neuron c(n)
2.b       For the m^th neuron, m from 1 to M
           Calculate the neighborhood function g(n,m,t)
2.c   Update the codebook
```

The neighborhood function in step 2.b has the expression:

$$g(n,m,t) = \exp\left(-\frac{\left\| r_{c(n)} - r_m \right\|^2}{2\beta^2(t)}\right) \tag{1}$$

where r_m denotes the coordinates of the m^{th} neuron on the SOM, $\beta(t)$ is the neighborhood radius which decays with the training episode t.

The algorithm within each step in the above pseudo code should be adapted to handle the self organizing of Gaussians. Algorithm adaptations for map initialization in step 1, the distance metric between the input training data (member Gaussians) and the output neurons (cluster Gaussians) in step 2.a and the codebook estimation in step 2.c will be covered in the section 2.1 and 2.2.

2.1 Distance Metric and Neuron Estimation

The Symmetric Kullback-Leibler Divergence (SKLD) is commonly used to measure the distance between a particular input member Gaussian and every neuron in step 2.a. If p and q are multivariate Gaussians, their SKLD is:

$$\begin{aligned} &SKLD(p,q) = \\ &\frac{1}{2}\mathrm{trace}\left\{(\boldsymbol{\Sigma}_p^{-1} + \boldsymbol{\Sigma}_q^{-1})(\boldsymbol{\mu}_p + \boldsymbol{\mu}_q)(\boldsymbol{\mu}_p + \boldsymbol{\mu}_q)' + \boldsymbol{\Sigma}_p\boldsymbol{\Sigma}_q^{-1} + \boldsymbol{\Sigma}_q\boldsymbol{\Sigma}_p^{-1} - 2\mathbf{I}\right\} \end{aligned} \tag{2}$$

The estimation of neurons is based on the approach in [9], where a method for finding the centroid of a set of Gaussians is derived. In their work, the centroid is the Gaussian that minimizes the sum of SKLD to each of the set members. In our work, we extend the results of [9] by minimizing the weighted within-cluster mean SKLD for the m^{th} neuron:

$$\overline{SKLD_m} = \frac{\sum_{n=1}^{N} \mathrm{SKLD}(n,m).\mathrm{g}(n,m,t)}{\sum_{n=1}^{N} \mathrm{g}(n,m,t)} \tag{3}$$

In step 2.b, equation (3) is minimized given the N member Gaussians and their corresponding weights, g(n,m,t), to update one of the M neurons. Hence the formula to re-estimate the mean of the m^{th} neuron is:

$$\overline{\boldsymbol{\mu}}_m^t = \left[\sum_{n=1}^{N} \mathrm{g}(n,m,t)(\boldsymbol{\Sigma}_n^{-1} + \boldsymbol{\Sigma}_m^{-1})\right]^{-1}\left[\sum_{n=1}^{N} \mathrm{g}(n,m,t)(\boldsymbol{\Sigma}_n^{-1} + \boldsymbol{\Sigma}_m^{-1})\boldsymbol{\mu}_n\right] \tag{4}$$

Matrix **B** is constructed to facilitate the re-estimation of the covariance matrices for the neurons:

$$\mathbf{B} = \begin{bmatrix} 0 & \mathbf{A} \\ \mathbf{C} & 0 \end{bmatrix} \tag{5}$$

where

$$\mathbf{A} = \sum_{n=1}^{N} \mathrm{g}(n,m,t)\left[(\boldsymbol{\mu}_n - \overline{\boldsymbol{\mu}}_m^t)(\boldsymbol{\mu}_n - \overline{\boldsymbol{\mu}}_m^t)' + \boldsymbol{\Sigma}_n\right] \tag{6}$$

and

$$\mathbf{C} = \sum_{n=1}^{N} \mathrm{g}(n,m,t)\boldsymbol{\Sigma}_n^{-1} \tag{7}$$

Suppose the member Gaussians are d-dimensional, then B has d positive and d symmetrically negative eigenvalues. Then a $2d$ by d matrix **V** is constructed whose columns are the d eigenvectors corresponding to the positive eigenvalues. **V** is partitioned in its upper halve **U** and lower halve **W**:

$$\mathbf{V} = \begin{bmatrix} \mathbf{U} \\ \mathbf{W} \end{bmatrix} \tag{8}$$

$$\overline{\boldsymbol{\Sigma}}_m^t = \mathbf{U}\mathbf{W}^{-1} \tag{9}$$

It can be seen from equation (4) and (6) that the procedure of estimating the neurons given the weights g(n,m,t) is iterative. The calculation of the mean depends on the previously calculated covariance and vice versa. The exit criterion is the convergence of mean SKLD defined in equation (3). The choice of the initial values is introduced in section 2.2.

2.2 Map Initialization

The purpose of step 1 in the pseudo code is to obtain faster or even better ordering convergence. A global Gaussian (or a single-neuron map) is calculated by averaging

the entire set of member Gaussians. Then the mean and covariance matrix of the global Gaussian are updated using equation (2) to (9) for several iterations till the mean SKLD in equation (3) is converged. Principal Component Analysis (PCA) is applied on the covariance matrix to find the first two principal eigenvectors $\boldsymbol{e}_1$, $\boldsymbol{e}_2$ and eigenvalues, λ_1, λ_2. The square roots of the two principal eigenvalues are used to determine the height h and width l whose product is the size of the code book, M. Then the map is spanned linearly along the directions of the two principal components, i.e. the means of the $(x,y)^{\text{th}}$ neurons are initialized as $(r_x l/\sqrt{\lambda_2})\boldsymbol{e}_2 + (r_y.h/\sqrt{\lambda_1})\boldsymbol{e}_1$, where r_x and r_y are the hexagonal coordinates on the map. The initial values of the covariance matrices are simply assigned with the average values over all member Gaussians' covariance matrices.

3 VQ Based Gaussian Selection Using SOM

Evaluation of Gaussians is one of the computationally intensive tasks in an HMM based LVCSR system. It typically consumes 40% to 70% of the total recognition time without any pruning approaches. Many methods of Gaussian selection emerged to reduce the number of Gaussians to be calculated during decoding. They can be classified as axis indexing based methods and VQ-based methods ([10] [11]). The former quickly locates the observation based on the indices which are already created in the training phase then decide which Gaussian is selected [13] or removed [14]. The latter evaluates the cluster Gaussians and selects only the member Gaussians belonging to those cluster Gaussians having high likelihoods during decoding [10] [11]. Other than Gaussian selection, the VQ based techniques can also be used for sub-space Gaussian clustering [8], which is a different idea to speed up the recognizer.

Our SOM-based Gaussian selection method utilizes the VQ-based methods. Whether a particular member Gaussian is selected is determined by a short list of neurons and the neuron-member Gaussian mapping table, as explained below.

3.1 Constructing the Short List of Neuron Selection

The motivation of Gaussian selection is that only a small portion of Gaussians dominate the likelihoods of the states of the HMM per frame, and are worth evaluating. The SOM consisting of connected neuron Gaussians preserves the topology of the input Gaussian space, i.e. only small regions of neurons on the map dominate the likelihoods of the entire map. In practice, around 80% of the neurons are negligible due to their close-to-zero likelihoods. The posterior probabilities of the remaining 20% neurons are calculated from their likelihoods and sorted. The list is then truncated further to the length such that 95% of the posterior probability mass is included, i.e. the length of the short list is the smallest j such that:

$$\sum_{k=1}^{j} p(G_k \mid O)^{\alpha} > 0.95 \sum_{m=1}^{M*0.2} p(G_m \mid O)^{\alpha} \quad \text{where} \quad p(G_k \mid O) \geq p(G_{k+1} \mid O) \tag{10}$$

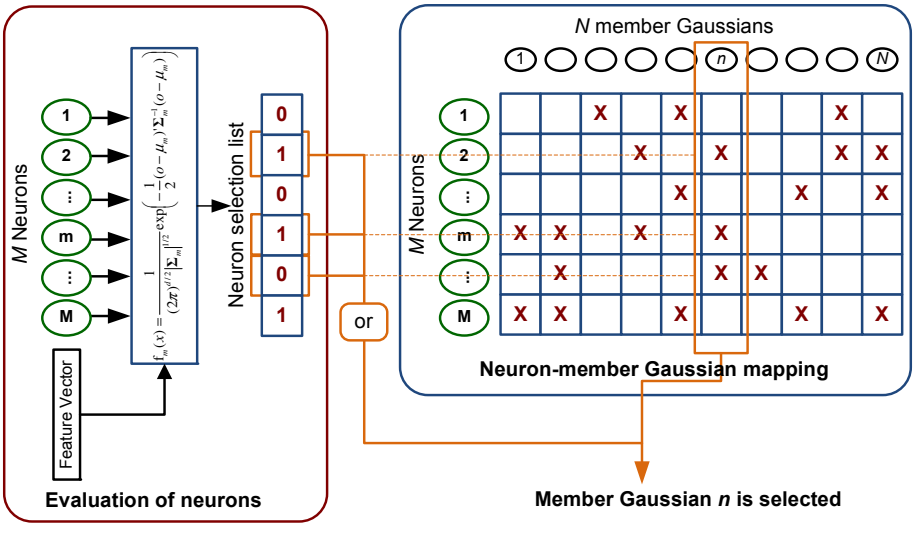


Fig. 2. Decision on the selection of a member Gaussian. During decoding, whether a member Gaussian is selected is determined by the neuron-member Gaussian mapping table and the neuron selection list ("1" indicates that the corresponding Gaussian is selected). The mapping table determines to which neurons the member Gaussian belongs. The recognizer then checks the mapping table and will evaluate the member Gaussian only if any of those neurons is selected.

where the exponent α is introduced to compensate for unmodeled correlations and will indirectly control the number of selected Gaussians. $p(G_k|O)$ denotes the posterior probability of cluster Gaussian k. In figure 2, the neurons labeled by "1" in the neuron selection table appear in the short list.

3.2 Mapping Member Gaussians to Neurons

The neuron-member Gaussian mapping table in figure 2 carries the "softness" of the SOM clustering. Each member Gaussian can be assigned to maximally 6 neurons, namely the first through 6th best matching units on the map. The neurons with SKLD greater than the multiplication of a pruning factor θ and the SKLD of the best matching unit are pruned from the mapping table. On average 3.6 neurons are retained for each member Gaussian. As shown in figure 2, a member Gaussian is selected if at least one of the neurons to which it belongs is activated in the neuron selection table.

Keeping a single neuron Gaussian per member Gaussian in the Gaussian mapping table certainly excludes more member Gaussians from being calculated, but it is found to degrade the recognition accuracy. The result is listed in table 1 as Single Mapping SOM.

4 Experiments

Speech recognition experiments were conducted on the Aurora4 [12] large vocabulary database, which is derived from the WSJ0 Wall Street Journal 5k-word dictation task.

The test set is the clean-condition subset containing 330 utterances from 8 different speakers.

4.1 Experiment Settings

The acoustic model is trained using the clean-condition training set which contains 7138 utterances from 83 speakers, which is equivalent to 14 hours of speech data. All recordings are made with the close talking microphone and no noise is added. The speech spectra, its first and second order derivatives are transformed into 39 dimensional MIDA (Mutual Information based Discriminant Analysis [16]) features and further decorrelated to ensure the model can use diagonal covariance Gaussians. There are 4091 tied states, or senones, and they share 21087 Gaussians. A bigram language model for a 5k-word closed vocabulary is provided by Lincoln Laboratory. The decoding is done with a time-synchronous beam search algorithm and the detail can be found at [17]. The recognizer was launched on a PC installed with Dual Core AMD Opteron Processor 280, whose main frequency is 2.4 GHz. Only one core is activated for the testing.

4.2 The SOM

The two dimensional hexagonal SOM is trained using SOM Toolbox [18] with the 21087 MIDA diagonal covariance Gaussians as input. The map size is 26 by 20, i.e. 520 neuron Gaussians are in the map. The covariance matrices of the output neurons are constrained to be diagonal as well. A rough training phase with only 6 iterations of ordering is carried out first to prevent the map from topological defects [5], then followed by a fine ordering with 24 iterations.

Though the convergence of the typical SOM is not strictly proved in higher-than-one-dimensional case theoretically [5], the mean SKLD is observed monotonously decreased during ordering process in the experiment.

4.3 Experiments Using Other Approaches

Two additional approaches, namely the K-Means and a HMM-based method, are implemented to compare with the SOM.

K-Means uses the following cost function:

$$f_{K-Means} = \sum_{n=1}^{N}\left(\sum_{m=1}^{M} w_{nm} SKLD(n,m) + \gamma \sum_{m=1}^{M} w_{nm} \log\frac{1}{w_{nm}}\right) \tag{11}$$

The weight to update cluster m using member n, w_{nm}, is

$$\begin{aligned} w_{nm} &= \arg\min_{\sum_{m=1}^{M} w_{nm}=1}\left(\sum_{m=1}^{M} w_{nm} SKLD(n,m) + \gamma \sum_{m=1}^{M} w_{nm} \log\frac{1}{w_{nm}}\right) \\ &= \exp\left(-SKLD(n,m)/\gamma\right) / \sum_{i=1}^{M} \exp\left(-SKLD(n,i)/\gamma\right) \end{aligned} \tag{12}$$

Here the softness of the clustering is controlled by γ. The number of clusters M is 520.

An alternative approach to acquire clusters is to train them using the data. A compact HMM containing 520 Gaussians is trained using the same training data as the full HMM containing the member Gaussians. It shares the same structure of tied states as the full model. These 520 Gaussians are used as the cluster Gaussians. An extra Viterbi segmentation pass after the training is carried out to set up the association table between the member Gaussians and the cluster Gaussians: each speech frame is hence assigned to a HMM state. The association count between the dominating member Gaussians of that HMM state and the dominating cluster Gaussian is then incremented by 1. This association table is used as the cluster-member Gaussian mapping table after a proper truncation, i.e. per member Gaussian keeping only the cluster Gaussians with the highest association count (3.6 cluster Gaussians on average).

4.4 Experimental Results

Table 1 shows the experimental results of the Gaussian selection using different approaches of Gaussian clustering, which are compared with the baseline system where none of the Gaussians are pruned during decoding. The percentage of calculated Gaussians is the ratio between the number of calculated Gaussians (including both neurons and selected member Gaussians) and 21087. The baseline is a 2.2×realtime system. We achieve a 0.67×realtime system using the SOM, thus 70% recognition time is saved while the Word Error Rate (WER) is even lower than the base line. The K-Means approach yields lower WER than the SOM, but calculates more Gaussians. The single mapping SOM, where only one neuron Gaussian is kept per member Gaussian in the mapping table, looses accuracy while reducing 1.2ms CPU time per frame, is not preferable. The HMM-based method cannot improve the performance but is slower than the SOM. The associated indexing based approach of Gaussian selection, called Fast Removal of Gaussians (FRoG) [14], of the recognizer [17] is also tested. It only calculated about 5% of the member Gaussians, but required 1.2×realtime.

The Gaussian selection systems based on the SOM, K-Means and data-driven approach also helps to reduce the beam search time because it removes unlikely Gaussians which dominate the unlikely states, hence the confusion among the different search paths is reduced.

Table 1. Word Error Rates and CPU time of SOM Gaussian selection on Aurora 4

	WER	CPU Time		%Gaussian calculated
		Gaussian Calculation (ms/frame)	Beam Search (ms/frame)	
SOM	6.76%	1.8	4.9	7%
Single Mapping SOM	7.02%	1.4	4.1	4%
K-Means	6.65%	2.4	5.1	11%
Data driven	6.82%	2.2	5.2	9%
FRoG	6.82%	The CPU time per frame is12ms in total		≈5%
No Gaussian Selection	6.87%	15.3	6.7	100%

5 Discussion and Future Work

SOM algorithms which are capable of ordering and clustering Gaussians are implemented, tested and proved valid by experimentation. They showed effective to selectively evaluate Gaussians in an automatic speech recognizer, resulting in competitive speed-ups.

The SOM algorithm we used can be interpreted as a soft K-Means with decreasing learning rate. As both K-Means and SOM have a multitude of variants, and the difference of the performance of the SOM and K-Means in our experiments are marginal, it is hard to decide which method is superior. K-Means and SOM can be the substitute of each other. However, SOM provides an insightful visualization of the space of Gaussians, which can facilitate analyzing and engineering the acoustic models for speech recognition.

In our SOM-based Gaussian selection, the evaluation of neurons is inevitable. There are few hundreds of neurons that need to be evaluated additionally over the member Gaussians. However, not all of these evaluations are worthwhile, because only small regions of neurons in the short list are likely. Watanabe et al. [10] presented a tree structure of Gaussians (i.e. clusters of clusters) where the search for likely clusters is narrowed down at the top levels of the tree, hence it can reduce the number of cluster Gaussians to be calculated. A similar idea could be implemented using the hierarchical SOM (also known as the growing SOM or GSOM) [2][21] to quickly locate the likely regions hierarchically. This could be an extension of our work in the future.

Acknowledgments. This research is financed by the MIDAS project of the Nederlandse Taalunie under the STEVIN programme.

References

1. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, Berlin (2001)
2. Rauber, A., Merkl, D., Dittenbach, M.: The Growing Hierarchical Self-Organizing Map: Exploratory Analysis of High-Dimensional Data. IEEE Transactions on Neural Networks 13, 1331–1341 (2002)
3. Sueli, A.M., Joab, O.L.: Comparing SOM neural network with Fuzzy c-means, K-means and traditional hierarchical clustering algorithms. European Journal of Operational Research, 1742–1759 (2006)
4. Ultsch, A., Siemon, H.P.: Kohonen's self organizing feature maps for exploratory data analysis. In: Proceeding of International. Neural Network Conference, Dordrecht, Netherlands, pp. 305–308 (1990)
5. Van Hulle, M.M.: Faithful Representations and Topographic Maps: From Distortion- to Information-Based Self-Organization. John Wiley & Sons, New York (2000)
6. Huang, X., Hon, H.W., Reddy, R.: Spoken Language Processing: A Guide to Theory, Algorithm, and System Development. Prentice-Hall, NJ (2001)
7. Bocchieri, E.: Vector quantization for efficient computation of continuous density likelihoods. In: Proceeding of ICASSP, vol. 2, pp. 692–695 (1993)
8. Bocchieri, E., Mak, B.K.-W.: Subspace Distribution Clustering Hidden Markov Model. IEEE Transactions on Speech and Audio Processing 9, 264–275 (2001)

9. Myrvoll, T.A., Soong, F.K.: Optimal Clustering of Multivariate Normal Distributions Using Divergence and Its Application to HMM Adaptation. In: Proceeding of ICASSP, pp. 552–555 (2003)
10. Watanabe, T., Shinoda, K., Takagi, K., Iso, K.-I.: High Speed Speech Recognition Using Tree-Structured Probability Density Function. In: Proceeding of ICASSP, vol. 1, pp. 556–559 (1995)
11. Shinoda, K., Lee, C.-H.: A structural Bayes approach to speaker adaptation. IEEE Transactions on Speech and Audio Processing 9, 276–287 (2000)
12. Parihar, N., Picone, J.: An Analysis of the Aurora Large Vocabulary Evaluation. In: Proceeding of Eurospeech, pp. 337–340 (2003)
13. Fritsch, J., Rogina, I.: The Bucket Box Intersection (BBI) Algorithm for Fast Approximate Evaluation of Diagonal Mixture Gaussians. In: Proceeding of ICASSP, Atlanta, vol. 2, pp. 273–276 (1996)
14. Demuynck, K.: Extracting, Modeling and Combining Information in Speech Recognition. PhD thesis, K.U.Leuven, ESAT (2001)
15. Du, X.-P., He, P.-L.: The clustering solution of speech recognition models with SOM. In: Wang, J., Yi, Z., Żurada, J.M., Lu, B.-L., Yin, H. (eds.) ISNN 2006. LNCS, vol. 3972, pp. 150–157. Springer, Heidelberg (2006)
16. Duchateau, J., Demuynck, K., Van Compernolle, D., Wambacq, P.: Class definition in discriminant feature analysis. In: Proceeding of European Conference on Speech Communication and Technology, Aalborg, Denmark, pp. 1621–1624 (2001)
17. SPRAAK: Speech Processing, Recognition and Automatic Annotation Kit, http://www.spraak.org/
18. SOM Toolbox, http://www.cis.hut.fi/somtoolbox
19. Wang, X., Xue, L., Yang, D., Han, Z.: Speech Visualization based on Robust Self-organizing Map (RSOM) for the Hearing Impaired. In: Proceeding of International Conference on BioMedical Engineering and Informatics, pp. 506–509 (2008)
20. Sarma, M.P., Sarma, K.K.: Speech Recognition of Assamese Numerals Using Combinations of LPC - Features and Heterogenous ANNs. In: Proceeding of International Conference on BioMedical Engineering and Informatics, pp. 506–509 (2008)
21. Souza Júnior, A.H., Barreto, G.A., Varela, A.T.: A speech recognition system for embedded applications using the SOM and TS-SOM networks. In: Mwasiagi, J.I. (org.) Self Organizing Maps: Applications and Novel Algorithm Design, pp. 97–108. IN-TECH publishing, Viena, Áustria (2010)

Self Organizing Maps in NLP: Exploration of Coreference Feature Space

Andre Burkovski[1], Wiltrud Kessler[1], Gunther Heidemann[1], Hamidreza Kobdani[2], and Hinrich Schütze[2]

[1] Intelligent Systems Group
University of Stuttgart
Universitaetsstr. 38, 70569 Stuttgart, Germany
firstname.lastname@vis.uni-stuttgart.de,
firstname.lastname@vismail.informatik.uni-stuttgart.de
[2] Institute for Natural Language Processing
University of Stuttgart
Azenbergstr. 12, 70174 Stuttgart, Germany
firstname.lastname@ims.uni-stuttgart.de

Abstract. In Natural Language Processing, large annotated data sets are needed to train language models using supervised machine learning methods. To obtain such labeled data sets, time consuming manual annotation is required. To facilitate this process, we propose a SOM-based approach: The SOM sorts the data through unsupervised training, mapping the space of linguistic features to a 2D-grid. The grid visualization is used for efficient interactive labeling of the data clusters. In addition, the interactive SOM visualization allows computational linguists to explore the topology of the feature space and design new features.

Keywords: self organizing maps, coreference resolution, annotation, visualization, natural language processing, feature engineering.

1 Introduction

Supervised machine learning tasks require large amount of training data. Especially in Natural Language Processing (NLP) it is common that people annotate data to create a training data set. Although optimized annotation tools exist, it still is expensive and time consuming. In this paper we present an approach to speed up the annotation process by using self-organizing maps (SOMs) [1].

Besides data annotation, computational linguists require tools which support the understanding of the feature space of the underlying model and data. SOM-based visualizations help the experts to explore the feature space and to understand how the features influence the machine learning methods. The analysis motivate the design of new features which may be better suited to separate the linguistic data.

There are many different data sets which require a speed up of the annotation process and many research areas where exploration of the feature space is

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 228–237, 2011.

valuable. Although we focus on coreference resolution, the proposed approach is suitable for many other data sets. In short, coreference resolution is the task of identifying the entities to which noun phrases in a text refer (see Section 2). Annotators need a full understanding of the text and a basic level of linguistic knowledge in order to annotate a text with coreference information. Thus, especially in the coreference resolution case, the creation of the training data sets is time consuming.

Coreference resolution is an active research area and one of the most challenging topics in NLP. Applications like information retrieval, machine translation, text summarization, and question answering, all benefit from coreference resolution to create better results. Coreference resolution is typically solved with supervised machine learning methods. Current quality metric scores, even with state of the art methods, are still low compared to the desired quality. To improve the quality metric scores, these methods require a large amount of samples for the training.

We combine SOMs with visualizations and interaction techniques to support the annotation task and the exploration of the feature space. We make extensive use of component planes (CPs) to show relevant areas in the SOM. We added interactions, like multiple node selection, interactive component planes alternation, and data visualization for annotation purposes. Interaction enables researchers to explore the feature space. This helps computational linguists to identify areas in the SOM where coreferent data is not clearly separable. The researcher is able to utilize this information for the design of new features which can better solve a specific problem.

SOMs have been employed for visualizations in NLP before. One popular example of SOMs in NLP is WEBSOM [2], where similar documents are clustered together. Another application can be found in the lexical domain [3], where the authors used different SOMs to simulate the language acquisition of children. In our case we focus on linguistic data – plain texts and coreferences. Our visualization and annotation approach is similar to the methods described by Heidemann *et al.* [4] and Moehrmann *et al.* [5], where SOMs are used to cluster and label images. Instead of showing images in the SOM, we use text-based visualization for coreference data.

2 Data and Features

A SOM [1] is an unsupervised machine learning method and easy to visualize. It is a neural network where neurons (or nodes) are connected to each other by a low dimensional topology. The coherence between the low dimensional map and high dimensional data enables a creation of intuitive visualizations. In the following we describe the linguistic data and the features we use to create high dimensional feature vectors. These features are used in the training and visualization of the SOM. We will also show, how we use component planes of features for exploration and annotation purposes.

2.1 Data

The data we are working with are plain texts. Plain texts contain entities or objects (e.g. persons, locations, organizations, things, etc.) and are called noun phrases. Simplifying slightly, a noun phrase is a group of words in a sentence that can be replaced by a noun, another noun phrase, or pronoun. For example *the house* or *my small house that my father built when I was a kid* can be replaced by the pronoun *it*. Coreference means that the two noun phrases of the link can replace one another without changing the meaning of a sentence. The two noun phrases are disreferent, the opposite of coreferent, if by replacing they change the meaning of a sentence. In the domain of coreference resolution such a pair of two noun phrases is called a link. A link has a label that contains information about its co-/disreference status.

In some cases coreference is simple to determine. It is easy to determine that the expressions *([Michael] Jordan)* and *(Jordan)* in the example in Figure 1 are coreferent, because one is a substring of the other. Intuitively two different names (like *Jordan* and *Wizards*) are likely not coreferent. For human readers the coreference status of *(the Washington Wizards)* with the previously mentioned entity *(The Wizards)* is easy to resolve. Although, if the first noun phrase would only consist of *(The Washington team)*, we (as human readers) would need to know that the Wizards team is located in Washington. Otherwise, the text should contain that information elsewhere. To resolve a pronoun like *(he)* we also need the context to decide that it refers to *(Jordan)*.

> ***(The Wizards)*** *may not want* ***([Michael] Jordan)****, but the head of the expansion team says* ***(Jordan)*** *could run operations there, if* ***(he)*** *wants to. [. . .] After playing the last two seasons with* ***(the Washington Wizards)****, Jordan said he expected to return to the front office .*

Fig. 1. A slightly modified example for coreference from the Ontonotes corpus [6]

The research in the area of coreference resolution mainly focuses on machine learning methods. The research on visualization, feature space exploration, or annotation is a very recent activity in the domain. Elango *et al.* [7] present a survey on coreference resolution and Clark *et al.* [8] give a good overview on state of the art coreference problems and solutions.

2.2 Features

Computational linguists use feature extraction methods [9] to process each link and create a high-dimensional feature vector. We extracted many linguistic features inspired by Ng and others [10,11,12]. From the extracted features we construct a feature vector for every link. These feature vectors are used as input for the training of the SOM. In the following we describe only a subset of the features we use for training.

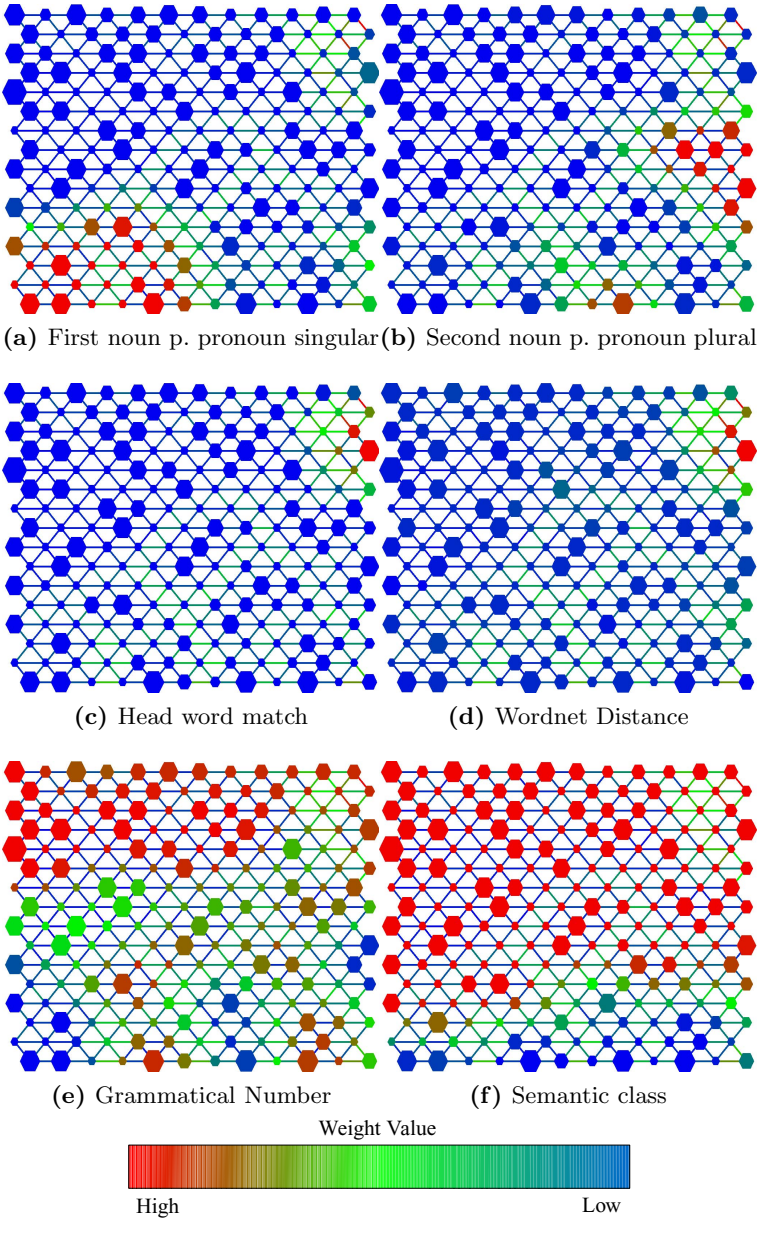


Fig. 2. Component planes for selected features on the graph-based U-matrix visualization. Component planes help the user to identify clusters of links with desired features, like the pronoun feature of a noun phrase in a link (2a and 2b). The user also utilizes component planes to identify strong coreference features (high values in figures 2c and 2d) and strong disreference features (low values in figures 2e and 2f). The size of the nodes represent the number of feature vectors assigned to that node.

The *Head Match* feature uses the head word of a noun phrase. The head is calculated as the last word in the first noun cluster in the noun phrase. Alternatively, it is the last word of the noun phrase if it does not contain a noun cluster [13]. The feature checks if the head words of both noun phrases are identical.

Wordnet [14] is a database for word senses and semantic relations between them. The *Wordnet Distance* feature is the normalized symmetric difference distance [15] of hypernym sets (semantic relations) for both noun phrases, also known as the Jaccard coefficient. All hypernyms of the head words of both noun phrases are retrieved for the calculation. This value represents how the two noun phrases relate semantically.

The *Grammatical Number* and *Semantic Class* features checks if the two noun phrases in a link agree in grammatical number and semantic class respectively. A disagreement is a strong indicator for disreference. The semantic class in our data consists of seven categories: Person, Organization, Location, Facility, Vehicle, Weapon and Geo-Political Entity.

The pronoun features return whether one of the noun phrases of a link is a pronoun and its grammatical number. In Figure 2 we show only two such features: *first noun phrase is a singular pronoun* and *second noun phrase is a plural pronoun*.

3 Interactive Visualization

The SOM provides multiple visualizations from which the user is able to interpret the data distribution. We developed an interactive user interface which utilizes visualizations based on the well-known *unified distance matrix* (U-matrix) [16]. The U-matrix is intended as an intuitive representation of distances between nodes, with high U-matrix values indicating large distances between neighboring nodes in feature space. A common approach to visualize the U-matrix is to display cells for both nodes and edges. In contrast, our visualization treats the U-matrix as a graph. The color of the edges represent the euclidean distance of the neighboring nodes according to the map topology. The nodes represent the actual map units in the topology grid and the size is used to display the number of feature vectors assigned to this node.

In our work we focus on the component planes of the U-Matrix [17]. Component planes color nodes based on weight values of nodes in a single codebook vector component. Component planes allow the visualization of the influence of a single feature on the cluster formation. Through component planes users get a fast overview of which features dominate which regions of the SOM. In Figure 2 high influence of a feature for the projection of data vectors is displayed in red, low influence in blue.

In addition to the U-matrix-based SOM visualizations, we also experimented with a number of alternatives: the P-matrix (a probability density estimation in a map unit), the U*-matrix (a combination of U-matrix and P-matrix), distribution of weight influence (a bar or pie chart for a map units weight values similar to component planes), and PCA projection of the map. We did not find any

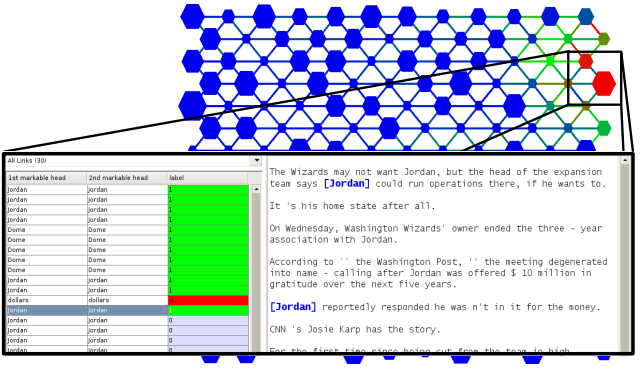


Fig. 3. Using the component plane for the head match feature 2c annotators can select the influential nodes and inspect or annotate the contents of the node in a text-based visualization

advantages of these visualizations regarding annotation and feature engineering compared to the U-matrix.

Interaction is the key element in the coreference annotation task. We created UI elements for the configuration of the SOM. This provides computational linguists with a fast and easy method to try different SOM calculations. Switching interactively between CPs of the U-Matrix helps users to identify interesting areas. These areas contain valuable information which users utilize for annotation. Multiple node selection allows users to inspect the data of several nodes in the SOM and CP. For selected nodes we immediately visualize the coreference data in a different view (Figure 3). We also support easy annotation of nodes. The annotation of a node results in the annotation of data elements in that node.

4 Applications

There are only a few software tools for coreference annotation. These tools use only a text-based visualization for coreference information. The best known general purpose tool for computational linguists is the GATE framework [18]. The coreference annotation module of GATE [19] provides a link visualization based on plain text. Another tool for coreference annotation is MMAX2 [20]. MMAX2 utilizes plain text and HTML for visualization. MMAX2 visualizes coreference information with edges between the noun phrases in the text.

Both tools can utilize extensions for coreference resolution ([19] for GATE and BART [21] for MMAX2). These extensions use a coreference resolution model, which is based on supervised machine learning methods and supports users in the annotation process by highlighting confidence values for coreferences.

In contrast to the GATE framework and BART+MMAX2 annotation tools, we rely on human pattern recognition of coreferences without using supervised machine learning methods. We use interactive SOMs, which visualize feature space of the coreference data and allow a systematic annotation of the data. Such

an annotation approach is valuable in two domains. First, in many non-English languages, which do not have enough labeled data to train supervised methods, our SOM-based method may help to acquire coreference annotations. Second, supervised methods are trained on specific texts, like news articles. Therefore these methods degrade on texts from other domains, like books or reports. Here, SOM is independent of the domain, because of its unsupervised learning method.

Our software enables computational linguists and annotators to gain insight into the coreference feature space and allows several methods for annotation. Interaction is a key property in our approach and we developed interactive methods to assist the users in their tasks.

The main motivation for the development of the tool comes from the wish to better understand the coreference feature space and the need to annotate large data sets. We present three applications for this tool: *feature space exploration*, *feature engineering*, and *annotation*. However, our method is not limited to coreference data. Computational linguists may use the approach for many different tasks where data is described by linguistic feature vectors, like in the areas of information retrieval, parsing, text classification, and machine translation.

4.1 Feature Space Exploration

Utilizing our approach, computational linguists are able to investigate which links are assigned to a node by selecting that node. In addition, with already annotated text and data, the user can directly see which links are coreferent or disreferent. The user can then color-code the nodes of the map with the proportion of dis- and coreferent data.

Usually, data clusters are created by the influence of one or more features. Features responsible for the formation of the cluster can be identified using the component planes. Computational linguists can utilize the coreference information from annotated links to gain insight how well a feature contributes to the separation of the data.

The component plane of one feature, a simple head match, is shown in Figure 2c. High values (red) show a high influence of the head match feature in the upper right corner of the map. Head match is a good indicator for coreference and users may find many coreferences in such a cluster.

Figure 2d shows the component plane of the Wordnet distance feature. Again, high values have a red color and one can conclude, that some links in the Wordnet cluster may be coreferent. Although the Wordnet component plane and head match component plane (Figure 2c) are similar one can find a cluster in the lower right corner of the map (Figure 2d). This cluster contains links with noun phrases that have a high semantic relation, but do not match in head words. Such noun phrases are interesting for text analysis. There are blue areas in the map for which the Wordnet distance value is undefined. This is the case if no Wordnet relations were found for the noun phrase – a common value if the words are both proper names. The same deduction technique also can be used to detect disreferent feature vectors. Depending of the features, such data can be found in clusters where the feature has high or low influence.

Interactive feature space exploration via the component planes enables computational linguists a fast judgment of how well the map has clustered the data.

4.2 Feature Engineering

The feature space exploration gives a good insight into how well the features are suited for the SOM. Computational linguists can identify clusters of nodes where the separation of the data is not clear. With annotated data, the nodes can be color-coded according to the proportion of coreferent links they contain, as mentioned above. The nodes also can display the number of co- and disreferent data.

In some regions expert users may find mixed nodes which contain coreferent links, but have some disreferent links as well. Using component planes, they are able to inspect such difficult cases. This indicates that new features should be developed to better separate coreferent and disreferent links. Computational linguists can inspect these nodes and view the corresponding noun phrases and surrounding text. This allows the experts to understand what the noun phrases have in common and why they were assigned to the same node. E.g. Figures 2a and 2b show clusters where the expert will find links where one of the noun phrases is a pronoun. Such clusters have high weight values in the corresponding component plane. Pronouns are often difficult to resolve, because of the lack of advanced features. The data in the nodes can inspire computational linguists to design new and better features.

Additionally, our tool allows a recalculation of a new SOM with links in nodes selected by expert users. The computational linguists can also change the features for the links and use a subset of available features or add new features. The new SOM calculation sometimes results in a different, better ordering of the links.

4.3 Annotation

The annotators use the U-Matrix, component planes and additional text-based visualizations (Figure 3) for links to label the data. The component planes reveal good clusters of only coreferent (Figures 2c and 2d) and disreferent (Figures 2e and 2f) links. For new, unlabeled data annotators can apply the same SOM and inspect previous clusters but with the new data. After learning where to find good coreferent and good disreferent clusters, annotators are able to efficiently annotate the links in these clusters. The SOM and component planes allow a systematical approach for annotation. From the component planes, annotators can identify regions of nodes with links which have the same or similar properties. E.g. the combination of the component planes in Figures 2a and 2b show regions where one of the noun phrases is a pronoun. The annotator can then use these nodes to focus on the coreference resolution of pronouns. The most valuable links for computational linguists are links which are difficult to resolve. Annotators may learn from computational linguists and feature engineers (as discussed previously) which regions contain nodes with coreferent and

disreferent links. Annotators are then able to label the links in that regions with a high precision, thus creating high quality annotations of difficult links.

5 Conclusion

In this work, we presented an approach which uses self-organizing maps to help computational linguists to gain insight into the feature space. A key feature of the proposed method is the interaction with the SOM and the underlying data. Our approach is general, but we focus on the currently active research area of coreference resolution.

In coreference resolution SOMs allow the computational linguists to identify good indicators for coreferent and disreferent data. Our interactive exploration approach helps to understand how the features influence data separation and create clusters. Our method also helps computational linguists to find nodes and clusters of difficult-to-resolve data. For such data they can design new features which may improve the data separation. Such features may not only improve clustering of links in the SOM, but also enhance the quality scores of other supervised learning methods used by computational linguists.

The efficient annotation of large amounts of data with linguistic information is a major challenge in NLP. Large sets of annotated data are required to train supervised machine learning methods. The quality of the annotation data is extremely important since it will decrease the classification performance of supervised machine learning methods otherwise.

We presented the application of SOMs for annotation purposes. Interaction and visualization plays a major role in this task. We use the U-Matrix, component planes, and interactive methods for annotation of the SOM and the data. SOMs present two key properties for annotations. First, the annotators can focus on the labeling of data with similar properties. The clusters and component planes of the SOM show the annotators regions where they can find data with desired features. Second, the annotators can focus on labeling data for which supervised machine learning methods are not able to easily find a solution. Such data is highly valuable in the training of linguistic models.

References

1. Kohonen, T.: The Self-Organizing Map. Proceedings of the IEEE 78(9), 1464–1480 (1990)
2. Kaski, S., Honkela, T., Lagus, K., Kohonen, T.: WEBSOM - Self-Organizing Maps of Document Collections. Neurocomputing 21, 101–117 (1997)
3. Li, P., Farkas, I., MacWhinney, B.: Early lexical development in a self-organizing neural network. Neural Networks 17(8-9), 1345–1362 (2004)
4. Heidemann, G., Saalbach, A., Ritter, H.: Semi-Automatic Acquisition and Labelling of Image Data Using SOMs. In: European Symposium on Artificial Neural Networks, pp. 503–508 (2003)
5. Moehrmann, J., Bernstein, S., Schlegel, T., Werner, G., Heidemann, G.: Optimizing the Usability of Interfaces for the Interactive Semi-Automatic Labeling of Large Image Data Sets. In: HCI International, Springer, Heidelberg (to appear, 2011)

6. Pradhan, S.S., Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., Weischedel, R.: OntoNotes: A Unified Relational Semantic Representation. In: International Conference on Semantic Computing, pp. 517–526 (2007)
7. Elango, P.: Coreference Resolution: A Survey. In: Technical Report, University of Wisconsin Madison (2005)
8. Clark, J., Gonzàles-Brenes, J.: Coreference: Current Trends and Future Directions. In: Technical Report, Language and Statistics II Literature Review (2008)
9. Kobdani, H., Schütze, H., Burkovski, A., Kessler, W., Heidemann, G.: Relational feature engineering of natural language processing. In: Proceedings Association for Computational Linguistics International Conference on Information and Knowledge Management, pp. 1705–1708 (2010)
10. Ng, V., Cardie, C.: Improving Machine Learning Approaches to Coreference Resolution. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguisticst, pp. 104–111 (2002)
11. Ng, V.: Unsupervised models for coreference resolution. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing 2008, pp. 640–649 (2008)
12. Kobdani, H., Schütze, H.: SUCRE: A Modular System for Coreference Resolution. In: Proceedings of the SemEval 2010, pp. 92–95 (2010)
13. Tjong Kim Sang, E.F.: Memory-Based Shallow Parsing. The Journal of Machine Learning Research, 559–594 (2002)
14. Fellbaum, C.: Wordnet: An Electronic Lexical Database. In: Brandford Books (1998)
15. Yianilos, P.N.: Normalized Forms for Two Common Metrics. In: Report 91-082-9027-1, NEC Research Institute (1991)
16. Ultsch, A., Siemon, H.P.: Kohonen's Self Organizing Feature Maps for Exploratory Data Analysis. In: International Neural Networks Conference, pp. 305–308. Kluwer Academic Press, Paris (1990)
17. Vesanto, J.: SOM-based Data Visualization Methods. Intelligent Data Analysis 3, 111–126 (1999)
18. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A framework and graphical development environment for robust NLP tools and applications. In: Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (2002)
19. Dimitrov, M.: Light-weight Approach to Coreference Resolution for Named Entities in Text. In: Mastersthesis, University of Sofia (2002)
20. Müller, C., Strube, M.: Multi-level annotation of linguistic data with MMAX2. In: Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods, pp. 197–214 (2006)
21. Versley, Y., Ponzetto, S.P., Poesio, M., Eidelman, V., Jern, A., Smith, J., Yang, X., Moschitti, A.: BART: a modular toolkit for coreference resolution. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies, pp. 9–12 (2008)

On Wires and Cables: Content Analysis of WikiLeaks Using Self-Organising Maps

Rudolf Mayer and Andreas Rauber

Institute of Software Technology and Interactive Systems
Vienna University of Technology, Austria

Abstract. The Self-Organising Map has been frequently employed to organise collections of digital documents, especially textual documents. SOMs can be employed to analyse the content and relations between the documents in a collection, providing an intuitive access to large collections.

In this paper, we apply this approach to analysing documents from the Internet platform WikiLeaks. This document collection is interesting for such an analysis for several aspects. For one, the documents contained cover a rather large time-span, thus there should also be an quite divergence in the topics discussed. Further, the documents stem from a magnitude of different sources, thus different styles should be expected. Moreover, the documents have very interesting, previously unpublished content. Finally, while the WikiLeaks website provides a way to browse all documents published by certain meta-data categories such as creation year and origin of the cable, there is no way to access the documents by their content. Thus, the SOM offers a valuable alternative mean to provide access to the content of the collection by their content.

For analysing the document collection, we employ the Java SOMToolbox framework, which provides the user with a wealth of analysis and interaction methods, such as different visualisations, zooming and panning, and automatic labelling on different levels of granularity, to help the user in quickly getting an overview of and navigating in the collection.

1 Introduction

Self-Organising Maps (SOMs) enjoy high popularity in various data analysis applications. Due to its interesting properties, the SOM has been used in several applications to automatically organise documents in a digital library by their content. Examples of text document organisation with the SOM include the WEBSOM project [3], where the contents of a newsgroup collection containing a million of articles were clustered on the map, or the SOMLib digital library system, which has been applied to various collections, such as news texts [1].

Organising the content of textual documents, aided by fitting mechanisms to visualise the structures and to label the map, allows for an intuitive approach to explore the contents of a previously unknown collection. The SOM provides a two-dimensional knowledge map interface to the collection, where users can

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 238–246, 2011.

quickly identify sets of related or distinct documents. Labelling methods function as an aid in identifying the topics of these groups of documents.

We apply this technique to the collection of diplomatic cables published by the Internet Platform WikiLeaks. These documents are comparable to longer newspaper articles, in both length, writing style, and information content. The collection is rather diverse in several aspects. It is diverse in temporal origin, with the single documents being authored in a span of several years to decades. Secondly, the documents were authored by many different authors, albeit from the same profession and country of origin; still, they exhibit different styles of writing. Moreover, the documents cover topics from different regions all over the world.

There are several ways to browse the collection on the WikiLeaks website – by creation year, by origin, and also by tag. While the latter seems most promising to access documents of a certain topic, they are in fact rather unusable for that purpose: there is a total of 590 tags, most of them are non-obvious 3-4 letter acronyms. Only some of the tags contain e.g. names of people. Thus, other means of organising the collection are necessary. The Self-Organising Map provides a valuable means to organise the documents by their content, and present these to the user in a way that they can on the one hand quickly get an overview of the topics in the collection, and one the other hand efficiently navigate through the single documents.

The remainder of the paper is structured as follows. In Section 2 we briefly describe the SOM framework employed. Section 3 then provides details on the WikiLeaks diplomatic cable collection, and the feature extraction method employed. In Section 4 we then present an analysis of the collection's content.

2 SOM Framework

We employ the Java SOMToolbox framework[1], developed at the Vienna University of Technology. Besides the standard SOM learning algorithm, the framework includes several implementations and modifications to the basic algorithm, such as the Growing Grid or the Growing Hierarchical Self-Organising Map (GHSOM). The core of the framework is an application that supports the user in an interactive, exploratory analysis of the data organised by the map training process. This application allows for zooming, panning and selection of single nodes and regions among the map.

To facilitate the visual discovery of structures in the data, such as clusters, a wealth of approximatively 15 visualisations are provided. The visualisations utilised in the experiments later in this paper are now described briefly.

The unified distance matrix, or U-Matrix [9], is one of the earliest, and most popular visualisations of the SOM. It aims at visualising cluster boundaries in the SOM grid. It is calculated as the input-space distance between the model vectors vectors of adjacent map units. These distances are subsequently displayed on the

[1] http://www.ifs.tuwien.ac.at/dm/somtoolbox/

map by colour-coding. Figure 1 depicts an U-Matrix with a grayscale colour map, where lighter shades of grey indicate high distances, and thus cluster boundaries.

The objective of Smoothed Data Histograms [5] (c.f. Figure 2(a)) is to uncover the clusters in the data through an estimation of the probability density of the data on the map. They build on the basic principle of an hit histogram, but they rather rely on a smoothed data histogram which is computed by not only increasing the histogram of the best-matching unit of an input vector, but up to s additional units.

Gradient Fields [6] aims at visualising cluster structures of a SOM, while using a special analogy for the markers utilised in the method. The rationale is that many persons with engineering background are not familiar with colour-coded maps as used e.g. in the U-Matrix, while they are generally familiar with the concept of vector fields. The gradient is displayed on the map with one arrow per unit, with the length and direction of each arrow indicate the location of cluster centres. The arrows at large form a smooth vector field. The arrows are computed based on the model vectors, the map topology, and a neighborhood kernel. Each arrow a_i for a map unit is obtained by computing weighted distances between the units model vector and all other model vectors, which are then aggregated and normalised.

The Thematic Classmap visualisation [4] shows the distribution of meta-data labels or categories attached to the data vectors mapped on the SOM. It colours the map in continuous regions in such a way that the regions reflect the distribution and location of the categories over the map, similar as e.g. a political map does for countries. The method is based on a segmentation of the SOM grid using Voronoi diagrams. A Voronoi diagram of a set of Points $P = p_1, \ldots, p_n$ partitions a plane in exactly n Voronoi regions, each being assigned to one point $p \in P$, so that all the points in a region are closest to p_i. Applied to SOMs, the plane is the visual representation of the map, and the number of regions is equal to the number of units with at least one data item mapped onto. Units with no data items will be split by the algorithm to become parts of other, adjacent regions. The voronoi cells are then coloured according to the categories attached to the data mapped onto the units in each voronoi cells. Details on the colouring can be found in [4].

To assist the user in interpreting the content of the SOM, we automatically generate labels for the map. We employ the LabelSOM method [7], which assigns labels to the units of the SOM describing the features of the data items mapped onto them. The method utilises the *quantisation error* of the vector elements. The quantisation error(q_{i_k}) is the sum of the distances for a feature k between the node's weight vector m_i and the input vectors x_j mapped onto the node. A low quantisation error thus characterises a feature that is similar in all input vectors to the weight vector, which is assumed to be a descriptive feature. To eliminate features that have low quantisation error due to not being present on that node, i.e. having zero values, we require the mean value of the feature to be above a certain threshold.

As the SOM does not generate a partition of the map into separate clusters, a clustering of the units is applied to identify the regions in the map computationally. Of advantage are hierarchical algorithms, which result in a hierarchy of clusters the user can browse through, allowing different levels of granularity; the framework provides the Ward's linkage [2] and several other linkage methods. Having clusters or regions identified, the framework also provides labelling of these entire regions. Making use of the properties of the hierarchical clustering, we can also display two or more different levels of labels, some being more global, some being more local.

Even though labelling the map regions assists the user in quickly getting a coarse overview of the topics, labels can still be ambiguous or not conveying enough information. Thus, the framework also employs Automatic Text Summarisation methods to provide a summary of the contents of the documents of interest, allowing the user to get a deeper insight into the content. The summarisation can either be on single documents, documents from a certain node, a cluster, or from a user-selected set of nodes or documents. Different summarisation algorithms are provided; the user can also specify the desired length of the summary.

3 Collection

The WikiLeaks diplomatic cable collection[2] is composed of United States embassy cables, allegedly "the largest set of confidential documents ever to be released into the public domain". The cables date from the 1960s up until February 2010, and contain confidential communications between 274 embassies in countries throughout the world and the State Department in Washington DC.

The cables are released subsequently, thus currently a subset of 3,319 documents is available. The subset contains cables originating from 165 different sources (embassies, consulates and other representations), and covers mostly the last few years. Details on release year and origin of the dataset are given in Table 1.

It can be noted that a rather large portion of approximatively 12% of the cables were issued by the embassy in Tripoli. A large numbers of documents also originates from Brazil (10.4%, including the cables from the consulates in Sao Paolo and Rio de Janeiro), and Iceland (8.6%). Countries where the USA are involved in military actions, such as Afghanistan or Iraq, have not been published yet in large quantities, thus distinguishing this collection from the Afghan and Iraq war diaries published earlier by WikiLeaks.

To obtain a numeric representation of the document collection for our experiments, we used a bag-of-words indexing approach [8]. From the resulting list of 65,000 tokens, the features for the input vectors were selected according to their document frequency, skipping stop-words, as well as too frequent (in more than 50% of the documents) and too infrequent (in less than 1% of the documents) terms. This resulted in a feature vector of approximately 5,500 dimensions for

[2] http://wikileaks.ch/cablegate.html

Table 1. Cablegate document collection as of February 2011

(a) Documents per year

period	documents
1960s & 1970s	6
1980s	6
2000-2002	11
2003	24
2004	100
2005	167
2006	292
2007	378
2008	684
2009	1270
2010	434

(b) Document origin

Origin	Documents
Libya	406
Brazil	351
Iceland	290
Spain	202
Secretary of State	158
The Netherlands	146
France	122
Russia	93
Egypt	81
Afghanistan	77
UK	75
Pakistan	59
China	58

each document, which formed the basis of the maps subsequently trained. The values of the of the vector are computed using a standard $tf \times idf$ weighting scheme [8], which assigns high weights to terms which appear often in a certain document (high tf value), and infrequent in the rest of the document collection (high idf value), i.e. words that are specific for that document.

4 Experimental Analysis

We trained a map of the size of 35 × 26 nodes, i.e. a total of 910 nodes for the 3,319 text documents. Due to the uncertain legal situation of the Wikileaks documents, we have to refrain from publishing any quotes from the cables, or other details, in this paper.

After inspection of the initial map, it became obvious that the map was dominantly organised along the origin of documents. The reason is that most cables describe events in the country the embassies are located in, thus the names of such countries are too predominantly represented. Thus, for having a more topic-oriented map, we decided to remove the most frequent country names from the feature vector. While this step influences the content of cables that might talk about foreign countries, this side-effect seems acceptable.

The U-Matrix visualisation of this map is depicted in Figure 1. However, on this data set, only a few local boundaries become apparent. The existence of smaller, interconnected clusters is also confirmed by the Smoothed Data Histograms, which visualises density in the map, in Figure 2(a). Figure 2(b) shows the Vector Fields visualisation, where the arrows point towards local cluster centres. These clusters overlap very well with a clustering of the weight (model) vectors of the map, with the clustering for 40 target clusters being superimposed in the same illustration. It can be observed that especially the centre area does not seem to have a clear cluster centre.

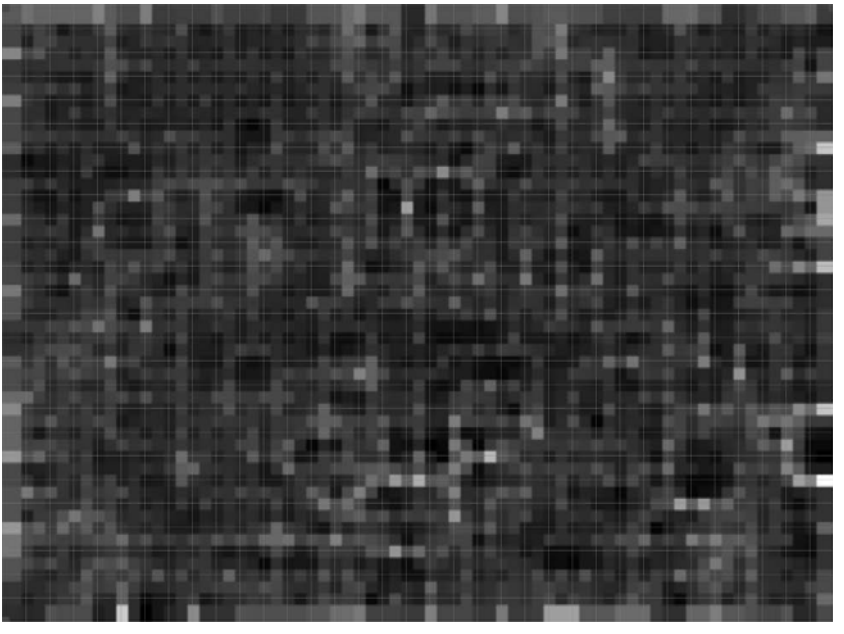
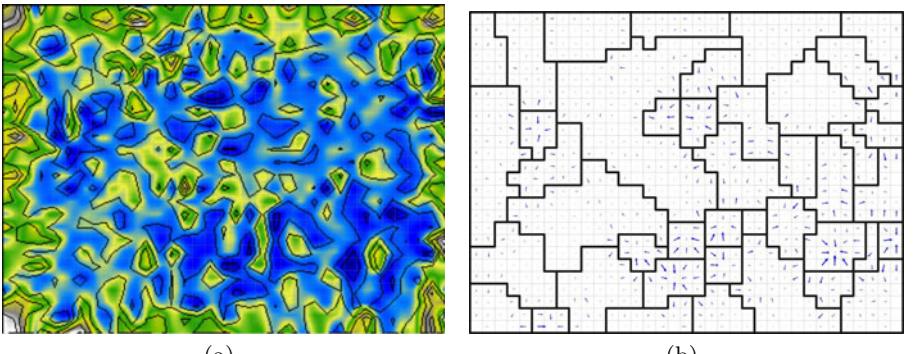


Fig. 1. U-Matrix of the Cablegate SOM



(a) (b)

Fig. 2. Smoothed Data Histograms (a) and Vector Fields (b) visualisations

The Thematic Classmap visualisation depicted in Figure 3 shows the distribution of the origin of the cables. It can be observed that the SOM manages to separate the classes very well, especially on the edges of the map. Overlapping areas are mainly found in the centre of the map, which has previously been identified as an area without a clear cluster centre, and on the upper-left corner. It is often those areas, where the external classification scheme contradicts the topical similarity, which are the most interesting to uncover unexpected relations.

Figure 4 finally shows the Cablegate map with 40 clusters, each of which has two labels assigned, using the LabelSOM method described in Section 2. The display

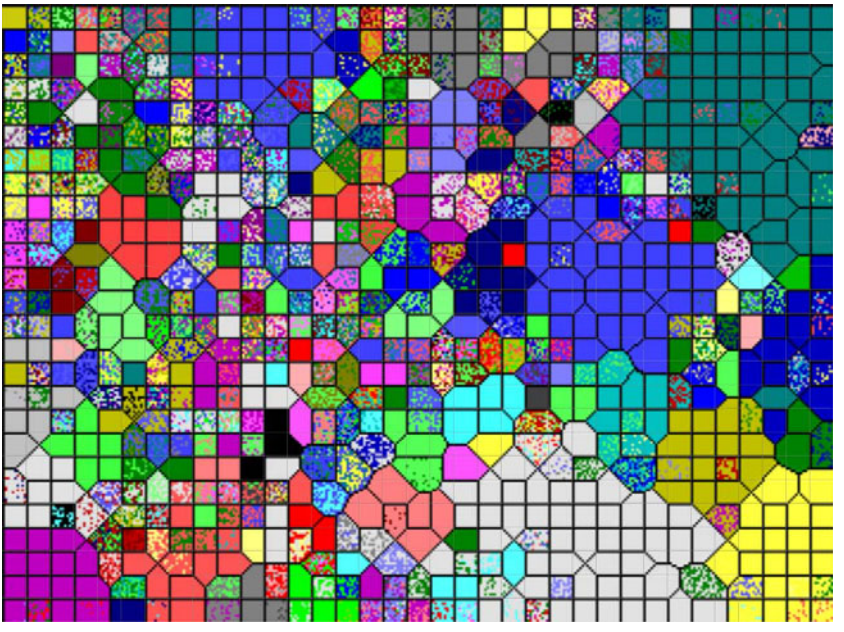


Fig. 3. Thematic Class Map showing the origin of the cables

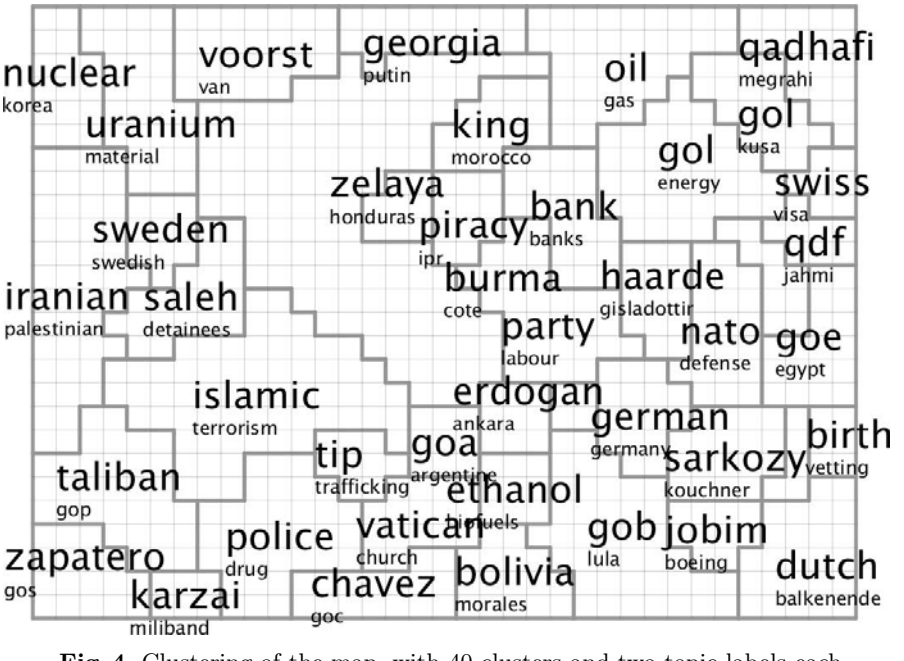


Fig. 4. Clustering of the map, with 40 clusters and two topic labels each

of labels on regions helps to quickly get an overview on the contents of the map, and where to find them. We will describe some of the regions in detail now.

The upper-left corner of the map prominently features diplomatic cables discussing nuclear programs, both of Iran and North Korea, and related issues, such as sanctions and the role of the International Atomic Energy Agency (IAEA). As this is a topic which involves international diplomacy on a large scale, also the sources of origin mentioning the topic are manifold – from the secretary of state and embassies of countries involved into the UN proceedings to cables from the UN representation in Vienna, seat of the IAEA. Topics also dealt with in this area of the map are weapons and the military in general.

The cluster on the central upper edge of the map features reports on the Russian-Georgian war in 2008, and other topics related to Russia. The neighbouring cluster, holding messages mostly about energy such as oil and gas, also features Russian politics, and Russian companies, as well as cables from other countries, such as Venezuela, Nigeria, and Libya. The cluster right next to it, in the top-right corner, then deals with further topics concerning the North-African country ('gol' stands for government of Libya). One topic is for example the diplomatic crisis between the country and Switzerland, which resulted in Switzerland refusing Schengen-Visa.

To the left of this, towards the centre of the map, are two clusters with reports on Iceland, one of them identified by the names of the former prime minister Geir Haarde and the former minister for foreign affairs, Ingibjörg Gísladóttir, who had to step down from office due to the financial crisis hitting Iceland in 2008. The other cluster deals with reports on the Icelandic banks, which suffered intensely from the crisis.

In the lower-centre of the map, a large area is dedicated to topics regarding Brazil. These are dealing with ethanol and other biofuels, which Brazil is a major producer of. Other topics include the defense sector (Nelson Jobim serves as the Minister of Defense). On the left, two clusters deal with other South American issues, namely Bolivian politics, or the crisis between Colombia and Venezuela, reported by cables from both countries. Another topic in that region is the Venezuelan president Hugo Chavez.

Towards the left, certain documents talking about Afghanistan are located. Several of them deal with drugs, while others talk about the involvement of the UK and Spain in the war. Just above that, cables report on Taliban activity, and the situation in Pakistan, as well as cables from India about the attacks in Mumbai, which are linked to terrorists in Pakistan. The region right of that, towards the centre of the map, generally gathers cables from many different sources, all talking about terrorism and criminal activities, without a major topic dominating.

Another interesting arrangement of documents can be found on the left-centre area, which features the previously mentioned documents from Iran and closely to it also Sweden. Many of the documents in the cluster about Sweden deal with the Swedish stance towards the sanctions against Iran due the nuclear programme of the latter.

5 Conclusions

In this paper, we presented a case study for analysing text documents with Self-organising Maps. We employed a framework that provides extensive support in visualisations that uncover structures in the data, and other methods which help to quickly communicate the contents of the collection as a whole, and certain parts in particular, such as labelling on cluster level. With such an analysis tool, the user is able to rapidly get on overview on the interesting areas on the mapping, and gets access to the collection. This approach clearly exceeds the means available on the WikiLeaks website, which comprise of category-based browsing, but lack means to communicate the topics of the collection and exploring the collection by its content.

As the collection of cables is growing on a daily basis, an online version of the map is available at http://www.ifs.tuwien.ac.at/dm/, being regularily updated.

References

1. Dittenbach, M., Rauber, A., Merkl, D.: Business, Culture, Politics, and Sports - How to Find Your Way through a Bulk of News? In: Mayr, H.C., Lazanský, J., Quirchmayr, G., Vogel, P. (eds.) DEXA 2001. LNCS, vol. 2113, pp. 200–220. Springer, Heidelberg (2001)
2. Ward Jr., J.H.: Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association 58(301), 236–244 (1963)
3. Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Paatero, V., Saarela, A.: Organization of a massive document collection. IEEE Transactions on Neural Networks, Special Issue on Neural Networks for Data Mining and Knowledge Discovery 11(3), 574–585 (2000)
4. Mayer, R., Aziz, T.A., Rauber, A.: Visualising Class Distribution on Self-organising Maps. In: de Sá, J.M., Alexandre, L.A., Duch, W., Mandic, D.P. (eds.) ICANN 2007. LNCS, vol. 4669, pp. 359–368. Springer, Heidelberg (2007)
5. Pampalk, E., Rauber, A., Merkl, D.: Using Smoothed Data Histograms for Cluster Visualization in Self-Organizing Maps. In: Dorronsoro, J.R. (ed.) ICANN 2002. LNCS, vol. 2415, pp. 871–876. Springer, Heidelberg (2002)
6. Pölzlbauer, G., Dittenbach, M., Rauber, A.: Advanced visualization of self-organizing maps with vector fields. Neural Networks 19(6–7), 911–922 (2006)
7. Rauber, A., Merkl, D.: Automatic labeling of Self-Organizing Maps for Information Retrieval. Journal of Systems Research and Inf. Systems (JSRIS) 10(10), 23–45 (2001)
8. Salton, G.: Automatic text processing – The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley Longman Publishing Co., Inc., Amsterdam (1989)
9. Ultsch, A., Siemon, H.P.: Kohonen's Self-Organizing Feature Maps for Exploratory Data Analysis. In: Proceedings of the International Neural Network Conference (INNC 1990), pp. 305–308. Kluwer Academic Publishers, Dordrecht (1990)

Media Map: A Multilingual Document Map with a Design Interface

Timo Honkela[1], Jorma Laaksonen[1],
Hannele Törrö[2], and Juhani Tenhunen[2]

[1] Aalto University School of Science,
Department of Information and Computer Science
P.O. Box 15400, FI-00076 Aalto, Finland
[2] Aalto University School of Art and Design,
Department of Media P.O. Box 31000, FI-00076 Aalto, Finland

Abstract. We present a selection of results produced in a project called Media Map. The project aims at developing an intuitive user interface to a library information system containing data on projects and publications. The user interface is a two-dimensional visual display created with the Self-Organizing Map algorithm. The map has been computed using the hierarchical self-organizing map, and a specific graphical design supports the visualization and use of the map interface. In the design, there are specific iconic representations for the projects, publications and persons displayed on the map. The novel aspects in this WEBSOM-type of document map are that the texts on the map are written in different languages, and there are different types of textual objects mapped on the same map. The interlingual mapping is based on applying machine translation on non-English documents. Even when the translation is not fully correct, the approach works well when large enough proportion of relevant terminology has become translated.

Keywords: Self-Organizing Map, text mining, machine translation, library information system, publication map, project map, person map, WEBSOM, PicSOM.

1 Introduction

In this article, we present intermediate results from a project called Media Map in which the use of the Self-Organizing Map (SOM) as an interface to a multifaceted academic library collection is demonstrated. First, we discuss the background for this work and introduce several projects of related work. We continue by presenting the data and methods used and showing the experimental results with the emphasis on describing the basic concept and providing information on the overall system. We do not, however, aim to evaluate each of the subcomponents systematically. This will constitute a future task which includes, for instance, a quantitative analysis of the performance of applying machine translation in content vector creation (see e.g. [1]) as well as qualitative usability and quantitative performance and evaluations (see e.g. [14,19]).

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 247–256, 2011.

1.1 Self-Organizing Map for Information Retrieval

The basic alternatives for information retrieval are (1) searching using keywords or key documents, (2) exploration of the document collection supported by organizing the documents on some manner, and (3) filtering. The keyword search systems can be automated rather easily whereas document collections have traditionally been organized manually. The organization is traditionally based on some (hierarchical) classification scheme, and each document is usually assigned manually to one class.

In the WEBSOM method (see e.g. [2,5,8]), the Self-Organizing Map algorithm [6] is used to map documents onto a two-dimensional grid so that related documents appear close to each other. The WEBSOM automates the process of organizing a document collection according to the contents. It does not only classify the documents, but also creates the classification system based on the overall statistics of the document collection.

The PicSOM content-based visual analysis framework[1] (see e.g. [11,12,17] is based on using relevance feedback with multiple parallel SOMs. It has been developed and used for various types of visual analysis, including image and video retrieval, video segmentation and summarization. It has also served as the implementation platform for the experiments described in this paper.

1.2 Our Approach

In this article, we present a method that follows the basic WEBSOM approach for creating document maps with the following three main novel developments. First, we present a method for creating maps of multilingual document collections in which the documents with similar semantic contents are mapped close to each other regardless of their language. In our experiment, we have English and Finnish documents. Second, the map calculation is conducted with a Tree-Structured Self-Organizing Map (TS-SOM) algorithm [9]. Third, we have developed a design interface for the specific purpose of retrieval and exploration of a database of three different types of entities — *people*, *projects*, and *publications* — in the area of design, media and artistic research.

Our objectives have been three-fold:

- to provide a map as an overview of the contents of an academic research database,
- to design an attractive and informative visualization of the map, and
- facilitative information retrieval from the database regardless of the language used in the documents.

1.3 Related Work

The SOM is widely used as a data mining and visualization method for complex numerical data sets. Application areas include, for instance, process control,

[1] http://www.cis.hut.fi/picsom

economical analysis, and diagnostics in industry and medicine (see e.g. [18]). The SOM has also been used to visualize the views of candidates in municipality elections [4], or the items provided by museum visitors [15]. A variant has been developed in which the shape of the SOM is modified so that it coincides with some well-known shape like the country of Austria [16].

The WEBSOM method for text mining and visualization has been used for various kinds of document collections including conference articles [13], patent abstracts [8], and competence descriptions [3].

2 Data and Methods

2.1 Data

ReseDa[2] is the public web-based research database of the Aalto University School of Art and Design. It is designed to support the school's research, assist the administration of research activities and give them wider visibility. In general, ReseDa provides information on the school's research activities, its expertise, and artistic activities related to art, design, and media.

Table 1 details what kinds of data fields are contained in each of ReseDa's three record types relevant for our experiments. In practice, we started by collecting data on total of 94 projects described by abstracts in either English or Finnish. From the project data we then extracted the identifiers of all involved persons resulting in a set of 101 people. Starting from these people, we finally collected their publications whose abstracts were available in ReseDa in either English of Finnish.

The last type of entities involved in our studies are *units* (such as departments and institutions) with which the projects and publications are associated. In the current data, there were seven units that had more than ten projects and publications. While the data retrieval was one directional, i.e., from projects to people and from people to publications, and from those to the units, we also maintained the reverse mappings in the opposite directions as depicted with solid and dashed lines in Fig. 1. The quantities of the collected data are summarized in Table 2.

Table 1. ReseDa database record types and their contents used in the experiments

Publications	Projects	People
publ-id	proj-id	person-id
publ-title	proj-title	person-name
publ-abstract	proj-abstract	person-publs
publ-people	proj-people	
...	...	...

[2] http://reseda.taik.fi

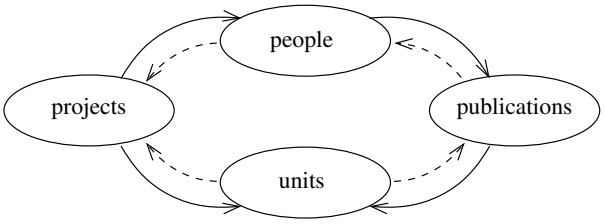


Fig. 1. The link relations between the ReseDa record types

Table 2. The counts of the ReseDa record types used in the experiments

Document category	English	translated
Publications	293	9
Projects	66	28
Persons	101	n/a

2.2 Content Vectors from Multilingual Documents

In our pilot data set, the smallest number of words in the publications is 14 and largest 711. The average number of words is 99.9 with the standard deviation of 90.0. The corresponding numbers for the projects are 27, 867, 117.4 and 156.4. This indicates that the data is skewed, i.e., for many publications there are only a short description.

In order to generate a list of relevant terms, a frequency count of all unigrams, bigrams and trigrams was calculated and sorted in a decreasing order of frequency. Altogether $4934 + 23063 + 32919 = 60916$ term candidates were available. Among these, the words and phrases appearing at least 5 times were considered. Finally, in the manual selection 268 single words such as "adaptive", "advertising", and "aesthetic", 66 bigrams like "augmented reality", "cultural heritage", "design process", and 16 trigrams including "digital cultural heritage", "location based information", and "research based design" were included in the terminology. These 350 terms were used in the encoding of the 497 text documents into document vectors.

The average number of terms in the project descriptions was 21.6 and for publications 16.6. The persons were represented as a concatenation of the publications and projects in which they have been involved. Therefore, the average number of terms for persons, i.e., 44.1 is considerably higher than any of the other two. Unlike persons, each of whom is represented with one text document obtained by concatenation, the departments and other units are represented as collections of their associated projects and publications.

Google Translate[3] was used in the translation. The terms found in the translations was 10.6 when the number of words in translation was 77.5, i.e. lower than for publications or projects. Only a small number (34) of Finnish words were not

[3] http://translate.google.com

translated. Among them, 4 words were misspelled, and the rest 30 were typically inflectional word forms of rare or newly invented words or compounds such as "palvelukehityskin" (even the service development), "julistemaalareiden" (of the poster painters), "innovaatiokoneistosta" (from the innovation machinery) or "kunnollisuudesta" (from decency).

2.3 Document Map Creation

In creating the document maps from the content vectors, we used the hierarchical, Tree-Structured Self-Organizing Map algorithm [9] that is extensively used in the PicSOM content-based image retrieval system [10,11]. The hierarchical structure of TS-SOM drastically reduces the complexity of training large SOMs, thus enabling scalability of the approach into much larger document collections. The computational savings follow from the fact that the algorithm can exploit the hierarchy in finding the best-matching map unit for an input vector.

3 Document Maps

In the following, we describe different kinds of maps produced in the Media Map project. We also present the basic interface design and some design questions when a number of people, projects, publications and organizational units are projected on a map.

3.1 Term Distributions

An elementary analysis of a document map is to study how the different terms are organized on the map. This can be done simply by observing the component-wise distribution of values of the SOM weight vectors. Figure 2 illustrates the distributions of the four most common terms — *design* , *media*, *art* and *learning* — existing in our data. We can see that these terms are quite orthogonal to each other in our material as they appear clearly in non-overlapping map areas. The areas are also mostly contiguous for all of these terms except for *learning*.

3.2 Class Distributions

The aim of the Media Map project has been to place the researchers and their publications and projects on a map in a way that the topology reflects the similarity of the content of the activities. A secondary aim has been to study how well such a mapping also maintains the characteristics of the associated research units. These two questions are addressed in the following.

Figure 3 displays how one researcher's publications ($n = 20$) and projects ($n = 7$) are mapped on the document map. The locations of the documents have been indicated with impulses that have then been low-pass filtered in order to amplify the visibility of spatial topology of the data. The most significant U-matrix [20] distances are illustrated with horizontal and vertical bars. In this researcher's case, Fig. 3 shows that his projects and publications occupy separate,

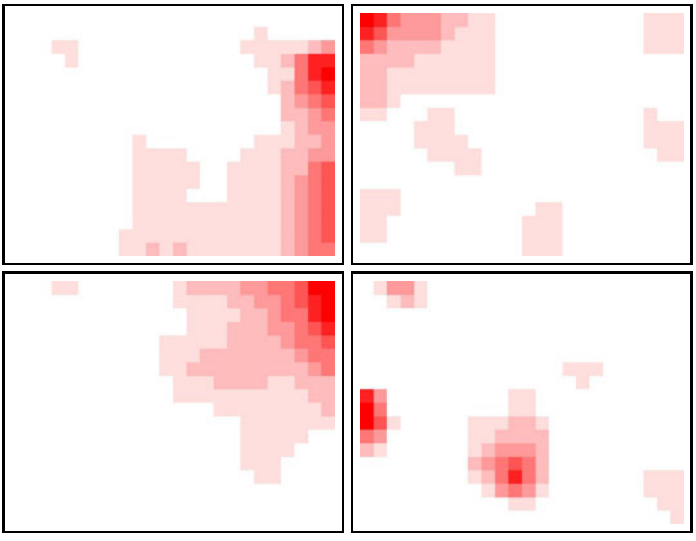


Fig. 2. Occurrences of the words *design* (top left), *media* (top right), *art* (bottom left) and *learning* (bottom right) on the document map

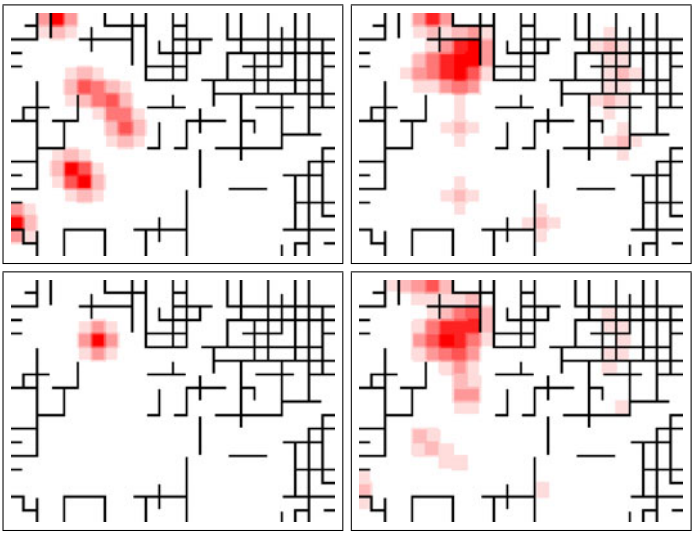


Fig. 3. Distributions of one person's projects (top left) and publications (top right) on the document map. Also the person's own location (bottom left) and the union of the previous three distributions (bottom right) are shown.

but closely situated map areas, and our method maps the researcher himself in a map location close to both areas.

The distribution of publications and projects associated with four research units are shown in Fig. 4. It can be seen that the activities of the units appear

mostly in non-overlapping map areas, but that the units' distributions are not unimodal. The Media department has the largest number of publications and projects and this is reflected in that unit's relatively largest area. Comparing this figure with the two previous ones, some observations can be made. First, as the names of the research units match quite accurately with the most common terms in Fig. 2, also the term and activity distributions are pairwise somewhat similar. Second, the activities of the researcher in Fig. 3 seem to fall inside the activities of the Media department, and this could be expected as the researcher actually is a staff member of that unit.

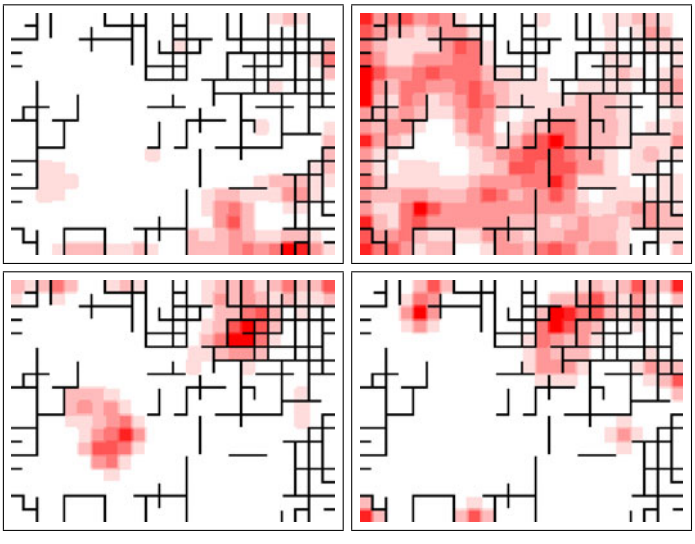


Fig. 4. Distributions of four units' activities on the document map. The units are *Design* (top left), *Media* (top right), *Art* (bottom left) and *Visual culture* (bottom right).

3.3 Map Interface Design

Figure 5 shows an example of the planned map interface designed specifically for the Media Map project. Similarly to Figs. 3 and 4, the location of persons is also in this figure indicated as a specific point on the map whereas the departments and other research units occupy larger areas on the map, respectively. For the persons, an icon is used and other icons exist for publications and projects. As can be seen, the areas of the research units have been planned to be coded with colors that can be overlaid without losing clarity. The figure shows that there exist a slider and control arrows on the left hand side of the map for zooming and panning of the map.

Even though Fig. 5 has still been created by a designer (H.T.), we already have the necessary mechanisms for creating similar illustrations automatically. An open issue is still how zooming into a specific area of the document map could

gradually reveal more and more details of the data. In this manner, the highest-level view would show only information on the research unit level, the mid-level views could show objects and activities on the person level, and only the most detailed view would display data on particular projects and publications. Also different kinds of connections between the entities would be illustrated on the map on different zoom levels.

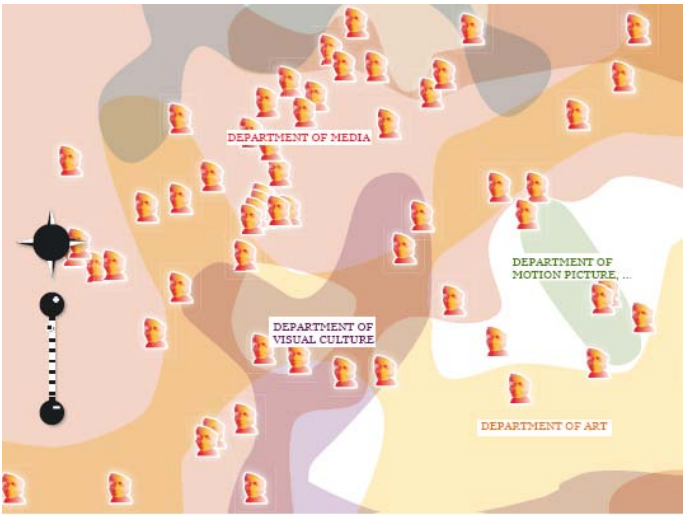


Fig. 5. An example of the interface design of the Media Map

An important design question related to the data to be presented is the way how the items in each category (people, projects, publications and units; see Fig. 1) are visualized. It is natural to represent each individual document as one location on the map (even though it would be possible to find multiple locations for multi-topic documents, see [7]). In our case, each person is represented as a combination of the articles that he or she has written. Our solution is thus to visualize a person as one location on the map. The same solution is in use when projects are considered. However, it would be possible to show all locations where the articles written by a person (or published by a project) are located. This would possibly endanger the readability of the map. On the other hand, it is a natural choice to represent organizational units as the smoothed areas where the articles written by the employees of and the projects hosted by the unit are located.

4 Conclusions and Discussion

We have presented a selection of preliminary results of a project that creates an interface to a library collection in the area of art, design and media research.

Central research and development themes are related to the multilinguality, versatility and interlinked structure of the document collection. There are documents in English and in Finnish concerning projects, publications and people in the database. We have presented a methodology to create document maps in this kind of basic setting and a map interface design that is meant to support information exploration and search.

In the future work, we plan to extend the database to cover all schools of the Aalto University, i.e., schools of Chemical Technology, Economics, Electrical Engineering, Engineering, and Science, in addition to the School of Art and Design involved in the current pilot. This will increase the size of the database considerably because there are more than 300 professors at the Aalto University and the number of people in the academic staff exceeds 2000. Also, we will implement automatic incorporation of the designed user interface elements and facilitate on-line use of the created maps with zooming, panning and clickable links to the original on-line data.

The map interface provides an alternative view to researchers' research areas and their results in contrast with the traditional classification systems that only slowly adapt to the developments in research topics and methods. It is important to note that creative inventions often include introduction of new concepts that do not fit into the existing classification systems. If this aspect is not properly taken into account, and the semantic processing in information infrastructures for research are based on some rigid standards, the innovation activities may even slow down. We believe that the Self-Organizing Map provides a viable alternative and efficient solution for organizational information management.

References

1. Fujii, A., Utiyama, M., Yamamoto, M., Utsuro, T.: Evaluating effects of machine translation accuracy on cross-lingual patent retrieval. In: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, pp. 674–675. ACM Press, New York (2009)
2. Honkela, T., Kaski, S., Lagus, K., Kohonen, T.: Newsgroup exploration with WEBSOM method and browsing interface. Tech. Rep. A32, Helsinki University of Technology, Laboratory of Computer and Information Science, Espoo, Finland (1996)
3. Honkela, T., Nordfors, R., Tuuli, R.: Document maps for competence management. In: Proceedings of the Symposium on Professional Practice in AI. IFIP, pp. 31–39 (2004)
4. Kaipainen, M., Koskenniemi, T., Kerminen, A., Raike, A., Ellonen, A.: Presenting data as similarity clusters instead of lists - data from local politics as an example. In: Proceedings of HCI 2001, pp. 675–679 (2001)
5. Kaski, S., Honkela, T., Lagus, K., Kohonen, T.: WEBSOM—self-organizing maps of document collections. Neurocomputing 21, 101–117 (1998)
6. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (2001)
7. Kohonen, T.: Description of input patterns by linear mixtures of SOM models. In: Proceedings of WSOM 2007, Workshop on Self-Organizing Maps (2007)
8. Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V., Saarela, A.: Self organization of a massive text document collection. In: Kohonen Maps, pp. 171–182. Elsevier, Amsterdam (1999)

9. Koikkalainen, P., Oja, E.: Self-organizing hierarchical feature maps. In: Proc. IJCNN 1990, Int. Joint Conf. on Neural Networks, vol. II, pp. 279–285. IEEE Service Center, Piscataway (1990)
10. Laaksonen, J., Koskela, M., Oja, E.: Application of tree structured self-organizing maps in content-based image retrieval. In: Ninth International Conference on Artificial Neural Networks (ICANN 1999), Edinburgh, UK, September 1999, pp. 174–179 (1999)
11. Laaksonen, J., Koskela, M., Oja, E.: Class distributions on SOM surfaces for feature extraction and object retrieval. Neural Networks 17(8-9), 1121–1133 (2004)
12. Laaksonen, J., Koskela, M., Sjöberg, M., Viitaniemi, V., Muurinen, H.: Video summarization with SOMs. In: Proceedings of the 6th Int. Workshop on Self-Organizing Maps (WSOM 2007), Bielefeld, Germany (2007), http://dx.doi.org/10.2390/biecoll-wsom2007-143
13. Lagus, K.: Map of WSOM 1997 abstracts—alternative index. In: Proceedings of WSOM 1997, Workshop on Self-Organizing Maps, June 4-6, 1997, pp. 368–372. Helsinki University of Technology, Neural Networks Research Centre, Espoo, Finland (1997)
14. Lagus, K.: Text Mining with the WEBSOM. Ph.D. thesis, Helsinki University of Technology (2000)
15. Legrady, G., Honkela, T.: Pockets full of memories: an interactive museum installation. Visual Communication 1(2), 163–169 (2002)
16. Mayer, R., Merkl, D., Rauber, A.: Mnemonic soms: Recognizable shapes for self-organizing maps. In: Proceedings of the Fifth International Workshop on Self-Organizing Maps (WSOM 2005), pp. 131–138 (2005)
17. Oja, E., Laaksonen, J., Koskela, M., Brandt, S.: Self-organizing maps for content-based image retrieval. In: Oja, E., Kaski, S. (eds.) Kohonen Maps, pp. 349–362. Elsevier, Amsterdam (1999)
18. Pöllä, M., Honkela, T., Kohonen, T.: Bibliography of Self-Organizing Map (SOM) papers: 2002–2005 addendum. Tech. Rep. TKK-ICS-R23, Aalto University School of Science and Technology, Department of Information and Computer Science, Espoo (December 2009)
19. Saarikoski, J., Laurikkala, J., Järvelin, K., Juhola, M.: A study of the use of self-organising maps in information retrieval. Journal of Documentation 65(2), 304–322 (2009)
20. Ultsch, A., Siemon, H.P.: Kohonen's self organizing feature maps for exploratory data analysis. In: Proceedings of International Neural Network Conference (INNC 1990), Paris, France, pp. 305–308 (July 1990)

A New Label Maximization Based Incremental Neural Clustering Approach: Application to Text Clustering

Jean-Charles Lamirel[1], Raghvendra Mall[2], Shadi Al Shehabi[3], and Ghada Safi[3]

[1]LORIA, Campus Scientifique,
BP 239, Vandœuvre-lès-Nancy, France
jean-charles.lamirel@inria.fr,
http://www.loria.fr
[2]Center of Data Engineering, IIIT Hyderabad,
NBH-61, Hyderabad, Andhra Pradesh, India
raghvendra.mall@research.iiit.ac.in,
http://www.iiit.ac.in
[3]Department of Mathematics, Faculty of Science, Aleppo University,
Aleppo, Syria
shadialshehabi@gmail.com,ghada1@scs-net.org

Abstract. Neural clustering algorithms show high performance in the general context of the analysis of homogeneous textual dataset. This is especially true for the recent adaptive versions of these algorithms, like the incremental growing neural gas algorithm (IGNG) and the label maximization based incremental growing neural gas algorithm (IGNG-F). In this paper we highlight that there is a drastic decrease of performance of these algorithms, as well as the one of more classical algorithms, when a heterogeneous textual dataset is considered as an input. Specific quality measures and cluster labeling techniques that are independent of the clustering method are used for the precise performance evaluation. We provide variations to incremental growing neural gas algorithm exploiting in an incremental way knowledge from clusters about their current labeling along with cluster distance measure data. This solution leads to significant gain in performance for all types of datasets, especially for the clustering of complex heterogeneous textual data.

1 Introduction

Most of the clustering methods show reasonable performance on homogeneous textual dataset. However, the highest performance on such datasets are generally obtained by neural clustering methods [6]. The neural clustering methods are based on the principles of neighbourhood relation between clusters, either they are preset (fixed topology), like the "Self-Organizing Maps" also named SOM [4], or dynamic (free topology), like static "Neural Gas" (NG) [10] or "Growing Neural Gas" (GNG) [3]. As compared to usual clustering method, like K-means [9], this strategy makes them less sensitive to the initial conditions,

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 257–266, 2011.

which represents an important asset within the framework of data analysis of highly multidimensional and sparse data, like textual data.

The most known neural clustering method is the SOM method which is based on a mapping of the original data space onto a two dimensional grid of neurons. The SOM algorithm is presented in details in [4]. It consists of two basic procedures: (1) selecting a winning neuron on the grid and (2) updating weights of the winning neuron and of its neighbouring neurons.

In the NG algorithm [10], the weights of the neurons are adapted without any fixed topological arrangement within the neural network. Instead, this algorithm utilizes a neighbourhood ranking process of the neuron weights for a given input data. The weight changes are not determined by the relative distances between the neuron within a topologically pre-structured grid, but by the relative distance between the neurons within the input space, hence the name "neural gas" network.

The GNG algorithm [3] solves the static character of the NG algorithm bringing out the concept of evolving network. Hence, in this approach the number of neuron is adapted during the learning phase according to the characteristics of the data distribution. The creation of the neurons is made only periodically (each T iteration or time period) between the two neighbour neurons that accumulated the most important error for representing the data.

Recently, an incremental growing neural gas algorithm or (IGNG) [11] has been proposed by Prudent and Ennaji for general clustering tasks. It represents an adaptation of the GNG algorithm that relaxes the constraint of periodical evolution of the network. Hence, in this algorithm a new neuron is created each time the distance of the current input data to the existing neuron is greater than a prefixed threshold σ. The σ value is a global parameter that corresponds to the average distance of the data to the center of the dataset. Prudent and Ennaji have proved that their method produces better results than the existing neural and the non-neural methods on standard test distributions.

More recently, the results of the IGNG algorithm have been evaluated on heterogeneous datasets by Lamirel & al. [6] using generic quality measures and cluster labeling techniques. As the results have been proved to be insufficient for such data, a new incremental growing neural gas algorithm using the cluster label maximization (IGNG-F) has been proposed by the said authors. In this strategy the use of a standard distance measure for determining a winner is completely suppressed by considering the label maximization approach as the main winner selection process. Label maximization approach is more precisely detailed in [6]. One if its important advantage is that it provides the IGNG method with an efficient incremental character as it becomes independent of parameters.

In this paper we present several variations of IGNG-F approach based on combination of distance based criteria and cluster label maximization. We also use generic quality measures like *Micro-Precision*, *Micro-Recall*,*Cumulative Micro-Precision* and cluster labeling techniques for cluster evaluation. In the next section, we throw some light on these clustering quality measures. The third section provides a detailed analysis of the various adaptations of the

IGNG-F approach. Following which we present the results of our experiments on 2 different datasets of highly different complexity. The results on both datasets, especially on the most complex heterogeneous dataset, reflects the disadvantages of IGNG-F algorithm and other former neural clustering algorithms as compared to our new adaptations.

2 Clustering Quality Evaluation

An inherent problem of cluster quality evaluation persists when we try to compare various clustering algorithms. It has been shown in [5] that the inertia measures, or their adaptations [1], which are based on cluster profiles are often strongly biased and highly dependent on the clustering method. A need thus arised for such quality metrics which validate the intrinsic properties of the numerical clusters. We have thus proposed in [5] unsupervised variations of the recall and precision measures which have been extensively used in IR systems for evaluating the clusters.

For such purpose, we transform the recall and precision metrics to appropriate definitions for the clustering of a set of documents with a list of labels, or properties. The set of labels S_c that can be attributed to a cluster c are those which have maximum value for that cluster, considering an unsupervised $Recall-Precision$ metric [7][1]. The greater the unsupervised precision, the nearer the intentions of the data belonging to the same cluster will be with respect to each other. In a complementary way, the unsupervised recall criterion measures the exhaustiveness of the contents of the obtained clusters, evaluating to what extent single properties are associated with single clusters.

Global measures of Precision and Recall are obtained in two-ways. *Macro-Precision* and *Macro-Recall* measures are generated by averaging Recall and Precision of the cluster labels at the cluster level, in a first step, and by averaging the obtained values between clusters, in a second step. *Micro-Precision* and *Micro-Recall* measures are generated by averaging directly Recall and Precision of the cluster labels at a global level. Comparison of *Macro* measures and *Micro* measures makes it possible to identify heterogeneous results of clustering [8].

It is possible to refer not only to the information provided by the indices *Micro-Precision* and *Micro-Recall*, but to the calculation of the *Micro-Precision* operated cumulatively. In the latter case, the idea is to give a major influence to large clusters which are most likely to repatriate the heterogeneous information, and therefore, by themselves, lowering the quality of the resulting clustering. This calculation can be made as follows:

$$CMP = \frac{\sum_{i=|c_{inf}|,|c_{sup}|} \frac{1}{|C_{i+}^{-}|^2} \sum_{c \in C_{i+}^{-}, p \in S_c} \frac{|c_p|}{|c|}}{\sum_{i=|c_{inf}|,|c_{sup}|} \frac{1}{C_{i+}^{-}}} \quad (1)$$

[1] The IGNG-F algorithm uses this strategy as a substitute for the classical distance based measure which provides best results for homogeneous datasets [6].

where C_{i+} represents the subset of clusters of C for which the number of associated data is greater than i, and:

$$inf = argmin_{c_i \in C}|c_i|, sup = argmax_{c_i \in C}|c_i| \quad (2)$$

3 Various Adaptations of IGNG-F

The major problem IGNG-F faces in the case of heterogeneous datasets is that it can associate a data with labels completely different from the ones existing in the prototypes. This leads to strong decrease in performance in such case as labels belonging to different clusters are clubbed together. So to cope with this problem we propose three important variations of IGNG-F approach as mentioned below.

3.1 IGNG-Fv1

We use a distance based criteria to limit the number of prototypes which are investigated for a new upcoming data point. This allows to set a neighbourhood threshold and focus for each new data point which is lacking in the IGNG-F approach. It is similar to using the sigma parameter of IGNG, the only difference being the criteria is user oriented and can be varied in accordance. Generally, we see that there is a particular value of this criteria that can be used as threshold, beyond which all the prototypes are selected within the neighbourhood.

3.2 IGNG-Fv2 and IGNG-Fv3

Similarly to IGNG-Fv1, we use here a distance based criteria to define a neighbourhood threshold for a new data point. An important variation is that the *F-measure* which is to be computed with an added data point must not consider the new labels issued from the upcoming data. The new labels are the ones which were not taking part in the list of labels of a prototype before "augmenting" this prototype with the upcoming data (refer to [6]). This operation concerns all the prototypes that have been selected as potential winners for the new data point. This strategy will avoid to associate to a prototype data with labels that are completely different from the one exiting in this prototype and prevent the formation of heterogeneous clusters. We modify the step of calculation of labeling influence of the input data on the existing prototypes as proposed in IGNG-F algorithm: in the $F - Max$ or label maximization approach of this algorithm a label is attached to the prototype which has maximum *F-measure* for that label. However, in our case if there is more than one maximizer prototype we temporarily attach the label to all the maximizers. Considering now the behaviour of the algorithm, when there is an increase of *F-measure* produced on some selected prototypes by an upcoming data, we choose the winner prototype according to the formula:

$$Kappa = \Delta_{L-F_x(P_1)} \times SL(P_1) - \frac{EucDist(P_1, x)}{criteria \times att_1} \quad (3)$$

By doing so we try to maximize the increase in *F-measure* (Δ) along with maximizing the number of shared labels ($SL()$) between the challenger prototype (P_1) and the current data point (x). We also make use of an additional Euclidean distance ($EucDist()$) based measure between the prototype profiles and the upcoming data point profile. The lower is the distance, the higher is the probability of the data point being associated to the prototype. The *criteria* variable is the same as that used for limiting the neighbourhood size in IGNG-Fv1. It is used so as to keep *Kappa* value non-negative. The att_1 variable is a general attenuation factor whose role is to reduce the dominance of the Euclidean distance and have nearly equal contribution from Δ *F-measure*, shared labels ($SL()$) and distance between challenger prototype (P_1) and upcoming data point(x) (it has been set to 10 in our experiments).

The IGNG-Fv3 version is very similar to the IGNG-Fv2. The only difference is that the Cosine similarity is substituted to the Euclidean distance in the Kappa measure.

4 Datasets Description

For the experimental purpose we use 2 different datasets of highly different complexity, namely the Total-Use and the Lorraine datasets.

The **Total-Use** dataset consisted of 220 patent records related to oil engineering technology recorded during the year 1999. This dataset contains information such as the relationship between the patentees, the advantages of different type of oils, what are the technologies used by patentees and the context of exploitation of their final products. Based on the information present in the dataset, the labels belonging to the datasets have been categorized by the domain experts into four viewpoints or four categories namely Patentees, Title, Use and Advantages.

The task of extracting the labels from the dataset is divided into two elementary steps. At the step 1, the rough index set of each specific text sub field is constructed by the use of a basic computer-based indexing tool. At the step 2, the normalization of the rough index set associated to each viewpoint is performed by the domain expert in order to obtain the final index sets. In our experiment, we solely focus on the labels belonging to the Use viewpoint. Thus, the resulting corpus can be regarded as a homogeneous dataset as soon as it covers an elementary description field of the patents with a limited and contextual standardized vocabulary of 234 keywords or labels spanning over 220 documents.

The **Lorraine** dataset is build up from a set of bibliographic records resulting from the INIST PASCAL database and covering all the research activity performed in the Lorraine region during the year 2005. The structure of the records makes it possible to distinguish the titles, the summaries, the indexing labels and the authors as representatives of the contents of the information published in the corresponding article. In our experiment, the research topics associated with the labels field are solely considered. As soon as these labels cover themselves a large set of different topics (as far one to another as medicine from structural physics or forest cultivation, etc ...), the resulting experimental dataset can be

considered as a highly heterogeneous dataset. The number of records is 1920. A frequency threshold of 2 is applied on the initial label set resulting in a description space constituted of 3556 labels. Although the keyword data do not come directly from full text, we noted that the distribution of the terms took the form of a Zipf law (linear relation between the log of the frequencies of terms and the log of their row) characteristic of the full text data. The final keyword or label set also contains a high ratio of polysemic terms, like age, system, structure, device, etc ...

5 Results

For each method, we do many different experiments letting varying the number of clusters in the case of the static methods and the neighbourhood parameters in the case the incremental ones (see below). We have finally kept the best clustering results for each method regarding to the value of *Recall-Precision F-measure* and the *Cumulative Micro-Precision*.

We first conducted our experiments on the Total-Use dataset which is homogeneous by nature. Figure 1 shows the $Macro\text{-}F\text{-}Measure$, $Micro\ F\text{-}Measure$ and the $Cumulative\ Micro\text{-}Precision(CMP)$ for the dataset. The number of clusters represents the actual number of non-empty clusters among the total number of clusters used for clustering the dataset. We see that the $Macro\ F\text{-}Measure$ and $Micro\ F\text{-}Measure$ values are nearly similar for the different clustering approaches. However, the CMP value shows some difference. We see that for the SOM approach more than half of the clusters are empty. The NG and GNG algorithms have good $Macro$ and $Micro\ F\text{-}Measure$ but lower CMP than the IGNG approaches (except IGNG-Fv1).

In the case of the IGNG-Fv1 method, there is a maximum difference between the $Macro\ F\text{-}measure$ and the $Micro\ F\text{-}measure$. This difference clearly illustrates the presence of unbalanced or degenerated clustering results including some noisy clusters of large size. Hence, big noisy clusters cannot be detected by the $Macro\ F\text{-}measure$ as soon as they coexist with smaller relevant ones. This phenomenon is also illustrated by the low value of the CMP measure which we use as a central index for determining the best clustering quality. For IGNG-Fv1, the value of CMP is extremely low signifying that the properties (i.e. the labels) are unable to distinguish the different clusters efficiently, although we have high CMP values for IGNG and IGNG-F. The lower number of average label per document results in normalized label vectors for Total-Use dataset that are far apart from one another as in comparison to other datasets. This would signify higher distance between similar documents which can be one of the reasons for poor result of IGNG-Fv1. We see that the best results are obtained for IGNG-Fv2 which uses the neighbourhood criteria and $Kappa$ based selection procedure for winning neurons. The dataset is homogeneous by nature, so it is possible to reach such high precision values. Thus small, distinct and non-overlapping clusters are formed.

The standard K-Means approach produces the worst results on the first dataset. Furthermore, we leave out neural NG method for our next experiments because of its too high computation time.

Clustering Method	Nb of clusters	F-Measure Macro	F-Measure Micro	Cumulative Micro-Precision	MSE
SOM	90 (196)	0.766719	0.692499	0.530187	0.346252
K-Means	53 (55)	0.758615	0.594195	0.291856	0.139917
NG	82(96)	0.815711	0.714835	0.6	0.225764
GNG	86 (93)	0.784056	0.755068	0.670653	0.244798
IGNG	86 (87)	0.837917	0.796471	0.729932	0.259827
IGNG-F	68 (81)	0.83054	0.745341	0.690571	0.270658
IGNG-Fv1	70 (72)	0.823281	0.690168	0.184432	0.261331
IGNG-Fv2	93 (96)	0.855971	0.798538	0.731729	0.182697
IGNG-Fv3	91 (96)	0.851049	0.793669	0.723774	0.21983

Fig. 1. Clustering quality results of various algorithms on Total-Use homogeneous dataset

Even if it embeds stable topics, the Lorraine dataset is a very complex heterogeneous dataset as we have illustrated earlier. In a first step we restricted our experiment to 198 clusters as beyond this number, the GNG approach went to an infinite loop (see below). A first analysis of the results on this dataset shows that most of the clustering methods have huge difficulties to deal with it producing consequently very bad quality results, even with such high expected number of clusters, as it is illustrated in Figure 2 by the very low CMP values. It indicates the presence of degenerated results including few garbage clusters attracting most of the data in parallel with many chunks clusters representing either marginal groups or unformed clusters. This is the case for K-Means, IGNG, IGNG-Fv1 methods and at a lesser extent for GNG method.

This experiment also highlights the irrelevance of Mean Square Error (MSE) (or distance-based) quality indexes for estimating the clustering quality in complex cases. Hence, the K-Means methods that got the lowest MSE practically produces the worth results. This behaviour can be confirmed when one looks more precisely to the cluster content and the cluster size distribution for the said method, or even to the labels that can be extracted from the clusters in an unsupervised way using the expectation maximization methodology that is described in section 2. Hence, cluster label extraction permits to highlight that the K-means method mainly produced a "garbage" cluster with very big size that collects more than 80% of the data and attracts (i.e. maximizes) many kinds of different labels (3234 labels among a total of 3556) relating to multiple topics. Conversely, the good results of the IGNG-Fv2 method can be confirmed in the same way. Indeed, label extraction also shows that this latter method produces different clusters of similar size attracting semantically homogeneous labels groups. In addition, those groups clearly figure out the main research topics of the dataset that might also be identified by looking up to the different PASCAL classification codes which have been initially associated to the data by the analysts.

The CMP value for GNG approach was surprisingly greater for 136 clusters than for 198 clusters. Thus, increasing the expected number of clusters is not helpful to the method to discriminate between potential data groups in the

Clustering Method	No of clusters	F-Measure Macro	F-Meausre Micro	Cumulative Micro-Precision	MSE
SOM	216 (225)	0.35142	0.295278	0.136492	0.913648
K-Means	198 (400)	0.392339	0.288297	0.027891	0.582161
GNG	136 (136)	0.327476	0.256538	0.07864	0.73263
IGNG	198 (198)	0.339541	0.279246	0.028195	1.351955
IGNG-F	198 (198)	0.351322	0.280931	0.030203	1.334549
IGNG-Fv1	198 (198)	0.351322	0.280931	0.030203	1.334549
IGNG-Fv2	198 (198)	0.34936	0.278785	0.166262	1.333788
IGNG-Fv3	198 (198)	0.352215	0.280295	0.028852	1.333063

Fig. 2. Clustering quality results of various algorithms on Lorraine heterogeneous dataset

Lorraine dataset context. At the contrary, it is even lead the method to increase its garbage agglomeration effect. For higher number of clusters the GNG method does not provide any results on this dataset because of its incapacity to escape from an infinite cycle of creation-destruction of neurons (i.e. clusters).

The only consistent behaviour is shown by SOM and IGNG-Fv2 methods. The grid constrained learning of the SOM method seems to be a good strategy for preventing to produce too bad results in such a critical context. Hence, it enforces the homogeneity of the results by splitting both data and noise on the grid. The best results on this complex heterogeneous dataset are obtained with IGNG-Fv2 method. It is mainly because of the use of neighbourhood criteria for limiting the selection of prototypes and most importantly because of the choice of winner neuron based on the *Kappa* measure. It highlights the importance of taking into account the combination of maximum number of the shared labels with an upcoming data point, the maximum positive increment in *F-measure* and also consider the distance between the current prototype and the new data point.

We run a test for the cumulative Micro-Precision for IGNG-Fv2 and IGNG-Fv3 as they are similar in nature and differ only in the measure of similarity used. We found that as we increase the number of clusters beyond 198 clusters the actual peak value for the two methods are reached. For the other IGNG algorithms the results were not as efficient as for IGNG-Fv2 and IGNG-Fv3. For these two algorithms we allow the label to be associated to more than one winner neuron (cluster), the same label might thus belong to many different clusters. So when we perform an analysis on the clusters obtained by the two approaches, we see that there are 82 coherent clusters having more than 10 documents with each cluster having < 50 labels associated to it i.e. on an average 5 labels per document, significantly < 8.99 (global label average). From Figure 3, we observe that for the IGNG-Fv2 the maximum CMP value occurs at 290 clusters (0.25) while for IGNG-Fv3 the maximum CMP value occurs at 286 clusters (0.248). They follow very similar trends for large number of clusters though IGNG-Fv3 reaches high CMP value more consistently than IGNG-Fv2. Thus, they are able to much more appropriately cluster this highly complex dataset.

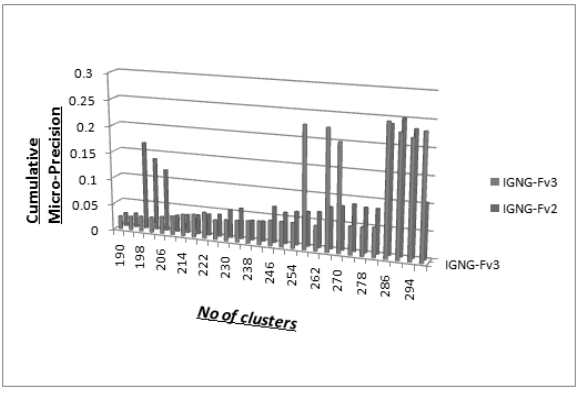


Fig. 3. Cumulative Micro-Precision vs Total Clusters for IGNG-Fv2 & IGNG-Fv3 algorithms on Lorraine dataset

6 Conclusion

Neural clustering algorithms show high performance in the usual context of the analysis of homogeneous textual dataset. This is especially true for the recent adaptive versions of these algorithms, like the incremental neural gas algorithm (IGNG). Nevertheless, this paper highlights clearly the drastic decrease of performance of these algorithms, as well as the one of more classical non neural algorithms, when a very complex heterogeneous textual dataset is considered as an input. Specific quality measures and cluster labeling techniques that are independent of the clustering method are used for performance evaluation. One of the main contributions of our paper has been to propose incremental growing neural gas algorithm exploiting knowledge issued from clusters current labeling in an incremental way in combination with the use of distance based measures. This solution led us to obtain very significant increase of performance for the clustering of textual data. Our IGNG-Fv2 approach is the most stable approach for the different datasets. It produces high quality clusters for each dataset unlike the other neural and non-neural approaches which have highly varying on the datasets. In our experiment the use of stable evaluation methodology has certainly represented a key point for guaranteeing the success of our approach. In the near future, we would like to enhance the approach by taking into consideration all the data points for clustering. We also aim at finding appropriate strategy to split the largest cluster and would like to adapt our enhanced label maximization similarity principles to several other clustering algorithms.

References

1. Davies, D., Bouldin, W.: A cluster separation measure. IEEE Transaction on Pattern Analysis and Machine Intelligence 1, 224–227 (1979)
2. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood for incomplete data via the em algorithm. Journal of the Royal Statistical Society B-39, 1–38 (1977)

3. Frizke, B.: A growing neural gas network learns topologies. Advances in neural Information processing Systems 7, 625–632 (1995)
4. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics 43, 56–59 (1982)
5. Lamirel, J.-C., Al-Shehabi, S., Francois, C., Hofmann, M.: New classification quality estimators for analysis of documentary information: application to patent analysis and web mapping. Scientometrics 60 (2004)
6. Lamirel, J.-C., Boulila, Z., Ghribi, M., Cuxac, P.: A new incremental growing neural gas algorithm based on clusters labeling maximization: Application to clustering of heterogeneous textual data. In: García-Pedrajas, N., Herrera, F., Fyfe, C., Benítez, J.M., Ali, M. (eds.) IEA/AIE 2010. LNCS, vol. 6098, pp. 139–148. Springer, Heidelberg (2010)
7. Lamirel, J.-C., Phuong, T.A., 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)
8. Lamirel, J.-C., Ghribi, M., Cuxac, P.: Unsupervised recall and precision measures: a step towards new efficient clustering quality indexes. In: Proceedings of the 19th Int. Conference on Computational Statistics (COMPSTAT 2010), Paris, France (August 2010)
9. MacQueen, J.: Some methods of classification and analysis of multivariate observations. In: Proc. 5th Berkeley Symposium in Mathematics, Statistics and Probability, vol. 1, pp. 281–297. Univ. of California, Berkeley, USA (1967)
10. Martinetz, T., Schulten, K.: A neural gas network learns topologies. Artificial Neural Networks, 397–402 (1991)
11. Prudent, Y., Ennaji, A.: An incremental growing neural gas learns topology. In: Neural Networks, 13th European Symposium on Artificial Neural Networks, Bruges, Belgium (April 2005)

A SOM-Based Analysis of Early Prosodic Acquisition of English by Brazilian Learners: Preliminary Results

Ana Cristina C. Silva[1], Ana Cristina P. Macedo[2], and Guilherme A. Barreto[3]

[1] Department of Linguistics, State University of Piauí, Brazil
cris0708@gmail.com
[2] Department of Linguistics, Federal University of Ceará, Brazil
pelosi@ufc.br
[3] Department of Teleinformatics Engineering, Federal University of Ceará, Brazil
guilherme@deti.ufc.br

Abstract. In this paper the SOM is used in an exploratory analysis of transfer phenomena from first language (L1) to the second language (L2) related to word/lexical stress. The basic hypothesis tested is whether the parameterization of the speech signal of the learner's utterances by standard signal processing techniques, such as Linear Predictive Coding (LPC), used to encode the input of the network results in efficient categorization of speakers by the SOM. Preliminary results indicates that the combination LPC+SOM is indeed able to produce well-defined clusters of speakers that possess similarities regarding the transfer of stress patterns among Brazilian students in learning English as a foreign language.

Keywords: Self-organizing map, word/lexical stress, linear predictive coding, U-matrix.

1 Introduction

Connectionist models have been playing an important role in language development in several areas, such as lexical and pronoun acquisition, syntactic sistematicity, language disorder modeling and prosodic analysis [16, 17, 20], just to mention a few. Most of these works are based on feedforward or recurrent supervised neural network architectures [4, 6, 8, 11], such as the MLP and Elman networks, but self-organizing neural network models have also been used as the primary linguistic model [9, 10, 13, 14, 18, 19].

For example, Li and co-workers [13, 14] simulated the lexical acquisition in infants using a self-organizing neural network model. The main objective of the research was to use the properties of topographic preservation of the Self-Organizing Map (SOM) [12] to study the emergence of linguistic categories and its organization throughout the stages of lexical learning. The model captured a series of important phenomena occurring in children's early lexical acquisition

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 267–276, 2011.

and had significant implications for models of language acquisition based on self-organizing neural networks.

Also using the SOM, Gauthier et al. [10] studied whether and how children could learn prosodic focus directly from input continuous speech. The authors explored how the focus could be learned from acoustic continuous signals in Mandarin, which were produced with co-occurring lexical tones and by various speakers. The results of this study showed that neural networks can develop unsupervised groupings of specific focus from the continuous dynamic speech signal, produced by various speakers in various lexical tone conditions, which may eventually lead to the acquisition of the prosodic focus.

Of particular interest to the current paper is the lexical stress, one of the most important prosodic elements. The stress in English has multiple functions ranging from an emphatic role, through the contrastive power to indicate syntactic relationships between words and word parts, such as the oppositions of pairs of noun and verb words, e.g. (**OB**ject, ob**JECT**), (**DE**sert, de**SERT**), (**CON**flict, con**FLICT**), etc.

In Brazilian Portuguese (BP) there is a tendency the trisyllabic and polysyllabic nouns be paroxytone. Trisyllabic and polysyllabic verbs in BP suffer a tendency to be oxytone. There is another trend: that of trisyllabic and polysyllabic adjectives in BP are paroxytone. According to some studies in the area of phonology of interlanguage [1, 2, 3, 15, 21], the lexical stress is most responsible for cases of language transfer, i.e. the influence of the predominant accent of L1 (first language) in L2 (second language) learning.

Despite some previous works involving the connectionist modeling of prosodic features for language development and identification [4, 6, 10], to the best of our knowledge, there has been no systematic investigation nor an exploratory analysis of transfer phenomena from L1 to L2 related to word/lexical stress by means of connectionist model, such as the SOM. Furthermore, we are not aware of studies on the application of artificial neural networks to investigate how the knowledge of Brazilian learners of English is organized in relation to the acquisition of early L2 stress and transference of stress pattern from L1 to L2.

From the exposed, this article aims to investigate whether and how the SOM network is able to build well-defined groups (clusters) of speakers that possess similarities regarding the transfer of stress patterns among Brazilian students in learning English as a foreign language. The ultimate goal of this research is to use the SOM as a tool to evaluate the proficiency level of students. The basic hypothesis tested is whether the parameterization of the speech signal of the learner's utterances by standard signal processing techniques, such as Linear Predictive Coding (LPC), for encoding the input of the network is efficient in the categorization of speakers.

The remainder of the paper is organized as follows. In Section 2 we briefly describe the SOM, the corpus and the speech parameterization technique used in this research. The results of computer simulations and comments about them are presented in Section 3. The paper is concluded in Section 4.

2 Methods

2.1 The Self-Organizing Map

In what follows, a brief description of the original SOM algorithm, introduced by Kohonen [12], is given. Let us denote $\mathbf{m}_i(t) \in \mathbb{R}^p$ as the weight vector of the i-th neuron in the map. After initializing all the weight vectors randomly or according to some heuristic, each iteration of the SOM algorithm involves two steps. First, for a given input vector $\mathbf{x}(t) \in \mathbb{R}^p$, we find the current winning neuron, $i^*(t)$, as follows

$$i^*(t) = \arg\min_{\forall i}\{\|\mathbf{x}(t) - \mathbf{m}_i(t)\|\}. \tag{1}$$

where t denotes the iterations of the algorithm. Then, it is necessary to adjust the weight vectors of the winning neuron and of those neurons in its neighborhood:

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \eta(t)h(i^*, i; t)[\mathbf{x}(t) - \mathbf{m}_i(t)], \tag{2}$$

where $0 < \eta(t) < 1$ is the learning rate and $h(i^*, i; t)$ is a Gaussian weighting function that limits the neighborhood of the winning neuron:

$$h(i^*, i; t) = \exp\left(-\frac{\|\mathbf{r}_i(t) - \mathbf{r}_{i^*}(t)\|^2}{2\sigma^2(t)}\right), \tag{3}$$

where $\mathbf{r}_i(t)$ and $\mathbf{r}_i^*(t)$, are respectively, the positions of neurons i and i^* in a predefined output array where the neurons are arranged in the nodes, and $\sigma(t) > 0$ defines the radius of the neighborhood function at time t. To guarantee convergence of the algorithm, $\eta(t)$ and $\sigma(t)$ decay exponentially in time according to the following expressions:

$$\eta(t) = \eta_0\left(\frac{\eta_T}{\eta_0}\right)^{(t/T)} \qquad \text{and} \qquad \sigma(t) = \sigma_0\left(\frac{\sigma_T}{\sigma_0}\right)^{(t/T)}, \tag{4}$$

where $\eta_0(\sigma_0)$ and $\eta_T(\sigma_T)$ are the initial and final values of $\eta(t)$ $(\sigma(t))$.

The incremental learning process defined by Eqs. (1) and (2) can often be replaced by the following batch computation version which is usually faster.

1. Initialize the weight vectors $\mathbf{m}_i$, $\forall i$.
2. For each neuron i, collect a list of all those input vectors $\mathbf{x}(t)$, whose most similar weight vector belongs to the neighborhood set N_i of neuron i.
3. Take as the new weight vector $\mathbf{m}_i$ the mean over the respective list.
4. Repeat Step 2 a few times until convergence is reached.

Steps 2 and 3 of the batch SOM algorithm need less memory if at Step 2 one only make lists of the input vectors $\mathbf{x}(t)$ at those neurons that have been selected for winner, and at Step 3 we take the mean over the union of the lists that belong to the neighborhood set N_i of neuron i.

In addition to usual vector quantization properties properties, the resulting ordered map also preserves the topology of the input samples in the sense that adjacent input patterns are mapped into adjacent neurons on the map. Due to this topology-preserving property, the SOM is able to cluster input information and spatial relationships of the data on the map.

2.2 Input Corpus and Data Representation

The corpus of this research is composed of interview recordings with 30 students (learners) of a higher education institution in the city of Fortaleza, federal state of Ceará, aged between 18 and 25, all Brazilians, of both genders, who have never traveled to an English-speaking country until the time of interview. It was decided to allocate the 30 participants in five different levels of development, using the criterion of length of exposure to language. Based on this fact, number of classroom hours accumulated in the discipline of English language obtained through interviews and questionnaires completed by participants was established as a circumstantial criterion of classification and organization of the individuals.

The participants' utterances were recorded in the software *SoundForge* (version 5.0) in WAV audio files at a sampling rate of 44.1 KHz with 16-bit resolution, single channel (mono). After this phase, each word representing the lexical item to be investigated (e.g. object, separate, desert, etc.) was manually segmented with the help of a phonetician using the same software.

As the speech signal cannot be directly used to feed the network because it contains thousands of samples, which would make their processing very slow, and also for being very noisy, which makes it extremely difficult to extract knowledge, the solution is to represent it numerically with a set of coefficients obtained from the application of mathematical techniques such as linear prediction coefficients and mel-cepstral coefficients, with the speech signal divided into multiple frames. Thus, the speech signal of the learners is numerically represented by coefficient vector sets computed using the PRAAT software [5].

2.3 Speech Signal Parametrization (Feature Extraction)

The process of feature extraction of the speech signal is a crucial step in the connectionist approach to pattern classification and clustering. This step consists in applying standard signal processing techniques to the original speech signal in order to convert it to more suitable compact mathematical representation that permits the identification of a given utterance by a connectionist model.

Linear predictive coding (LPC) is a signal processing technique widely used for the parametrization of the speech signal in several applications, such as speech compression, speech synthesis and speech recognition [7]. Roughly speaking, the LPC[1] technique represents small segments (or frames) of the speech signal by the coefficients of autoregressive (AR) linear predictors. For example, if the speech signal has 500 frames, it will be parameterized by a set of 500 coefficient vectors. To assure stationarity, each frame usually has a short duration ($\sim$ 10-30ms).

The set of LPC coefficient (LPCC) vectors associated with the utterance of a given word are then organized along the rows of a matrix of coefficients. For

[1] LPC coefficients can extract the intensity and frequency of the speech signal. These two characteristics are closely associated with the prosodic element "accent". In English, the stress is the junction of three perceptual factors interrelated: 1) quantity / length (measured in ms) related to the size of the syllable, 2) intensity (measured in dB) related to amplitude and 3) height (measured in Hz), i.e., the value of higher F0 in an utterance.

example, if 500 coefficient vectors generated, one vector for each frame, the corresponding matrix of coefficients has 500 rows. The number of columns of this matrix is equal to order of the AR predictor used in the LPC analysis. The matrices of coefficients are then used to train the SOM.

2.4 SOM Training and Data Visualization

Four simulations were ran with parameters that varied according to the need to adjust to the phenomenon in question. The experiments were design in order to verify whether the network could organize (discriminate) learners depending on the transference of stress pattern from L1 to L2 are detailed below. When applied to the problem of interest, the simulation process of the SOM and the analysis of results of the training involves the following steps:

1. Startup and training (learning) of the network;
2. Evaluation of the quality of the map using the quantization error (QE) and topological error (TE);
3. Generation of the U-matrix and labeled map after each training run;
4. Validation of clusters through the Davies-Bouldin index (DB);
5. Tabulation of the data for all outcome measures of network performance (the quantization error and topological error).

All simulations were conducted using a two-dimensional hexagonal SOM, with hexagonal neighborhood structure, Gaussian neighborhood function, random initiation of weights and batch learning. For all the experiments we simulated a 5×5 SOM, for 250 epochs (50 for rough training, 200 for fine tuning) with initial and final neighborhood of 4 and 1, respectively. The maximum numbers of clusters used by the DB index was set to 10. These specifications proved adequate to treat the phenomena in question. The SOM toolbox [23] was used to run all the experiments to be described.

As mentioned in the Subsection 2.3, every word uttered by a speaker generates a coefficient matrix. In order to identify this speaker in a posterior analysis of the results, it is necessary to label the data (row) vectors in that matrix as belonging to that particular speaker. For this purposes, an alphanumeric label is appended to each row vector in an additional column. Finally, the text files containing labeled data related to the utterance of a specific word for all the speakers are concatenated into a single file.

It is noteworthy that in addition to the label that identifies the speaker, other labels can be associated with a given coefficient matrix of that speaker. For instance, a second label can identify the linguistic category in which the word pronounced is inserted. This Multi-Label (ML) Analysis is introduced in this paper with the goal of determining which labeling is more appropriate to the type of parameterization used. In other words, ML analysis can help inferring which linguistic properties of the speech signal are encoded in the LPC coefficients.

Finally, the U-matrix [22] is used as a tool to visualize the clusters formed during the learning process.

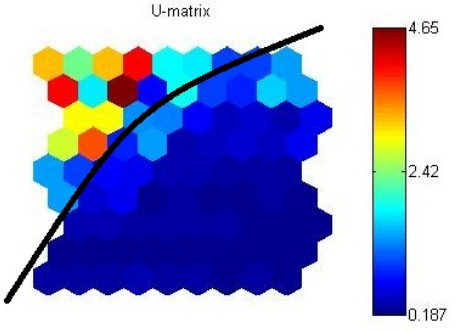


Fig. 1. U-matrix revealing the formation of two major groups, one probably related to speakers who transfer the stress pattern and the other to speakers who do not transfer

3 Results and Discussion

This simulation aims at investigating whether the SOM would be able to organize the speakers in clusters, according to the process of transferring the stress pattern of Brazilian Portuguese into English. All the 30 speakers were asked to utter 30 different English sentences containing situations where certain words of interest act sometimes as a verb or as a noun. In this paper, we report only the results obtained for the sentence 'I object to going to a bar', where the word of interest is the verb 'obJECT'. The full corpus is available to the interested reader upon request.

Three types of graphics were generated after SOM training: U-Matrix, labeled map (majority rule) and clustered map. The U-matrix and the clustered map requires no labeled data to be constructed. The labeled map is more useful for our purposes if labeled data are available since labels may provide a better understanding of speakers' organization as a function of their linguistic abilities. It is worth pointing out that all the SOM computations are performed using unlabeled data, i.e. it runs totally in an unsupervised way. The labels are used only in the analysis of the results.

Two criteria were followed for labeling purposes. At first, the speakers' labels carry no information about errors in L2 stress, i.e., the transfer pattern of L1 to L2. In this case, the data from a given speaker is labeled by a number indicating his/her formal education level in L2 studies (i.e. period in an English course) and his/her order in the interview process. For example, the label '608' denotes a speaker in the 6th semester ranked 8th in the list of individuals interviewed for this research. The second labeling criterion added the characters "er" to the label when a speaker misses the pronunciation, i.e. when he/she transfers the pattern from L1 (Brazilian Portuguese) to L2 (English). For example, the label '203er' denotes a speaker in the 2nd semester, ranked 3rd in the interview sequence and who missed the pronunciation.

Table 1. DB index values for different values of K

$K=2$	$K=3$	$K=2$	$K=3$	$K=2$	$K=3$	$K=2$	$K=3$	$K=10$
0.4306	0.8494	0.6018	0.6100	0.7914	0.6484	0.7520	3.1829	11.3072

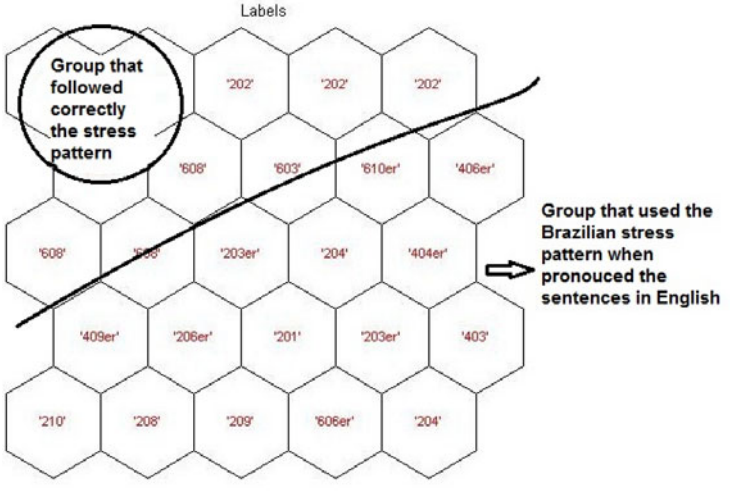


Fig. 2. Labeled map associated to the U-matrix shown in Figure 1, confirming the expectation of two major groups of students, one containing mainly individuals who transfer the BP stress pattern and one that does not transfer

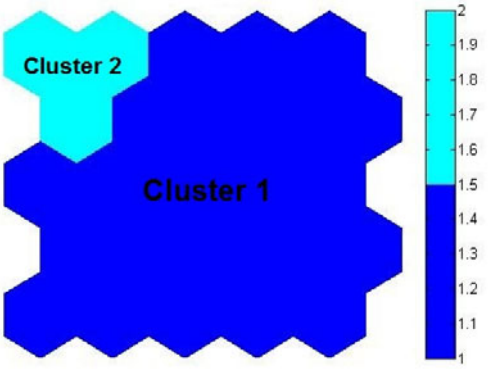


Fig. 3. Clustered map suggesting the existence of two well-defined clusters, according to Davies-Bouldin index

Figures 1 to 3 illustrate the obtained results. Errors in the pronunciation of this word occur when it is pronounced as a noun (**OB**ject), instead of a verb (ob**JECT**). For each speaker, the speech signal segment corresponding to the word "Object" is manually selected and parameterized by the LPC method. The order of the AR predictor for this experiment was set to 10 and the duration of each frame was set to 25ms.

The resulting U-matrix is shown in Figure 1. By analyzing this figure one can clearly see the signs of formation of two clusters: a larger group (the one with prototypes closer to each other - in blue in the figure) and a smaller group (the one with prototypes more separated from each other - in different colors in the figure). Based on the a priori analysis of the frequency of occurrence of the labels provided by the phonetician, the larger group probably is the one containing the individuals who transfer the primary stress. This hypothesis can only be confirmed by analyzing the labeled map shown in Figure 2.

Neuron labeling is carried out by the majority rule, i.e. the neuron inherits the label that occurs most frequently among the data vectors (LPCC vectors) mapped to that neuron. The labeled map confirms the hypothesis of two groups raised by the U-matrix. The neurons whose labels include the characters "er" are located below the solid line separating the map into two parts.

It is worth noting that this clear separation of students was carried out in an unsupervised way by the SOM using solely the information provided by the LPC coefficients, i.e. the network organizes the students by similarity between their feature (LPCC) vectors only. No a priori linguistic knowledge was used during the feature extraction process nor the SOM training. Label information was indeed provided by an expert but it is used only for the purpose of interpretation of the trained map.

The clustered map in Figure 3 adds corroborating evidence to the results provided by the labeled map and the U-matrix concerning the emergence of two well-defined groups. Clustering of the SOM was carried out using the K-means algorithm, varying K from 1 to 10, following the approach proposed in [23]. The optimal value for K, according to the DB index, was $K_{opt} = 2$ (see Table 1).

4 Conclusion

The preliminary results presented in this paper can serve as a starting point to demonstrate that an unsupervised neural network can be useful to visualize the cluster formation of prosody-related linguistic phenomenon, in this case, the transference of lexical stress. We started from the assumption that the parameterization of the speech signal through the LPC coefficients would be effective in the categorization of speakers for prosodic features.

The segregation of the map in regions of well-defined clusters suggested that the learners were grouped by similar phonetic-acoustic features. According to the rounds of experiments, it was confirmed that the network discriminated speakers according to prosodic features and organized them according to similarities on these characteristics. Importantly, within these two large groups (the group that

transfers the BP stress pattern and what does not transfer) there can be subgroups (subclusters) which, when closely examined in isolation, might reveal rich information for the linguistic analysis of learner's utterances as well as to contribute to understanding the organization of the data set. We are currently, developing experiments to analyze these subgroups.

Further tests are to be made and with more results, we hope to perfect the proposed SOM-based methodology and use it in the future as a tool for determining the language proficiency level classification in foreign languages.

Acknowledgements. The authors thank FUNCAP and CAPES (Brazilian agencies for promoting science) for the financial support to this research.

References

1. Albini, A.B.: The influence of the portuguese language in accentuation of english words by brazilian students (in portuguese). Revista Prolíngua 2(1), 44–56 (2009)
2. Archibald, J.: A formal model of learning L2 prosodic phonology. Second Language Research 10(3), 215–240 (1994)
3. Baptista, B.O.: An analysis of errors of Brazilians in the placement of English word stress. Master's thesis, Postgraduate Program on Linguistics, Federal University of Santa Catarina, Brazil (1981)
4. Blanc, J.M., Dominey, P.F.: Identification of prosodic attitudes by a temporal recurrent network. Cognitive Brain Research 17, 693–699 (2003)
5. Boersma, P., Weenink, D.: Praat: doing phonetics by computer (2011), http://www.praat.org, version 5.2.10 (retrieved January 11, 2011)
6. Cummins, F., Gers, F., Schmidhuber, J.: Language identification from prosody without explicit features. In: Proceedings of the 6th European Conference on Speech Communication and Technology (EUROSPEECH 1999), pp. 305–308 (1999)
7. Deller, J., Hansen, J.H.L., Proakis, J.: Discrete-Time Processing of Speech Signals. John Wiley & Sons, Chichester (2000)
8. Elman, J.L.: Finding structure in time. Cognitive Science 14, 179–211 (1990)
9. Farkas, I., Crocker, M.W.: Syntactic systematicity in sentence processing with a recurrent self-organizing network. Neurocomputing 71, 1172–1179 (2008)
10. Gauthier, B., Shi, R., Xu, Y.: Learning prosodic focus from continuous speech input: A neural network exploration. Language Learning and Development 5, 94–114 (2009)
11. Kaznatcheev, A.: A connectionist study on the interplay of nouns and pronouns in personal pronoun acquisition. Cognitive Computation 2, 280–284 (2010)
12. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, Heidelberg (2001)
13. Li, P., Farkas, I., MacWhinney, B.: Early lexical development in a self organizing neural network. Neural Networks 17, 1345–1362 (2004)
14. Li, P., Zhao, X., MacWhinney, B.: Dynamic self-organization and early lexical development in children. Cognitive Science 31, 581–612 (2007)
15. Mairs, J.L.: Stress assignment in interlanguage phonology: an analysis of the stress system of spanish speakers learning english. In: Gass, M., Schatcther, J. (eds.) Linguistic Perspectives on Second Language Acquisition, Cambridge University Press, Cambridge, USA (1989)

16. McClelland, J.L.: The place of modeling in cognitive science. Topics in Cognitive Science 1, 11–38 (2009)
17. McClelland, J.L., Botvinick, M.M., Noelle, D.C., Plaut, D.C., Rogers, T.T., Seidenberg, M.S., Smith, L.B.: Letting structure emerge: connectionist and dynamical systems approaches to cognition. Trends in Cognitive Sciences 14, 348–356 (2010)
18. Miikkulainen, R.: Dyslexic and category-specific aphasic impairments in a self organizing feature map model of the lexicon. Brain and Language 59, 334–366 (1997)
19. Miikkulainen, R., Kiran, S.: Modeling the bilingual lexicon of an individual subject. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 191–199. Springer, Heidelberg (2009)
20. Poveda, J., Vellido, A.: Neural network models for language acquisition: A brief survey. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 1346–1357. Springer, Heidelberg (2006)
21. Silva, A.C.C.: The production and perception of word stress in minimal pairs of the english language by brazilian learners. Master's thesis, Postgraduate Program on Linguistics, Federal University of Ceará, Brazil (in Portuguese) (2005)
22. Ultsch, A., Siemon, H.P.: Kohonen's self organizing feature maps for exploratory data analysis. In: Proceedings of the International Neural Network Conference (ICNN 1990), pp. 305–308. Kluwer Academic Publishers, Dordrecht (1990)
23. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11(3), 586–600 (2000)

A General Framework for Dimensionality Reduction for Large Data Sets

Barbara Hammer[1], Michael Biehl[2],
Kerstin Bunte[2], and Bassam Mokbel[1]

[1] CITEC centre of excellence, Bielefeld University, 33615 Bielefeld - Germany
[2] University of Groningen - Johann Bernoulli Institute for Mathematics and Computer Science, Nijenborgh 9, Groningen - The Netherlands
bhammer@techfak.uni-bielefeld.de

Abstract. With electronic data increasing dramatically in almost all areas of research, a plethora of new techniques for automatic dimensionality reduction and data visualization has become available in recent years. These offer an interface which allows humans to rapidly scan through large volumes of data. With data sets becoming larger and larger, however, the standard methods can no longer be applied directly. Random subsampling or prior clustering still being one of the most popular solutions in this case, we discuss a principled alternative and formalize the approaches under a general perspectives of dimensionality reduction as cost optimization. We have a first look at the question whether these techniques can be accompanied by theoretical guarantees.

1 Introduction

Numerous data visualization techniques and dimensionality reduction tools have been proposed in the last years which help humans to rapidly scan through large volumes of data relying on the astonishing cognitive capabilities of humans for visual perception [9,18,16,7,19]. These visualization tools offer flexible interfaces to the data in diverse areas such as robotics, medicine, the web, biology, etc., where electronic data sets as well as their complexity and dimensionality have increased dramatically in the past years. The problem of dimensionality reduction and data visualization is essentially ill-posed, such that a variety of different methods which impose different constraints on the visualization task has been proposed: Spectral dimensionality reduction techniques such as LLE [12], Isomap [15], or Laplacian eigenmaps [3] rely on the spectrum of the neighborhood graph of the data. They preserve important properties of this graph and lead to a unique algebraic solution. Many of these methods rely on very simple affinity functions such as Gaussians such that their results can be flawed. By using more complex affinities, techniques such as Isomap [15] or maximum variance unfolding [20] can partially avoid this problem at the prize of higher computational costs. Nonlinear methods often have the drawback of local optima albeit showing probably more appropriate results, see e.g. [6,16,5,19].

Visualization being an ill-posed problem, its formal evaluation is a matter of ongoing debate. While an evaluation is eventually not possible outside the given

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 277–287, 2011.

context, a few formal evaluation measures of dimensionality reduction have been proposed in the last years which quantify aspects which are universally important for data visualization. Essentially, the evaluation measures quantify in how far the notion of neighborhood coincides in the original data space and the projection space, yielding explicit formulas by means of trustworthiness and continuity, the co-ranking matrix, or an information retrieval point of view, for example [19,10]. This perspective even gives rise to new visualization techniques since it is possible to directly optimize the given evaluation measure [19]. Naturally, the respective formalization heavily biases the methods such that these evaluation measures can only serve as one indicator for the appropriateness of the maps.

With data sets getting larger and larger, quite a few problems arise for dimensionality reduction methods: on the one hand, the techniques are no longer feasible, their processing time often scaling quadratically with respect to the number of data. In addition, it is often not reasonable at all to map *all* given data points at once since the corresponding projection would be overloaded – the projection plane being completely filled with points in the limit of large data sets. Due to this fact, a simple heuristic is often used in such cases: a subset is picked either randomly or based on general principles (such as clustering) and this subset is subsequently projected with the dimensionality reduction technique (see e.g. [19]). If additional points are considered, they are projected using out-of-sample extensions which often require additional effort (in some cases, they are not directly given at all). While reasonable, this procedure is rather ad hoc and it is not clear in how far it can be substantiated by formal guarantees.

In this contribution, we formalize dimensionality reduction as cost optimization, including different popular techniques such as LLE, MDS, Isomap, or t-SNE. Within this general framework, we formalize the principle of subsampling and out of sample extensions as a way to find a dimensionality reduction mapping of the full data space by means of an implicit mapping. As an alternative, we propose a general principle to extend dimensionality reduction tools to obtain an explicit dimensionality reduction mapping with fixed prior shape. Dimensionality reduction with prior clustering can be seen as a special case of this setting, more complex functions such as e.g. locally linear functions being possible. This general framework has two consequences: we can generically consider out-of-sample extensions, which are given explicitly assuming an explicit mapping, and one can formally access the generalization ability of the models.

2 Dimensionality Reduction as Cost Optimization and Out-of-Sample Extensions

First, we shortly review a few popular dimensionality reduction methods. and put them into a general framework of cost optimization. In general, dimension reduction maps data points $\mathbb{R}^N \ni \boldsymbol{x}^i \to \boldsymbol{y}^i \in \mathbb{R}^2$. Corresponding distances are denoted as $d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^j)$ for the original manifold, and $d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j)$ for the projection space. Usually, $d_{\mathcal{E}}$ is chosen as the Euclidean distance, while $d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^j)$ is sometimes picked in more general form.

Multidimensional Scaling. Multidimensional scaling (MDS) preserves distances by minimizing $E_{\text{MDS}} = \sum_{ij} w_{ij}(d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^j) - d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j))^2$ with Euclidean distances, where the weights w_{ij} can be chosen appropriately, e.g. $w_{ij} = 1/d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^j)$ [9]. Optimization can take place by a gradient descent.

Isomap. Isomap [15] substitutes the Euclidean distance $d_{\mathcal{X}}$ by the geodesic distance as an approximation of the true manifold distance.

Locally Linear Embedding. Locally linear embedding (LLE) [12] first expresses local topologies by reconstructing a data point by its local neighborhood (denoted by $i \to j$): minimize $\sum_i (\boldsymbol{x}^i - \sum_{i \to j} w_{ij} \boldsymbol{x}^j)^2$ with $\sum w_{ij} = 1$. Afterwards, projections preserve the local linear relationships in a least squares sense: minimize $\sum_i (\boldsymbol{y}^i - \sum_{i \to j} w_{ij} \boldsymbol{y}^j)^2$ with $\sum \boldsymbol{y}^i = 0$ and $\mathbf{Y}^t \mathbf{Y} = \mathbf{n}$.

Laplacian Eigenmaps. Laplacian eigenmaps [3] also start with a local neighborhood graph and weight pairwise connections w_{ij} using the heat kernel. Projection is based on the eigendirections of the smallest eigenvalues larger than 0 as computed in the generalized eigenvalue problem given by the corresponding graph Laplacian. This is equivalent to minimizing $\sum_{i \to j} w_{ij} d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j)^2$ with Euclidean distance, under the constraint $\mathbf{Y}^t D \mathbf{Y} = \mathbf{1}$ and $\mathbf{Y}^t D \mathbf{1} = \mathbf{0}$, where D is the degree matrix and $\mathbf{Y}$ refers to the matrix of coefficients.

Maximum Variance Unfolding. Maximum variance unfolding (MVU) [20] also uses a neighborhood graph. It finds projections $\boldsymbol{y}^i$ such that the variance of the projection is maximized, i.e. $\sum_{ij} d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j)^2$ is maximum subject to a preservation of neighbors, i.e. $d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j) = d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^j)$ for all neighbored points $\boldsymbol{x}^i$ and $\boldsymbol{x}^j$, and the normalization $\sum \boldsymbol{y}^i = 0$. This can be reformulated as a convex problem by considering the variables $(\boldsymbol{y}^i)^\top \boldsymbol{y}$ instead. Further, an exact solution need not exist such that, possibly, slack variables have to be introduced.

Stochastic Neighbor Embedding. Stochastic neighbor embedding (SNE) [6] defines probabilities

$$p_{j|i} = \frac{\exp\left(\frac{-d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^j)^2}{2\sigma_i}\right)}{\sum_{k \neq i} \exp\left(\frac{-d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^k)^2}{2\sigma_i}\right)} \text{ and } q_{j|i} = \frac{\exp\left(-d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j\right)^2}{\sum_{k \neq i} \exp\left(-d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^k)^2\right)}$$

with Euclidean distances as default. Then it optimizes the Kullback-Leibler divergence $E_{\text{SNE}} = -\sum_{ij} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$, where bandwidths σ_i are determined based on the so-called perplexity which determines the number of neighbors of a given point. A gradient descent is used for the optimization.

T-distributed Stochastic Neighbor Embedding. t-distributed SNE (t-SNE) [16] slightly modifies the SNE cost function and uses a distribution in the embedding space with long tails, student-t. Its cost function is $E_{\text{t-SNE}} = \sum_i \sum_j p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right)$ where $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$ symmetrizes the conditional probabilities, n denoting the number of data points, and

$$q_{ij} = \frac{(1 + d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j)/\varsigma)^{-\frac{\varsigma+1}{2}}}{\sum_{k \neq l}(1 + d_{\mathcal{E}}(\boldsymbol{y}^k, \boldsymbol{y}^l)/\varsigma)^{-\frac{\varsigma+1}{2}}}$$

is given by student-t with parameter $\varsigma = -1$, for example. Optimization takes place by means of a gradient method.

Neighborhood Retrieval Visualizer. The neighborhood retrieval visualizer (NeRV) is derived from an information retrieval point of view, yielding, with a specific choice of the neighborhoods, the symmetric cost term

$$E_{\text{NeRV}} = -\lambda \sum_{ij} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} - (1 - \lambda) \sum_{ij} q_{j|i} \log \frac{q_{j|i}}{p_{j|i}}$$

with probabilities as for SNE and a weighting parameter $\lambda \in [0, 1]$ [19]. Similarly, t-NeRV generalizes t-SNE by considering the alternative symmetric pairwise probabilities p_{ij} and q_{ij} just as t-SNE in the original and projection space in the symmetric version of the Kullback-Leibler divergence.

A General View. These methods obey one general principle: characteristics of the data $\boldsymbol{x}$ are computed and projections $\boldsymbol{y}$ are determined such that the corresponding characteristics of the projections coincide with the characteristics of $\boldsymbol{x}$ as far as possible, fulfilling possibly additional constraints or objectives to achieve uniqueness. I.e. we compute characteristics $\text{char}(\boldsymbol{x})$, then projections $\boldsymbol{y}$ are determined such that $\text{error}(\text{char}(\boldsymbol{x}), \text{char}(\boldsymbol{y}))$ becomes small. Thereby, the methods differ in the way how data characteristics are defined and computed and how exactly the similarity of the characteristics is defined and optimized.

Table 1 summarizes the properties of the optimization methods under this point of view. Naturally, the methods severely differ with respect to the way in which optimization takes place: in some cases, the characteristics can be directly computed from the data (such as distances), in others, an optimization step is required (such as local linear weights). In some cases, the optimization of the error measure can be done in closed form (such as for Laplacian eigenmaps), in other cases, numerical optimization is necessary (such as for t-SNE).

3 Dimensionality Reduction for Large Data Sets

Many of the above methods depend on pairwise distances, such that their effort scales quadratically with the number of data. This makes them infeasible for large data sets. In addition, even linear techniques (such as e.g. presented in [4]) become infeasible for large data sets such that sublinear or even constant time techniques are required. Further, it usually does not make sense to project *all* data, the projection plane being almost completely filled with points. For this reason, often, simple random subsampling is used and the projections of just a subsample of the full data set are shown, see e.g. the overviews [18,19].

Table 1. Many dimensionality reduction methods can be put into a general framework: characteristics of the data are extracted. Projections lead to corresponding characteristics depending on the coefficients. These coefficients are determined such that an error measure of the characteristics is minimized, fulfilling probably additional constraints.

method	**characteristics of data**	**characteristics of projections**	**error measure**
MDS	Euclidean distance $d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^j)$	Euclidean distance $d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j)$	minimize weighted least squared error
Isomap	Geodesic distance $d_{\text{geodesic}}(\boldsymbol{x}^i, \boldsymbol{x}^j)$	Euclidean distance $d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j)$	minimize weighted least squared error
LLE	reconstruction weights w_{ij} such that $\sum(\boldsymbol{x}^i - \sum_{i\to j} w_{ij}\boldsymbol{x}^j)^2$ is minimum with constraints $\sum_j w_{ij} = 1$	reconstruction weights $\tilde{w}_{ij}$ such that $\sum(\boldsymbol{y}^i - \sum_{i\to j} \tilde{w}_{ij}\boldsymbol{y}^j)^2$ is minimum with constraints $\sum \boldsymbol{y}^i = 0$, $\mathbf{Y}^t\mathbf{Y} = \mathbf{n}$	enforce identity $w_{ij} = \tilde{w}_{ij}$
Laplacian eigenmap	negative heat kernel weights $-w_{ij} = \exp(-d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^j)^2/t)$ for $i \to j$	squared Euclidean distance $d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j)^2$ for $i \to j$ with constraints $\mathbf{Y}^t D\mathbf{Y} = \mathbf{1}$, $\mathbf{Y}^t D\mathbf{1} = \mathbf{0}$	maximize correlation
MVU	Euclidean distance $d_{\mathcal{X}}(\boldsymbol{x}^i, \boldsymbol{x}^j)$ for $i \to j$	Euclidean distance $d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j)$ for $i \to j$ such that $\sum_{ij} d_{\mathcal{E}}(\boldsymbol{y}^i, \boldsymbol{y}^j)^2$ is maximum and $\sum_j \boldsymbol{y}^i = 0$.	enforce identitiy (introducing slack variables if necessary)
SNE	probab. $p_{j\|i} = \frac{\exp\left(-d_{\mathcal{X}}(\boldsymbol{x}^i,\boldsymbol{x}^j)^2/2\sigma_i\right)}{\sum_{k\neq i}\exp\left(-d_{\mathcal{X}}(\boldsymbol{x}^i,\boldsymbol{x}^k)^2/2\sigma_i\right)}$	probab. $q_{j\|i} = \frac{\exp\left(-d_{\mathcal{E}}(\boldsymbol{y}^i,\boldsymbol{y}^j)^2\right.}{\sum_{k\neq i}\exp\left(-d_{\mathcal{E}}(\boldsymbol{y}^i,\boldsymbol{y}^k)^2\right)}$	minimize Kullback-Leibler divergences
t-SNE	probab. $p_{ij} = \frac{p_{j\|i}+p_{i\|j}}{2n}$	probab. $q_{ij} = \frac{(1+d_{\mathcal{E}}(\boldsymbol{y}^i,\boldsymbol{y}^j)/\varsigma)^{-\frac{\varsigma+1}{2}}}{\sum_{k\neq l}(1+d_{\mathcal{E}}(\boldsymbol{y}^k,\boldsymbol{y}^l)/\varsigma)^{-\frac{\varsigma+1}{2}}}$	minimize Kullback-Leibler divergence
(t-)NeRV	probab. $p_{j\|i}$ as for SNE or p_{ij} as for t-SNE	probab. $q_{j\|i}$ as for SNE or q_{ij} as for t-SNE	minimize sum of Kullback-Leibler divergences with weight $\lambda \in [0, 1]$

How can these results of a random subsample be used to inspect the full data set? One possibility is to add additional points on demand by means of out-of-sample extensions of the techniques. The general view as presented above offers a very easy way to describe the principle of out-of-sample extensions which are built on top of fixed dimensionality reduction mappings of a subset S of all data: Assume, the projections $\boldsymbol{y}^i$ of data $\boldsymbol{x}^i \in S$ are fixed. A further point $\boldsymbol{x}$ can be mapped to coefficients $\boldsymbol{y}$ by optimizing $\text{error}(\text{char}(\boldsymbol{x}), \text{char}(\boldsymbol{y}))$ whereby the coefficients $\boldsymbol{y}^i$ for elements in S are kept fixed. Depending on the method at hand, an explicit algebraic solution or numeric optimization are possible.

Assuming a deterministic optimization method for simplicity, this is essentially a way to determine a *function* of the full data space to the projection space $f : \mathbb{R}^N \to \mathbb{R}^2$ by means of an implicit formula: a data point $\boldsymbol{x}$ is mapped to the coefficients which minimize the cost function as specified above. Depending on the method at hand, f might have a complex form and its computation might be time consuming, albeit properties such as piecewise differentiability and smoothness follow from the smoothness of the cost function.

Explicit Dimensionality Reduction Mapping

We can avoid the computational complexity and complex form of such implicit function f by the definition of an *explicit* dimension reduction mapping $f : \mathbb{R}^N \to \mathbb{R}^2$, $\boldsymbol{x}^i \to \boldsymbol{y}^i = f(\boldsymbol{x}^i)$ with priorly fixed form. The formalization of dimensionality reduction as cost optimization allows to immediately extend the techniques to this setting: function parameters can be optimized according to the objective as specified by the respective dimensionality reduction method. That means, we fix a parameterized form $f_W : \mathbb{R}^N \to \mathbb{R}^2$ with parameters W. This function can be given by a linear function, a locally linear function, a feedforward neural network, etc. Then, instead of coefficients $\boldsymbol{y}^i$, the images of the map $f_W(\boldsymbol{x}^i)$ are considered and the map parameters W are optimized such that the costs

$$\text{error}(\text{char}(\boldsymbol{x}), \text{char}(f_W(\boldsymbol{x})))$$

become minimal. This principle leads to a well defined mathematical objective for the mapping parameters W for every dimensionality reduction method as summarized above. The way in which optimization takes place is possibly different as compared to the original method: while numerical methods such as gradient descent can still be used, it is probably no longer possible to find closed form solutions for spectral methods.

We can train a dimensionality reduction mapping for only a random subsample S of the data, providing an explicit out-of-sample extension for all data points by means of the explicit mapping. Hence this technique offers a constant time inference of a dimensionality reduction mapping provided S has fixed size.

In the literature, a few dimensionality reduction technologies with explicit mapping of the data can be seen as instantiations of this principle: Locally linear coordination (LLC) [14] extends locally linear embedding (LLE) by assuming locally linear dimensionality reduction methods, e.g. local PCAs, and glueing them together adding affine transformations. The additional parameters

are optimized using the LLE cost function. Parameterized t-distributed stochastic neighbor embedding (t-SNE) [17] extends t-SNE towards an embedding given by a multilayer neural network. The network parameters are determined using back propagation on top of the t-SNE cost function.

Supervised Locally Linear t-SNE Mapping

Here, we include one preliminary example to demonstrate the feasibility of the approach. We use t-SNE as dimensionality reduction method, and a locally linear function f_W induced by prototypes. We start with locally linear projections of the data obtained by means of a supervised prototype based method, in our case matrix learning vector quantization with rank two matrices [13,4]. These give us locally linear projections $\boldsymbol{x}^l \mapsto p_k(\boldsymbol{x}^l) = \Omega_k \boldsymbol{x}^l - \boldsymbol{w}^k$ with local matrices Ω_k and prototypes $\boldsymbol{w}^k$. Further, we obtain responsibilities r_{lk} of mapping p_k for $\boldsymbol{x}^l$, given by the receptive fields. Then a global mapping can be defined as

$$f_W : \boldsymbol{x}^l \mapsto \boldsymbol{y}^l = \sum_k r_{lk}(L_k \cdot p_k(\boldsymbol{x}^l) + l_k) \ ,$$

using local linear projections L_k and local offsets l_k to align the local pieces. The parameters L_k and l_k are determined using the t-SNE cost function.

Obviously, since we start from a supervised clustering, the resulting function is biased towards good discriminative properties. We compare the results of this technique to several state of the art supervised dimensionality reduction tools as reported in [19] on three benchmarks from [1,8] (see also [19]). For all settings, we use only a fraction of about 10% for training, extending to the full data set by means of the explicit mapping (unlike the results as reported in [19] which evaluate on a subset of the data only). The obtained classification accuracy by means of nearest neighbor classification is reported in Fig. 1, showing that the method leads to excellent results.

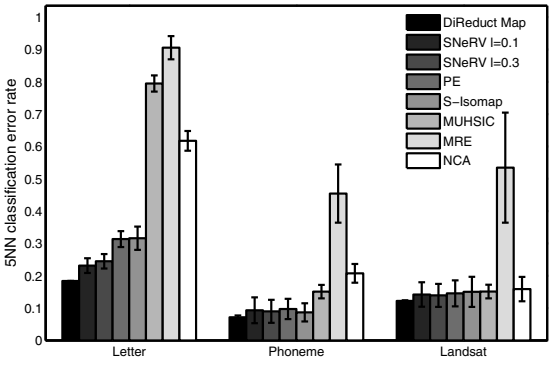


Fig. 1. Comparison of the 5 nearest neighbor errors for all data sets

Dimensionality Reduction and Clustering

Sometimes, large data sets are treated by prior clustering instead of random subsampling, whereby clusters are defined by receptive fields of prototypes. Subsequently, the cluster centers are projected to low dimensions. This procedure can be seen as a special case of the general framework as proposed above: for clustering, the function f_W is locally constant, whereby, depending on the type of clustering, the local pieces are given by receptive fields of prototypes; hence, assuming cluster centers $\boldsymbol{w}^i$ and projections $\boldsymbol{y}(\boldsymbol{w}^i)$, the function has the form

$$f_W(\boldsymbol{x}) \mapsto \boldsymbol{y}(\boldsymbol{w}^i) \text{ where } \|\boldsymbol{x} - \boldsymbol{w}^i\|^2 \text{ is minimum.}$$

Usually, the clustering is done separately from the dimensionality reduction. The general view as introduced above proposes a slightly optimized version: rather than optimizing only the projections $\boldsymbol{y}(\boldsymbol{w}^i)$, also the cluster centers could be adapted according to the general costs. One drawback of this proposal is given by the fact that assignments of data points to clusters are discrete such that simple gradient techniques are no longer applicable. A test of alternative optimization technique in this setting is the subject of ongoing work.

4 Generalization Ability

By treating dimensionality reduction as cost optimization, it is possible to extend dimensionality reduction techniques to large data sets based on a subsample S: either in terms of an implicit embedding function induced by posterior out-of-sample extensions, or in terms of an explicit embedding function with fixed form which parameters are optimized using S. In both cases, we obtain an embedding function f based on S which extends to arbitrary points $\boldsymbol{x}$ in terms of an implicit or explicit mapping. This allows the application to large data sets since the learning of the mapping itself is constant time, assuming a fixed size of S.

Now the question occurs whether this procedure can be substantiated by mathematical guarantees. The essential property which should be guaranteed is the *generalization ability*: assumed the mapping is satisfactory for the training set S, do we have any guarantees that it behaves well for arbitrary $\boldsymbol{x}$? taken according to the same probability as S? That means we have to ensure that the quality measure for all data is good assumed it is good for a given finite subsample used to determine the mapping parameters.

Recently, some work on a formal evaluation of dimensionality reduction has been proposed [10,19]. These evaluation measures rely on the measurement of local neighborhoods and their preservation while projecting the data. As such they are not directly suited as evaluation measures for a function since they rely on a finite sample only. Further, it is not clear whether they can adequately capture essential properties of a mapping f, see e.g. for problems when they are used to evaluate clustering [11].

As pointed out in [10], one objective of dimensionality reduction is to preserve the available information as much as possible. In consequence, the possibility to reconstruct the points $\boldsymbol{x}^i$ from their projections $\boldsymbol{y}^i$ can act as valid evaluation

measure. A drawback of this evaluation measure is that, usually, an explicit mapping and the reconstruction of data are not given for dimensionality reduction techniques which offer embedding coordinates for a finite data set only. Assuming a dimensionality reduction mapping $f : \mathcal{X} \to \mathcal{E}$ is given, an explicit formalization of this error measure becomes possible: the reconstruction error is

$$E(P) := \int_{\mathcal{X}} \|\boldsymbol{x} - f^{-1}(f(\boldsymbol{x}))\|^2 P(\boldsymbol{x}) d\boldsymbol{x}$$

where P defines the probability measure according to which the data $\boldsymbol{x}$ are distributed in $\mathcal{X}$ and f^{-1} constitutes an approximate inverse mapping of f, an exact inverse in general not existing. Thus, this objective allows us to evaluate dimensionality reduction mappings. In practice, of course, the full data manifold is not available, but a finite sample set only. In this case, the empirical error can be computed

$$\widehat{E}_n(\boldsymbol{x}) := \frac{1}{n} \sum_i \|\boldsymbol{x}^i - f^{-1}(f((\boldsymbol{x}^i)))\|^2$$

for a given data set $S = \{\boldsymbol{x}^1 \ldots, \boldsymbol{x}^n\}$. Now, a good generalization ability of a dimensionality reduction method can be formalized as the empirical error $\widehat{E}_n(\boldsymbol{x})$ being representative for the true error $E(P)$ for the dimensionality reduction f.

This setting can be captured in the classical framework of computational learning theory, as specified e.g. in [2]. We can adapt Theorem 8 from [2] to our setting: We consider a fixed function class

$$\mathcal{F} : \mathcal{X} \to \mathcal{E}$$

from which the dimensionality reduction mapping is taken. Note that, in every case as specified above, the form of the embedding function can be fixed: either it is given explicitly, e.g. as locally linear function or by means of a clustering, or it is given implicitly by means of a local optimum of a cost function. We assume without loss of generality, that the norm of the input data and its reconstructions under mappings $f^{-1} \circ f$, f^{-1} denoting the approximate inverse of $f \in \mathcal{F}$, are restricted (scaling the data priorly, if necessary), such that the reconstruction error is induced by the squared error, which is a loss function with limited codomain

$$\mathcal{L} : \mathcal{X} \times \mathcal{X} \to [0, 1], (\boldsymbol{x}^i, \boldsymbol{x}^j) \mapsto \|\boldsymbol{x}^i - \boldsymbol{x}^j\|^2$$

Then, as reported in [2] (Theorem 8), assuming i.i.d. data according to P, for any confidence $\delta \in (0, 1)$ and every $f \in \mathcal{F}$ the following holds

$$E(P) \leq \widehat{E}_n(\boldsymbol{x}) + R_n(\mathcal{L}_{\mathcal{F}}) + \sqrt{\frac{8 \ln(2/\delta)}{n}}$$

with probability at least $1 - \delta$ where

$$\mathcal{L}_{\mathcal{F}} := \{\boldsymbol{x} \mapsto \mathcal{L}(f^{-1}(f(\boldsymbol{x})), \boldsymbol{x}) \mid f \in \mathcal{F}\}$$

and R_n refers to the so-called Rademacher complexity of the function class.

The Rademacher complexity constitutes a quantity which, similar to the Vapnik Chervonenkis dimension, estimates the capacity of a given function class. We do not include its exact definition, rather, we refer to [2]. Note, however, that the Rademacher complexity of reasonable function classes (such as piecewise constant, piecewise linear functions, or polynomials of fixed degree) can be limited by a term which scales as $n^{-1/2}$, as long as the function class does not have infinite capacity e.g. due to an unlimited number of free paramters (e.g. polynomials with unbounded degree). See [2] for structural results and explicit bounds for e.g. linear functions, and e.g. [13] for explicit bounds on piecewise constant functions as induced by prototype based clustering. This result implies that the generalization ability of dimensionality reduction mappings is usually guaranteed since the Gaussian complexity of the class $\mathcal{L}_{\mathcal{F}}$ can be limited for reasonable choices of the mapping function $\mathcal{F}$. It remains a subject of future research to find explicit and good bounds. for concrete $\mathcal{F}$ as occur in standard methods.

5 Conclusion

We have introduced a general way to formalize dimensionality reduction which includes different general techniques to extend dimensionality reduction to large data sets: subsampling and out-of sample extensions by an implicit mapping, the training of an explicit mapping based on a subsample, or clustering and projection as a special case thereof. We demonstrated the feasibility for one preliminary example, and included first steps towards a formalization of the generalization ability of these approaches.

Acknowledgment. This work was supported by the "Nederlandse organisatie voor Wetenschappelijke Onderzoek (NWO)" under project code 612.066.620 and by the "German Science Foundation (DFG)" under grant number HA-2719/4-1. Further, financial support from the Cluster of Excellence 277 Cognitive Interaction Technology funded in the framework of the German Excellence Initiative is gratefully acknowledged.

References

1. Asuncion, A., Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998), http://archive.ics.uci.edu/ml/ (last visit June 19, 2009)
2. Bartlett, P.L., Mendelson, S.: Rademacher and gaussian complexities: risk bounds and structural results. J. Mach. Learn. Res. 3, 463–482 (2003)
3. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation 15, 1373–15396 (2003)
4. Bunte, K., Hammer, B., Wismüller, A., Biehl, M.: Adaptive local dissimilarity measures for discriminative dimension reduction of labeled data. Neurocomputing 73(7-9), 1074–1092 (2010)
5. Carreira-Perpiñán, M.Á.: The elastic embedding algorithm for dimensionality reduction. In: 27th Int. Conf. Machine Learning (ICML 2010), pp. 167–174 (2010)

6. Hinton, G., Roweis, S.: Stochastic neighbor embedding. In: Advances in Neural Information Processing Systems 15, pp. 833–840. MIT Press, Cambridge (2003)
7. Keim, D.A., Mansmann, F., Schneidewind, J., Thomas, J., Ziegler, H.: Visual analytics: Scope and challenges. In: Simoff, S.J., Böhlen, M.H., Mazeika, A. (eds.) Visual Data Mining. LNCS, vol. 4404, pp. 76–90. Springer, Heidelberg (2008)
8. Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J., Torkkola, K.: LVQ-PAK: The learning vector quantization programm package. Technical Report A30, Helsinki University of TechnologyLaboratory of Computer and Information Science, FIN-02150 Espoo, Finland (1996)
9. Lee, J., Verleysen, M.: Nonlinear dimensionality reduction, 1st edn. Springer, Heidelberg (2007)
10. Lee, J.A., Verleysen, M.: Quality assessment of dimensionality reduction: Rank-based criteria. Neurocomput. 72(7-9), 1431–1443 (2009)
11. Mokbel, B., Gisbrecht, A., Hammer, B.: On the effect of clustering on quality assessment measures for dimensionality reduction. In: NIPS workshop on Challenges of Data Visualization (2010)
12. Roweis, S.T., Saul, L.K.: Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science 290(5500), 2323–2326 (2000)
13. Schneider, P., Biehl, M., Hammer, B.: Adaptive relevance matrices in learning vector quantization. Neural Computation 21(12), 3532–3561 (2009)
14. Teh, Y.W., Roweis, S.: Automatic alignment of local representations. In: Advances in Neural Information Processing Systems 15, pp. 841–848. MIT Press, Cambridge (2003)
15. Tenenbaum, J.B., Silva, V.d., Langford, J.C.: A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science 290(5500), 2319–2323 (2000)
16. van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of Machine Learning Research 9, 2579–2605 (2008)
17. van der Maaten, L.J.P.: Learning a parametric embedding by preserving local structure. In: Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AI-STATS), 5, pp. 384–391. JMLR W&CP (2009)
18. van der Maaten, L.J.P., Postma, E.O., van den Herik, H.J.: Dimensionality reduction: A comparative review. Technical Report TiCC-TR 2009-005, Tilburg University (October 2009)
19. Venna, J., Peltonen, J., Nybo, K., Aidos, H., Kaski, S.: Information retrieval perspective to nonlinear dimensionality reduction for data visualization. J. Mach. Learn. Res. 11, 451–490 (2010)
20. Weinberger, K.Q., Saul, L.K.: An introduction to nonlinear dimensionality reduction by maximum an introduction to nonlinear dimensionality reduction by maximum variance unfolding. In: Proceedings of the 21st National Conference on Artificial Intelligence (2006)

Considering Spatialization Aspects When Positioning Input Vectors on Self-Organizing Map Output Grids

Tonio Fincke

Lab for Geoinformatics and Geovisualization, HafenCity Universität Hamburg, Hebebrandstr. 1, 22297 Hamburg, Germany
tonio.fincke@hcu-hamburg.de
http://www.geomatik-hamburg.de/g2lab/

Abstract. Several techniques have been put forward to describe characteristics of a Self-Organizing Map by depicting them on its output grid. These techniques form artificial landscapes, which are also called spatializations. Until now, relatively few methods exist for displaying distinct input vectors on the output grid. Those who exist either do not show patterns in the distribution of the input vectors or do not observe the spatial metaphors implied by the spatializations. This paper proposes an attempt to fill this gap. An approach is introduced where the input vector placement can be influenced by two parameters. The placement technique is tested with two data sets and analyzed through visual inspection. The results show that the approach can both indicate patterns in the input data as well as observe the spatial metaphors of the spatializations. It thereby allows for a meaningful combination of these visualization forms.

Keywords: Self-Organizing Maps, Input Vector Placement, Spatialization, Spatial Metaphor.

1 Introduction

Self-organizing maps (SOMs) are neural networks which are frequently used for the clustering and linear quantization of large, high-dimensional data sets [1]. An outstanding characteristic of a SOM is its ability to display its results visually. Since a topological order is defined over the codebook vectors, they can be depicted as cells of an output grid. Thus a SOM allows that after the computational part of the data mining a visual analysis of the results can take place. This is one of the main reasons why numerous tools and techniques for visualizing various characteristics of a SOM have been developed. Mostly, they display an additional color coded value on the cells of the SOM output grid. For example, a u-matrix shows the distances between the various codebook vectors [2]. Dark values indicate large average distances between adjacent codebook vectors, bright values stand for small distances. Such small distances between codebook vectors indicate clusterings of input vectors. When a u-matrix is displayed, the bright

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 288–297, 2011.

and dark values often resemble valleys and hills[1]. Clusters are then expected to appear in the valleys, which are separated by hills. Therefore, a u-matrix also is a spatialization.

A spatialization is a graphic representation of information using spatial metaphors. Such a spatial metaphor is a cognitive relationship between a spatial and a non-spatial property [3,4]. Spatial metaphors have proven to be intuitively understandable [5]. When a landscape metaphor is used, information is presented in the form of a 2- or 3-dimensional landscape. A landscape metaphor is composed of several other spatial metaphors, such as the distance-dissimilarity metaphor. This metaphor obeys the first law of geography that "everything is related to everything else, but near things are more related than distant things" [6]. In the case of 3-dimensional landscapes, such as the u-matrix, also a height-metaphor is employed, since the information is communicated through the visual impression of a relief. But also other SOM visualization techniques like component planes [1] or p-matrices [7] make use of such metaphors in order to communicate their information. The SOM output grid itself is an example for a 2D-landscape, since it maps codebook vectors which are similar in input space close together on the output grid [8,9]. These characteristics have led to an increased attention of the SOM in the GIScience Community. Several publications deal with the application of methods for spatial analysis on SOMs [10].

Whilst much effort has been undertaken to develop visualization techniques to describe the characteristics of a SOM and therefore its underlying data set, relatively few attempts have been made to depict the input data vectors on the output grid. However, for several applications it is helpful to provide a link back to the input data. For example, SOMs can be used to visualize large archives of data like, e.g., scientific papers [11]. In such archives, similar papers are mapped to similar Best Matching Units (BMUs). Such library SOMs often use u-matrices to make it easier for the user to find clusters of codebook vectors. This approach is less useful when the input vectors, i.e., the scientific papers, are not mapped onto the output grid. Often this limitation has been overcome by instead linking the codebook vectors back to the input data or to other visualization forms.

In section 2, several prior attempts to visualize input vectors on a SOM output grid are introduced. In section 3, an approach is presented to position input vectors in such a way that they conform to the landscape metaphor of a SOM. This approach is tested against two data sets by a visual inspection of their u-matrices. The tests are presented in section 4 and discussed in section 5. Section 6 concludes the paper.

2 Related Works

One of the earliest attempts to visualize input vectors on a SOM output grid was the usage of a SOM with more codebook vectors than input vectors [9].

[1] Please note that this assignment of gray values is reversed in some u-matrices. Such a color coding is also used for grayscale geographic maps. For reasons of legibility, in this paper valleys are bright and hills are dark.

The training of the SOM would cause some of the codebook vectors to adopt the values of one of the input vectors, whilst some codebook vectors would not be the BMU for any input vector. These empty grid cells would then indicate inter-vector distances. This attempt has the obvious disadvantage of not being compatible with other visualization techniques. Also, both the clustering and linear quantization capability of the SOM get lost.

In [11], the input vectors are placed randomly in the vicinity of their BMU. This placement technique has the advantage that the depiction of the input vectors complies with the spatial metaphors used by other visualization techniques. When, e.g., a BMU is located in the valley of a u-matrix, also the input vectors are located in this valley. They are quickly visible as members of the corresponding cluster. However, since the placement of the input vectors is random, inter-input vector distances are meaningless, thus the first law of geography is violated. The placement therefore might trick an analyst into believing that there exist relationships between input vectors which actually do not exist.

A third input vector placement method is the weighted response placement [12]. In this technique, not only the BMU but all codebook vectors are considered in the process of determining a position for a distinct input vector. For a m*n-SOM, the position p_x of a distinct input vector x is

$$p_x = \sum_{i=1}^{m} \sum_{j=1}^{n} R_{i,j}(x) o_{i,j} \tag{1}$$

where $o_{i,j}$ is the position of a codebook vector $w_{i,j}$ on the output grid. This position is the center of the output grid cell associated with the codebook vector. $R_{i,j}(x)$ is the response of $w_{i,j}$ to x and is given by

$$R_{i,j}(x) = \frac{g(q_{i,j}) N_c(i,j)}{\sum_{k=1}^{m} \sum_{l=1}^{n} g(q_{k,l}) N_c(k,l)}, i \in [1,m], j \in [1,n] \tag{2}$$

where c is the BMU and $N_c(j)$ is a neighborhood function around c. The purpose of the neighborhood function is to indicate which codebook vectors should be considered. $g(q_{i,j})$ is a weighting function decreasing monotonically in the interval between 0 and 1 and $q_{i,j}$ is the quantization error for a codebook vector $w_{i,j}$. The weight function indicates to which degree the codebook vectors should be included into the calculation. Both the neighborhood and the weighting function will decrease with increasing distance from the BMU.

In [12] the neighborhood around the BMU was kept very broad and included codebook vectors from all over the grid in the calculation process. Although in the result the inter-input vector distances were preserved in the output space and patterns of the input data became visible, the input vectors were often positioned between several well matching units. When this approach was combined with a u-matrix, input vectors were often not placed in the valleys or ridges associated with their respective BMUs, thus hurting the landscape metaphor.

This attempt showed similarities to other methods from multi-dimensional scaling like Sammon's Mapping [13], where high-dimensional vectors are mapped onto 2-dimensional output grids.

3 Outline of the Input Vector Placement Approach

The previous section conveys that there is yet no technique which positions input vectors in such a way that that their depictions form a coherent spatialization with other techniques as, e.g., u-matrices and at the same time indicate patterns in the distribution of input vectors. Therefore the aim of this section is to find a method to position the vectors in such a way that (a) the relational distances amongst them are mostly preserved but (b) they will not be placed far away from their respective BMUs on the output grid.

Since the approach from [12] allows for the alteration of both its neighborhood and weighting function, not a completely new method was developed. Instead, various parameter combinations of the weighted response placement approach with different data sets were tested and the results were analyzed visually.

Before the analysis began, some slight alterations of the approach were performed. The weighting function was changed to

$$g(x, w_{i,j}) = \exp\left(-\frac{\parallel x - w_{i,j} \parallel}{r}\right)^2 \tag{3}$$

for an input vector x and a codebook vector $w_{i,j}$. This applies a parameter r which can be altered by the analyst. Small values of r will lead to smaller weights for codebook vectors which are far away from the input vectors, while large values will cause that they receive larger weights.

Maybe even more crucial is the definition of the neighborhood function. The attempt presented here is to either include a codebook vector completely or not at all. This is achieved by including the best matching unit and the units in its surrounding. The neighborhood function is therefore defined as

$$N_c(i,j) = \begin{cases} 1, & \text{if } w_{i,j} \in Nh_c(k) \\ 0, & \text{else} \end{cases} \tag{4}$$

where $Nh_c(k)$ is the set of codebook vectors around the BMU c to be considered. The parameter k is used to control how many neighbors to include. For $k = 0$, only the BMU is considered. For $k = 1$, the BMU and its immediate neighbors are used. For $k = 2$, also the immediate neighbors of the BMU's immediate neighbors are included, and so on.

4 Tests

For testing, the input vectors were not merely placed on SOM output grids, but on u-matrices. There are two reasons for doing this: First, since the u-matrix employs a height metaphor, the spatialization is an extension of the SOM grid landscape. Any placement which forms a coherent spatialization with the u-matrix will do so as well with the plain output grid. Second, in this manner the point clusterings can be compared to the clusterings indicated by u-matrix valleys. The test itself consisted of examining whether (a) the input vector position distributions formed meaningful patterns and (b) the placement of the input

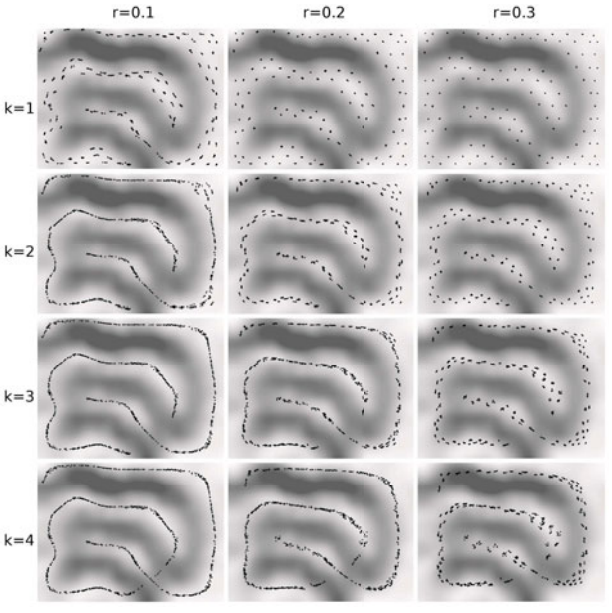


Fig. 1. Various placements of input vectors of the Chainlink data set on a u-matrix for its SOM

vectors corresponded to the landscape metaphor of the u-matrix. This would be the case when input vector clusters would be placed in u-matrix valleys.

The placement technique was tested against two different SOMs: one with artificial data, one with real-life data. In both cases, the input vector values were normalized onto the interval between 0 and 1 before the SOMs were created. Both SOMs have hexagonal topologies. The sizes of the grids were chosen such that the total number of codebook vectors would be roughly about $5 * \sqrt{n}$ where n is the number of input vectors. The ratio between the grid dimensions corresponds to the ratio between the two largest eigenvectors in the input data. This procedure is also applied in the SOM toolbox [14]. By altering the parameter r of the weight function and the parameter k of the neighborhood function, various position sets for the input vectors were derived.

The first SOM was created for the artificial Chainlink data set from the Fundamental Clustering Problem Suite (FCPS) [15]. Here, the input vectors are arranged as two clearly separated, but not linearly separable clusters. These two clusters form two intertwining rings in three-dimensional space. The SOM for this data set is a 13*12 SOM. The results for neighborhoods up to $k = 4$ and weights up to $r = 0.3$ are displayed on a u-matrix in figure 1.

The u-matrix shows explicit hills and valleys. The input vectors are mostly placed in those valleys. When the neighborhood k is set to 1, only the immediate topological neighbors of the BMU are considered and the input vectors will be placed in the area around the codebook vector. When r is increased, the position values of the input vectors with the same BMU become very similar. The third

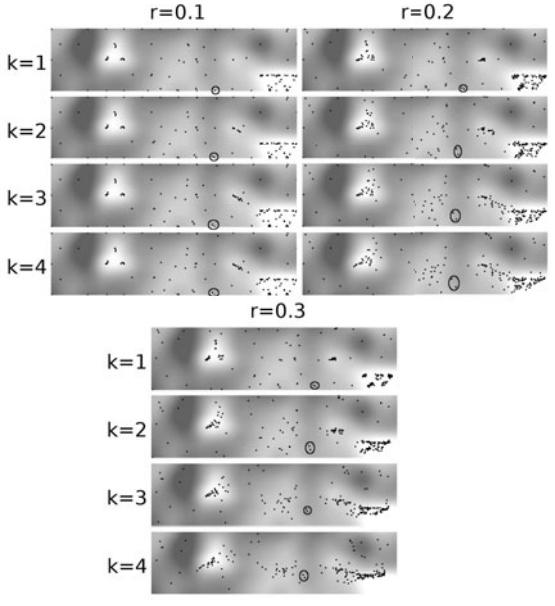


Fig. 2. Various placements of input vectors of the WHO data set on a u-matrix for its SOM. Circles are drawn around the input vectors of Ghana and Sudan.

depiction in the top row almost indicates the positions of the codebook vectors—at least for those which are not too close to the border of the grid. The placements in the bottom row show the effect of a large neighborhood value. In all depictions of that row, some of the input vectors are not only placed on a u-matrix hill, but seem to merge with input vectors from other valleys. Whilst this indicates that the clusters form rings in input space, it does not harmonize with the landscape laid out by the u-matrix. Also, note that in the most-right depictions the input vectors seem to be drawn to the center of the map. This is because the centroid of the included codebook vectors shifts towards the map center for increased values of k.

Arguably the best placements are given by the four depictions with $k \in [2, 3]$ and $r \leq 0.2$. In these depictions, the input vectors are clearly placed in the valleys of the u-matrix and at the same time form patterns.

The second set consists of data from the World Health Organization (WHO) [16]. It is comprised of 192 states with a total of 10 variables (population growth rate, total population, total urban population, male child mortality, female child mortality, cases of tuberculosis, male adult mortality, female adult mortality, neonatal mortality, and average life expectancy). The SOM for this data set is a 14*5 SOM. The results for neighborhoods up to $k = 4$ and weights up to $r = 0.3$ are displayed on a u-matrix in figure 2.

The u-matrix shows two salient valleys, one in the southeast (valley 1) and one to the west (valley 2). About half of the input vectors have been mapped to valley 1. It consists mainly of European states, but also of several states from

South America, Eastern Asia, and the Middle East. 23 African states plus Haiti were placed in valley 2. Whilst not as clearly designated as a valley by the u-matrix, the vector placement shows a third big accumulation to the northwest of valley 1. The input vectors clustered in this valley (valley 3) are a blend of states from Eastern Europe, several small island states, Southern Asia, and Egypt.

Again, for $k = 1$ the vectors are placed close to the positions of their BMUs. For $r = 0.1$ the input vector placements do not show patterns—the only remarkable event is the alignment of the input vectors from valley 3 towards valley 1. The several individual cases scattered all over the map do not change their positions. In all depictions, the two eastern clusters meet when $k = 3$. This also means that the input vectors are not located in their respective valleys, but are placed on top of hills, thereby violating the rules of the landscape metaphor.

In valley 1 the input vectors form sub-clusters. Input vectors placed in the northern part of valley 1 consist mainly of European states. This north-south divide is visible best for $k = 1$ or $r \leq 0.2$. In some of the depictions (best visible for $k = 3$ and $r = 0.2$) also a divide between the eastern and western part becomes apparent. In the eastern part, mainly states from Western Europe can be found, whilst the western part consists of many Eastern European states. Between valley 2 and valley 3 a large plain is situated which is separated by a chain of hills from the valleys. Distinct input vectors which are placed on the hills between the plain and valley 3 for $k = 1$ are located closer to either the plain or valley 3 when k is increased. In this context, note the two input vectors close to the bottom of the map (Ghana and Sudan, emphasized in figure 2 by circles) which lie in the middle between the plain and the valley, but show a tendency towards the plain for $k \geq 2$ and $r \geq 0.2$. The input vectors placed in valley 2 do not show any distinguishable patterns, but clearly form a cluster within the valley for $r \geq 0.2$ and $k \geq 1$.

It is also apparent that the patterns of the input vectors in valley 1 seem to dissolve for $k = 4$. For this increased neighborhood, the clusters start to merge and to move away from the valley. Also, the input vectors wander away from the borders to the middle of the map.

Arguably the best depiction is delivered by the map with $k = 2$ and $r = 0.2$. Here, input vectors with BMUs within the same valley form sub-clusters, but are not placed on hills or get mixed up with input vectors with BMUs from other valleys. Also, most hills are actually void of input vectors. Therefore, the landscape metaphor of the u-matrix is observed here.

5 Discussion of the Results

From these results, it can be seen that already small neighborhoods can cause placements of input vectors on hills and therefore violations of the landscape metaphor. A neighborhood of $k = 1$ will cause the input vectors to be placed around its BMU. The vectors will only be comparable to vectors with the same BMU, but not to others, even when their BMUs are topological neighbors. One reason for this is that of the seven codebook vectors which are used for

the calculation of the position of an input vector, only four are used for input vectors from adjacent BMUs.

When $k = 2$, 19 codebook vectors are used for the position calculation, and 14 of them are also used for the calculation of an input vector with a neighboring BMU. This ratio converges to 1 with an increasing k, but the amount of shared codebook vectors for $k = 2$ is sufficient to show similarities or dissimilarities between input vectors from different BMUs.

Another issue is the centering effect: When the neighborhood is increased, input vectors tend to be placed at the center of the map. A way to prevent this might be to assign extra weights to codebook vectors at the borders. However, this would also lead to an unjustifiedly strong accentuation of those codebook vectors and therefore distort the input vector distribution. Due to these findings, a neighborhood of $k \in [2, 3]$ seems preferable.

When r is increased, the calculated positions for input vectors with the same BMU become similar. This effect is alleviated when the neighborhood k is also increased. The distribution of the input vectors starts to show patterns then. However, some of the input vectors with BMUs placed in valleys are then positioned on hills. A striking difference in the performances of the two data sets is that for $r = 0.1$ the SOM for the Chainlink data showed a clear alignment of the input vectors, whilst except for the input vectors in valley 3 in the SOM for the WHO data set input vectors with different BMUs did not seem to form some sort of pattern. Patterns most clearly emerged for $r = 0.2$.

For the Chainlink data set, the actual pattern (two intertwined rings) was known and clearly distinguishable. Therefore it could easily be seen that the input vectors aligned in a way that resembled this pattern from input space. Doing so was harder for the WHO data set, because the underlying distribution was not known. However, also this data set showed patterns: Inter-differences between distinct input vectors or groups of vectors became apparent and sub-clusters located in valleys were indicated. Also, the placement of input vectors emphasized valley 3, which would not have been so easily to distinguish by merely inspecting the u-matrix. The distances between different input vectors or groups of input vectors have proven here to tell about the underlying distribution in input space. This means that for certain parameter combinations the spatial metaphor was obeyed and applied successfully. Of course, the parameter combinations which worked best here might not be the optimal solutions for other SOMs. The trial of other combinations is therefore encouraged; especially when one is dealing with data sets which are very different from those presented in this work.

6 Conclusion

In this paper an approach was presented to position input vectors on a SOM output grid in such a way that they can form a coherent spatialization with other SOM visualization techniques. This approach rests upon the weighted response placement [12]. The aim was to make alterations to this placement technique so that the placement of the input vectors was parameterized through a neighborhood function and a weighting function. These functions determine the influence

of the various codebook vectors on the calculation of a distinct input vector's position. Different parameter combinations were applied and the input vectors were placed on the u-matrices of SOMs for two different data sets, one with artificial and one with real-life data. The resulting placements were inspected visually. It was shown that with a considerate choice of parameters combinations of the two visualization forms could be achieved in which (a) the input vector distributions on the output grid formed patterns and (b) input vector position clusters were positioned in the valleys of u-matrices. This means that the landscape metaphor employed by the u-matrix and consequently by the SOM output grid was observed. In some cases, the explanatory power of the u-matrix was complemented by the input vector placements, since the distributions also allowed for the visual detection of sub-clusters within u-matrix valleys.

It should be noted that inter-input vector distances are not necessarily preserved. Especially when there are non-linear relationships in the data, codebook vectors with similar values might be placed on faraway places on the map. Since only topologically close codebook vectors are considered for the computation, the relationship between similar input vectors can not be seen when their BMUs are situated at differing regions of the grid. If, however, similar codebook vectors are topologically close, different input vectors are placed at different locations and dissimiliarities between them become visible in the form of spatial distances.

Until now, only a visual analysis of the results has been performed. Since these results were promising, it seems appropriate to continue with a systematic study. To do so, special measures are needed in order to quantify the performance of various parameter combinations. These could include distances between input vector positions, comparisons between distances in input and output space, or the u-height at an input vector's position.

The spatialization-aware placement of input vectors allows for the application of algorithms for spatial analysis. For instance, buffers can be drawn around input vectors to detect similar data items. Also, other visualization techniques than u-matrices might be used. When the vector placement technique is combined with, e.g., component planes, this could serve for the estimation of missing values in the input data. These examples form only a part of potential future work.

References

1. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (2001)
2. Ultsch, A., Siemon, H.P.: Kohonen's Self Organizing Feature Maps for Exploratory Data Analysis. In: Proceedings of International Neural Network Conference (1990)
3. Kuhn, W., Blumenthal, B.: Spatialization: Spatial Metaphors for User Interfaces. Department of Geoinformation, Technical University of Vienna, Vienna, Austria (1996)
4. Skupin, A., Buttenfield, B.P.: Spatial Metaphors for Visualizing Very Large Data Archives. In: Proceedings GIS/LIS 1996, pp. 607–617 (1996)
5. Tory, M., Sprague, D.W., Wu, F., So, W.Y., Munzner, T.: Spatialization Design: Comparing Points and Landscapes. IEEE Transactions on Visualization and Computer Graphics 13(6), 1262–1269 (2007)

6. Tobler, W.R.: A Computer Movie Simulating Urban Growth in the Detroit Region. Economic Geography 46(2), 234–240 (1970)
7. Ultsch, A.: Density Estimation and Visualization for Data containing Clusters of unknown Structure. In: Weihs, C., Gaul, W. (eds.) Classification - the Ubiquitous Challenge, Proceedings 28th Annual Conference of the German Classification Society (GfKl 2004), pp. 232–239. Springer, Heidelberg (2005)
8. Bação, F., Lobo, V., Painho, M.: Geo-SOM and its integration with Geographic Information Systems. In: Proc. Workshop on Self-Organizing Maps, Paris, France (2005)
9. Skupin, A., Hagelman, R.: Attribute Space Visualization of Demographic Change. In: Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems (2003)
10. Agarwal, P., Skupin, A. (eds.): Self-Organising Maps: Applications in Geographic Information Systems. Wiley, Chichester (2008)
11. Skupin, A.: A Cartographic Approach to Visualizing Conference Abstracts. Computer Graphics and Applications. IEEE Computer Graphics and Applications 22(1), 50–58 (2002)
12. Liao, G., Shi, T., Liu, S., Xuan, J.: A Novel Technique for Data Visualization Based on SOM. In: Duch, W., Kacprzyk, J., Oja, E., Zadrożny, S. (eds.) ICANN 2005. LNCS, vol. 3696, pp. 421–426. Springer, Heidelberg (2005)
13. Sammon, J.W.: A Nonlinear Mapping for Data Structure Analysis. IEEE Transactions on Computers 18(5), 401–409 (1969)
14. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: SOM Toolbox for Matlab 5. Technical Report A57, Helsinki University of Technology (2000), http://www.cis.hut.fi/projects/somtoolbox/
15. Ultsch, A.: Clustering with SOM: U*C. In: Proc. Workshop on Self-Organizing Maps, Paris, France, pp. 75–82 (2005)
16. World Health Organization - Data and Statistics, January 2011 (2011), http://www.who.int/research/en/

Aircraft Engine Fleet Monitoring Using Self-Organizing Maps and Edit Distance

Etienne Côme[1], Marie Cottrell[2], Michel Verleysen[3], and Jérôme Lacaille[4]

[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] Université Catholique de Louvain, Machine Learning Group
Place du Levant 3, 1348 Louvain-La-Neuve, Belgium
michel.verleysen@uclouvain.be
[4] Snecma, Rond-Point René Ravaud-Réau,
77550 Moissy-Cramayel CEDEX, France
jerome.lacaille@snecma.fr

Abstract. Aircraft engines are designed to be used during several tens of years. Ensuring a proper operation of engines over their lifetime is therefore an important and difficult task. The maintenance can be improved if efficient procedures for the understanding of data flows produced by sensors for monitoring purposes are implemented. This paper details such a procedure aiming at visualizing in a meaningful way successive data measured on aircraft engines and finding for every possible request sequence of data measurement similar behaviour already observed in the past which may help to anticipate failures. The core of the procedure is based on Self-Organizing Maps (SOM) which are used to visualize the evolution of the data measured on the engines. Rough measurements can not be directly used as inputs, because they are influenced by external conditions. A preprocessing procedure is set up to extract meaningful information and remove uninteresting variations due to change of environmental conditions. The proposed procedure contains four main modules to tackle these difficulties: environmental conditions normalization (ECN), change detection and adaptive signal modeling (CD), visualization with Self-Organizing Maps (SOM) and finally minimal Edit Distance search (SEARCH). The architecture of the procedure and of its modules is described in this paper and results on real data are also supplied.

1 Introduction

During the flights, some on-board sensors measure many parameters related to the behavior (and therefore the health) of aircraft engines. These parameters are recorded and used at short and long terms for immediate action and alarm

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 298–307, 2011.

generation, respectively. In this work, we are interested in the long-term monitoring of aircraft engines and we want to use these measurements to detect any deviations from a "normal" behavior, to anticipate possible faults and to facilitate the maintenance of aircraft engines. This work presents a tool that can help experts, in addition to their traditional tools based on quantitative inspection of some relevant variables, to easily visualize the evolution of the engine health. This evolution will be characterized by a trajectory on a two-dimensional Self-Organizing Map. Abnormal aging and fault will result in deviations with respect to normal conditions. The choice of Self-Organizing Maps is motivated by several points:

- SOMs are useful tools for visualizing high-dimensional data onto a low-dimensional grid;
- SOMs have already been applied with success for fault detection and prediction in plants and machines (see [8] for example).

This article follows another WSOM paper [3] 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 Section 4.

2 Data

Measurements are collected on a set of I engines. On each engine $i \in \{1, \ldots, I\}$, n_i measurements are performed successively flight after flight; there is thus no guarantee that the time intervals between two measures are approximately equal. Each observation is denoted by Z_{ij}, where $i \in \{1, \ldots, I\}$ is the engine number and $j \in \{1, \ldots, n_i\}$ is the flight number.

Each vector Z_{ij} contains two kinds of variables: those which are strictly related to the behavior of the engine (fuel consumption, static pressure, ...), and those which are related to the environment (temperature, altitude, ...). Let the p engine variables be denoted by $Y_{ij}^1, \ldots, Y_{ij}^p$ and the q environmental variables by $X_{ij}^1, \ldots, X_{ij}^q$. Each observation is therefore a $(p+q)$-vector Z_{ij}, where $Z_{ij} = [Y_{ij}, X_{ij}] = \left[Y_{ij}^1, \ldots, Y_{ij}^p, X_{ij}^1, \ldots, X_{ij}^q\right]$. The variables at disposal are listed in Table 1. There are $p = 5$ engine variables and $q = 15$ environmental variables. The dataset contains measurements for approximately one year of flights and $I = 91$ engines, that leads to a global dataset with $\sum_{i=1}^{91} n_i = 59407$ $(p+q)$-dimensional observations.

3 Methodology

The goal is to build the trajectories of all the engines, that is to project the successive observations of each engine on a Self-Organizing Map, in order to follow the evolution and to eventually detect some "abnormal" deviation. It is

Table 1. Variables names, descriptions and type

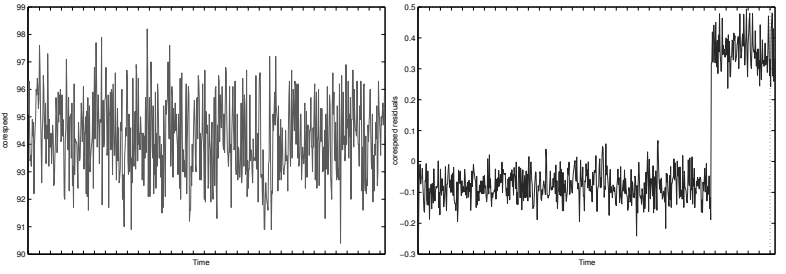
	Name	Description	Type	Binary
	aid	aircraft id		
	eid	engine id		
	fdt	flight date		
X_{ij}^1	temp	temperature	environment	
X_{ij}^2	nacelletemp	nacelle temperature	environment	
X_{ij}^3	altitude	aircraft altitude	environment	
X_{ij}^4	wingaice	wings anti-ice	environment	✓
X_{ij}^5	nacelleaice	nacelle anti-ice	environment	✓
X_{ij}^6	bleedvalve	bleed valve position	environment	✓
X_{ij}^7	isolationleft	valve position	environment	✓
X_{ij}^8	vbv	variable bleed valve position	environment	
X_{ij}^9	vsv	variable stator valve position	environment	
X_{ij}^{10}	hptclear	high pressure turbine setpoint	environment	
X_{ij}^{11}	lptclear	low pressure turbine setpoint	environment	
X_{ij}^{12}	rotorclear	rotor setpoint	environment	
X_{ij}^{13}	ecs	air cooling system	environment	
X_{ij}^{14}	fanspeedi	N1	environment	
X_{ij}^{15}	mach	aircraft speed	environment	
Y_{ij}^1	corespeed	N2	engine	
Y_{ij}^2	fuelflow	fuel consumption	engine	
Y_{ij}^3	ps3	static pressure	engine	
Y_{ij}^4	t3	temperature plan 3	engine	
Y_{ij}^5	egt	exhaust gas temperature	engine	

not valuable to use the rough engine measurements: they are inappropriate for direct analysis by Self-Organizing Maps, because they are strongly dependent on environment conditions and also on the characteristics of the engine (its past, its age, ...). The first idea is to use a linear regression for each engine variable: the environmental variables (real-valued variables) and the number of the engines (categorical variable) are the predictors and the residuals of these regressions can be used as standardized variables (see [3] for details). For each engine variable $r = 1, \ldots, p$, the regression model can be written as:

$$Y_{ij}^r = \mu^r + \alpha_i^r + \lambda_1^r X_{ij}^1 + \ldots + \lambda_q^r X_{ij}^q + \epsilon_{ij}^r \tag{1}$$

where α_i^r is the engine effect on the r^{th} variable, $\lambda_1^r, \ldots, \lambda_q^r$ are the regression coefficients for the r^{th} variable, μ^r is the intercept and the error term ϵ_{ij}^r is the residual.

Figure 1 presents for example the rough measurements of the *corespeed* feature as a function of time (for engine 6) and the residuals computed by model (1). The rough measurements seem almost time-independent on this figure, whereas the residuals exhibit an abrupt change which is linked to a specific event in the life of this engine. This simple model is therefore sufficient to bring to light interesting aspects of the evolution of this engine. However, the signals may contain



(a) (b)

Fig. 1. (a) Rough measurements of the **corespeed** variable as a function of time for engine 6, (b) residuals of the same variable and for the same engine using a simple linear model with the environmental variables and the engine indicator as predictors (see Table 1).

ruptures, making the use of a single regression model hazardous. The main idea of this work is to replace model (1) by a new procedure which deals with the temporal behavior of the signals. The goal is therefore to detect the ruptures and to use different models after each rupture. This new procedure is composed of two modules. The first module (Environmental Conditions Normalization, ECN) aims at removing the effects of the environmental variables to provide standardized variables, independent of the flight conditions. It is described in section 4.1. The second module uses an on-line change detection algorithm to find the above mentioned abrupt changes, and introduces a piecewise regression model. The detection of the change points is done in a multi-dimensional setting taking as input all the normalized engine variables supplied by the ECN module. The Change Detection (CD) module is presented in Section 4.2. As a result of these first two steps, the "cleaned" database can be used as input to a Self-Organizing Map with a "proper" distance for trajectories visualization. The third module (SOM) provides the "map" on which the trajectories will be drawn. Finally, engine trajectories on the map are gathered in a trajectory database which can be accessed through a SEARCH module, which use a dedicated Edit Distance to find similar trajectories. This four-steps procedure is summarized in Figure 2.

4 Description of the Four Modules

4.1 Environmental Conditions Normalization - ECN

The first module aims at removing the effects of the environmental variables. For that purpose, one regression model has to be fitted for each of the p engine variables. As the relationship between environmental and engine variables is complex and definitively not linear, the environmental variables can be supplemented by some non-linear transformations of the latter, increasing the number of explanatory variables. Interactions (all the possible products between two environmental variables), squares, cubes and fourth powers of the non binary environmental variables are considered. The number q of predictors in the model

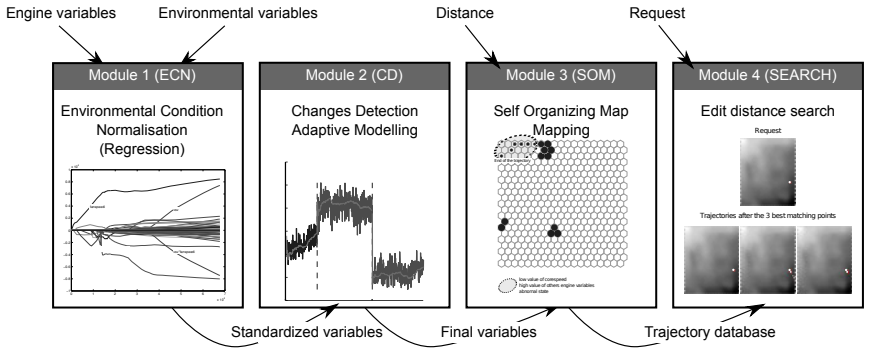


Fig. 2. Global architecture of the health monitoring tools

is therefore a priori equal to $(11+4)*(11+4-1)/2 = 105$ for the interactions variables and $11*4+4 = 48$ for the power of the continuous variables and the binary variables leading to a total of $q = 153$ predictors. This number is certainly too large and some of them are clearly irrelevant due to the systematic procedure used to build the non-linear transforms of environmental variables. A LASSO criterion [4] is therefore used to estimate the regression parameters and to select a subset of significant predictors. This criterion can be written using the notations from Section 2 for one engine variable Y^r, $r \in \{1, \ldots, p\}$ as :

$$\boldsymbol{\beta}^r = \arg\min_{\boldsymbol{\beta}^r \in \mathbb{R}^q} \sum_{i,j=1}^{I,n_i} \left(Y_{ij}^r - \sum_{l=1}^{q} \beta_l^r \, X_{ij}^l \right)^2 , \; \sum_{l=1}^{q} |\beta_l^r| < C^r \tag{2}$$

The regression coefficients are penalized by a condition which forces some of them to be null for a well chosen value of C^r. The LARS algorithm [4] is used to estimate all the solutions of the LASSO criterion (2) for all possible values of C^r. The optimal value of C^r with respect to the prediction error estimated by cross-validation (with 20 blocs) is finally selected. Another possibility could be to use BIC criterion instead of cross validation procedure to pick up the best model. The number of selected predictors and the coefficient of determination R^2 are listed in Table 2 for all engine variables. Engine variables are well explained by the proposed models as attested by the high value of the coefficients of determination.

A qualitative inspection of the model results was also carried out with the help of engine experts. The regularization path plot (as shown in Figure 3) is very

Table 2. Number of selected predictors and coefficients of determination for all engine variables

	corespeed	*fuelflow*	*ps3*	*t3*	*egt*
nb vars	25	43	31	30	41
R^2_{obs}	0.9875	0.9881	0.9773	0.9636	0.8755

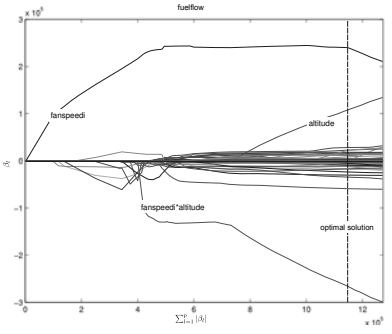


Fig. 3. Regularization path for the *fuelflow* variable: regression coefficients evolution with respect to C^r. The more significant explanatory variables are given and the best solution with respect to cross-validation is depicted by a vertical line.

interesting from the point of view of the experts, because it can be compared with their previous knowledge. Such a curve clearly highlights which are the more relevant predictors and they appear to be in very good adequateness with the physical knowledge on the system.

In summary, the first preprocessing module (ECN) provides $p = 5$ standardized engine variables denoted by $S^r = [S^r_{ij}, i \in \{1, \ldots, I\}, j \in \{1, \ldots, n_i\}]$, with $r \in \{1, \ldots, p\}$, which are the residuals of the selected regressions. They are independent of environmental conditions but still contain some significant aspects such as linear trends and abrupt changes at specific dates. We therefore propose to use an on-line Change Detection algorithm (CD) together with an adaptive linear model to fit the data.

4.2 Change Detection - CD

To take into account the two types of variation (linear trend and abrupt changes), we implement an algorithm based on the ideas from [5] and [7]. The solution is based on the joint use of an on-line change detection algorithm to detect abrupt changes and of a bank of recursive least squares (RLS) algorithms to estimate the slow variations of the signals. The algorithm works on-line in order to allows projecting new measurements on the map as soon as new data are available.The method can be described as follows:

1) One RLS algorithm is used for each one of the p standardized engine variables to recursively fit a linear model. For each $r \in \{1, \ldots, p\}$, for each engine $i \in \{1, \ldots, I\}$ and at each date l, one has to solve the following equation:

$$(\alpha^r_{il}, \beta^r_{il}) = \arg \min_{\alpha \in \mathbb{R}, \beta \in \mathbb{R}} \sum_{j=1}^{l} \lambda^{(l-i)} (S^r_{ij} - (\beta j + \alpha))^2, \tag{3}$$

where λ is a forgetting factor. The estimates α^r_{il} and β^r_{il} are respectively the intercept and the slope of the linear relationship. These estimates are then used

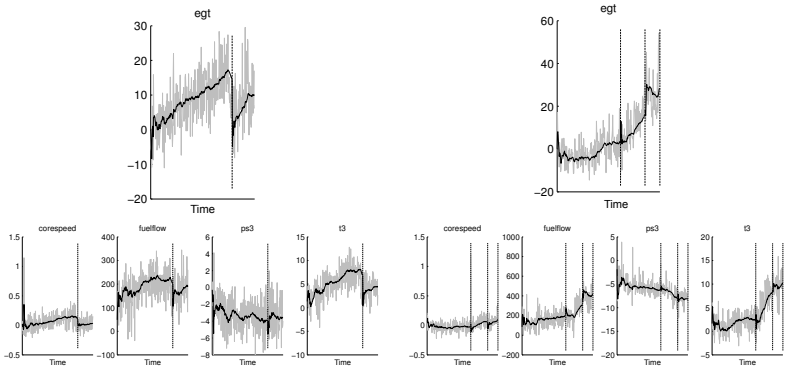


Fig. 4. Change detection results for two engines, (a) engine 2, (b) engine 41. Alarms are depicted by vertical lines, input signals are shown in light gray and signal estimates F^r using RLS are depicted by a black line. One figure (with respect to *egt*) is bigger than the others to present more clearly the RLS estimate of the signal.

to define the variables $\varepsilon_{il}^r = S_{il}^r - (\beta_{il}^r l + \alpha_{il}^r)$, which do not contain anymore the slow variations of the signals.

2) These values are concatenated in a vector $\boldsymbol{\varepsilon}_l = [\varepsilon_l^1, \ldots, \varepsilon_l^p]$, which is then used in a multi-dimensional Generalized Likelihood Ratio (GLR) algorithm [1] to detect the abrupt changes of the signals. The GLR algorithm is a sequential test procedure based on the following model:

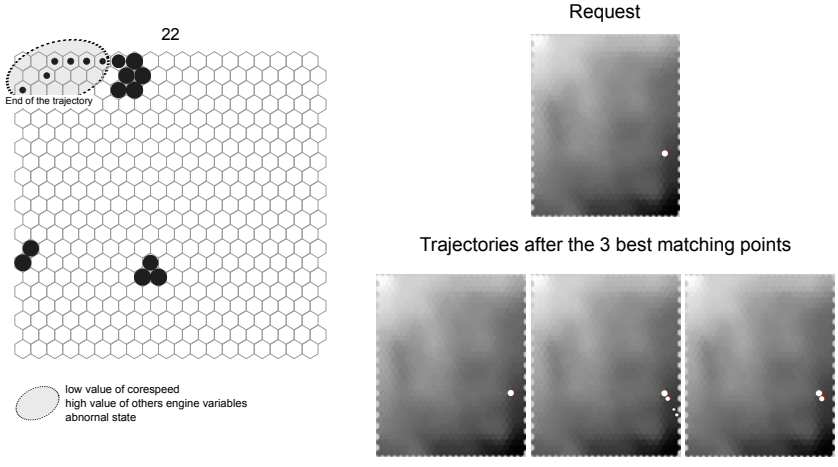
$$\boldsymbol{\varepsilon}_k \sim \mathcal{N}_p(\theta(k), \Sigma), \forall k > 0,$$

where $\mathcal{N}_p(\theta(k), \Sigma)$ is the multivariate normal distribution with variance Σ and mean $\theta(k) = \begin{cases} \theta \in \Theta_0 = \{||\theta|| < r_o\} & \text{if } k < t_0, \\ \theta \in \Theta_1 = \{||\theta|| > r_1\} & \text{if } k \geq t_0. \end{cases}$, ($r_0 < r_1$ are given constants).

3) Finally, when an alarm is sent by the GLR algorithm, all the RLS algorithms are re-initialized. The results supplied by this algorithm are the following:

- the alarm dates supplied by the multi-dimensional GLR algorithm;
- cleaned signals estimated by the RLS algorithm;
- slopes and intercepts estimated by the RLS algorithm.

Figure 4 presents the obtained results for two engines. One abrupt change was found for the first engine and 3 for the second one; all of them seem to be reasonable and a comparison between estimated alarm dates and recorded real events of the engine life have confirmed this fact. The estimated signals are also shown on these two figures. For more information on this aspect of the analysis process see [2]. From now, the observations corresponding to each flight are $F_{il} = [F_{il}^1, \ldots, F_{il}^p]$, where $F_{il}^r = \beta_{il}^r l + \alpha_{il}^r$ are the results of the transformations of the raw data performed by the first two modules (ECN and CD).



(a) (b)

Fig. 5. (a) Trajectories of engine 22 on the map. The sizes of the dots are proportional to the measurement date: smallest dots correspond to recent measurements, larger dots to older measurements. (b) Pieces of similar trajectories found using the edit distance (details are given in section 4.4)

4.3 Self-Organizing Maps - SOM

The cleaned signals F_{il} provided by the previous two modules are then used as input to a SOM for visualization purpose. To project the observations we use a $[20 \times 20]$ SOM implemented with the Matlab toolbox [9] and with defaults settings for the learning rate. Since the variables F_{il}^r are correlated, a Mahalanobis distance is used to whiten the data. A classical learning scheme is used to train the map. Figure 5 (a) presents one example of engine trajectories on the map, which clearly have different shapes. For the studied engine, available maintenance reports inform us that this engine suffers from an deterioration of its high pressure core. This fault is visible on the map at the end of the trajectory: the engine which was projected on the middle north of the map during a large part of its trajectory, suddenly moves towards the north-west corner of the map. This area of the map furthermore corresponds to abnormal values of the engine variables.

4.4 Similar Trajectory Matching - SEARCH

One of the final goal of the proposed tool concerns clustering and prediction of engine trajectories or pieces of engine trajectories. For this end, we have to define a proper distance between pieces of trajectories, which can be of different lengths. Before projection on the map, pieces of trajectories were sequences of $\mathbb{R}^p$-vectors, but as soon as measurements are projected, they can be described by sequences of integers corresponding to the units where measurements are projected. Such sequences will be denoted by $T = [k_1, \ldots, k_L]$ and as they take

their values in a finite set $\{1, \ldots, U\}$ (where U is the number of units), we will call them *strings*. The difficulty comes from the fact that the strings can have different lengths. Such a problem has been already investigated in other fields and one classical solution is to use Edit Distance, which is commonly used for approximate string matching [6].

To compare two strings, Edit Distance uses a *cost function*. This function gives individual cost for each unitary operation such as: suppression, addition or substitution. The cost of a sequence of operations $O = [o_1, o_2, \ldots]$ is simply equal to the sum of all the unitary costs, so: $cost(O) = \sum_t cost(o_t)$. Then, the Edit Distance between two strings $de(T, T')$ is defined as the minimal cost among all sequences of operations that fulfill the constraint $O(T) = T'$. Such a distance can be tuned to our end by carefully choosing unitary costs. We may in particular use the map topology to define meaningful substitution costs, by setting the cost of the substitution $k \leftrightarrow k'$ to the distance between unit k and unit k' on the map. With such a choice, we will take benefit of the fact that close units on the map can be exchanged with a small cost. Suppression and insertion costs are equal to the average of all the pairwise distances between units of the map.

With such a distance one can build classes of pieces of trajectories using hierarchical clustering. But this distance can also be used to supply clues on the possible future evolution of one trajectory. To perform such a task, the following method is proposed. Let T be a piece of engine trajectory:

1. compute the Edit Distance between T and all the pieces of engine trajectories recorded in the fleet database $[T_1, T_2, \ldots]$ (all these distances can be computed efficiently using dynamic programming [6]);
2. search for matching pieces T_x such that $de(T_x, T) < \eta$, where η is a given threshold;
   Note that these pieces are parts of already observed engine trajectories, which were recorded in the fleet database so that their evolutions after the matching points are therefore known, the third step uses this property.
3. look at the pieces that are just after the matching pieces. That gives an idea about the possible futures of T and enables the computation of probabilities of different types of maintenance events if the fleet database is connected to the maintenance database which recorded all the failures and maintenance operations performed on the fleet. We hope that it will be a useful tool to anticipate failures.

Figure 5 (b) presents preliminary results obtained using such an approach, T was built using the last 50 points of an engine trajectory. During this time period, this engine stays in the same unit. We show in Figure 5 (b) the pieces of trajectories that occurred after the 3 best matching points found in the fleet database using the proposed Edit Distance. These possible futures for T seem to be reasonable. Further works concern the connection with the maintenance database to perform a quantitative analysis of the results.

5 Conclusion

The method proposed in this paper is an nice tool to summarize and represent the temporal evolution of an aircraft engine health flight after flight. The regression approach used to deal with the problem of environmental condition normalization (ECN) seems to be effective, even if other model selection methods such as the BIC criterion could be investigated in further works to reduce the number of selected variables. The joint use of an adaptive algorithm to estimate signal evolution (RLS) and of a change points detection method (GLR) is also an interesting solution to deal with the non-stationary of the signals and to clean them (GLR module). Finally, Self-Organizing Maps (SOM) can be used to show the engine health evolution in a synthetic manner and to provide codes for synthetic representation of trajectories, that enables the development of predictive analysis tools (SEARCH module).

References

1. Basseville, M., Nikiforov, I.: Detection of Abrupt Changes: Theory and Application. Prentice-Hall, Englewood Cliffs (1993)
2. Côme, E., Cottrell, M., Verleysen, M., Lacaille, J.: Aircraft engine health monitoring using self-organizing maps. In: Springer (ed.) Proceedings of the Industrial Conference on Data-Mining (2010)
3. Cottrell, M., Gaubert, P., Eloy, C., François, D., Hallaux, G., Lacaille, J., Verleysen, M.: Fault prediction in aircraft engines using self-organizing maps. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 37–44. Springer, Heidelberg (2009)
4. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.J.: Least angle regression. Annals of Statistics 32(2), 407–499 (2004)
5. Gustafsson, F.: Adaptive filtering and change detection. John Wiley & Sons, Chichester (2000)
6. Navarro, G.: A guided tour to approximate string matching. ACM Computing Surveys 33, 2001 (1999)
7. Ross, G., Tasoulis, D., Adams, N.: Online annotation and prediction for regime switching data streams. In: Proceedings of ACM Symposium on Applied Computing, pp. 1501–1505 (March 2009)
8. Svensson, M., Byttner, S., Rognvaldsson, T.: Self-organizing maps for automatic fault detection in a vehicle cooling system. In: 4th International IEEE Conference on Intelligent Systems, vol. 3, pp. 8–12 (2008)
9. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: Som toolbox for matlab 5. Tech. Rep. A57, Helsinki University of Technology (April 2000)

Classification Using Topologically Preserving Spherical Self-Organizing Maps

Heizo Tokutaka[1], Masaaki Ohkita[1], Ying Hai[1],
Kikuo Fujimura[2], and Matashige Oyabu[3]

[1] SOM Japan Inc.
[2] Tottori University
[3] Kanazawa Institute of Technology
{Heizo Tokutaka,tokutaka}@somj.com

Abstract. A new classification method is proposed with which a multidimensional data set was visualized. The phase distance on the spherical surface for the labeled data was computed and a dendrogram constructed using this distance. Then, the data can be easily classified. To this end, the color-coded clusters on the spherical surface were represented based on the distance between each node and the labels on the sphere. Thus, each cluster can have a separate color. This method can be applied to a variety of data. As a first-example, we considered the iris benchmark data set. A boundary between the clusters was clearly visualizible with this coloring method. As a second example, the velocity (first derivative) mode of a Plethysmogram pulse-wave data set was analyzed using the distance measure on the spherical surface.

Keywords: Spherical Surface SOM, Colored Clustering, Distance Measurement, Boundary decision.

1 Introduction

There is a clear benefit in utilizing spherical Self-Organizing Maps for classifying multidimensional data [1], [2] and [3]. The discontinuation at the 4 borders and 4 corners of a 2D planar map affect the results obtained after the learning process [4]. In the spherical Self-Organizing Maps, these discontinuations don't happen. The case where the phase relationship between the data points is most clear and precise is on a spherical surface. In the spherical Self-Organizing Maps this relationship can be exploited for constructing the cluster, and next the dendrogram. To delineate the clusters, the boundary needs to be improved by a manual operation. It is important to draw the boundary on the map, precisely and correctly. This is necessary in order to realize a maximally correct classification at a later stage, when the clusters are assigned to classes, using class labels. The method of learning vector quantization (LVQ) [4] is proposed as a way of determining the cluster boundary automatically. With this method, it is necessary to choose a learning parameter.

A new classification method was proposed with which the multidimensional data can be visualized [1] and [2]. There, a phase distance on the spherical surface for the

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 308–317, 2011.

label data was computed and the dendrogram was created by the range distance calculation among the labels. Here, a color classification of groups of nodes forming a data cluster on the spherical surface, using labeled data points, was carried out by considering the distance between the unlabeled node and the labeled data point. The method can be applied to a variety of data. In this paper, the first example considered is the iris benchmark data set [5] and [6]. In a second example, the distances among the labels were used to classify Plethysmogram pulse-wave data.

2 Algorithm

Previously in [1] and [2], we have used the Iris benchmark as an example for classification. It consists of data from 50 of vergicolor (abbreviation ver), 50 of verginica (abbreviation gnc), 50 of setosa (abbreviation set), together forming the iris data [5] and [6]. At this time, a spherical surface was transformed up to the Griff value 1 (ref. [1] and [2]) emphasizing the U-matrix [7] which shows the boundaries between the clusters, where by the Griff value 0 a spherical surface is given, however, in the Griff value 1, the position of the maximum distance (the darkest part) is kept the radius 1 and the minimum (the brightest part) is 0, as shown in Fig. 1(a) (the Griff operation is detailed in ref. [1] and [2]). After transforming the sphere by the Griff value 1, a dendrogram was drawn using the group average-method like Fig. 1. In Fig. 1, the dendrogram near ver_23 and the spherical surface, with the Griff value 0, are shown in (a) and (b), respectively. The dendrogram is drawn mainly near ver_23 of the red character, but the boundary, decided from the dendrogram, is a dotted line, and the actual one should be a solid line.

The following strategy was proposed in order to decide on a correct boundary automatically. Using labeled data, it was examined which node of the group of ver, gnc or

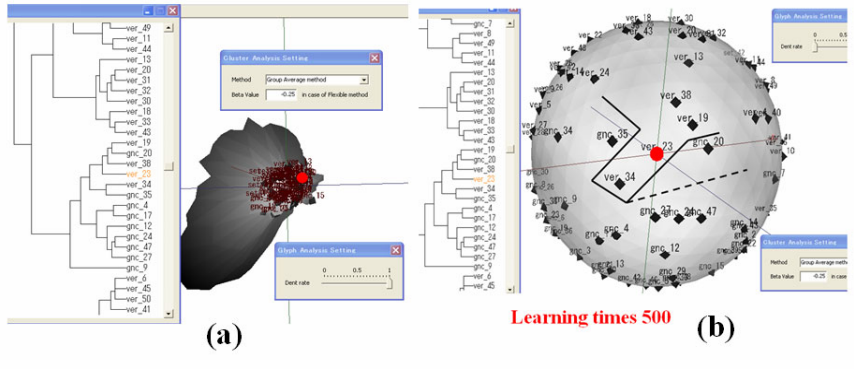


Fig. 1. (a) After a spherical surface was transformed to the Griff value 1, as explained in Fig. 3 of ref. [2], the dendrogram on the left was constructed by the group average-method. (b) When the spherical surface was returned to the full sphere using the Griff value 0, the solid line is the boundary that should be pulled out of the label, but the boundary became as shown with the dotted line, corresponding to the dendrogram on the right (from Fig. 6 of the ref. [2]).

set was near to each one of the 642 nodes that compose the spherical surface in Fig. 1. The same procedure was used for constructing the dendrogram of Fig. 1. In other words, a spherical surface was transformed based on the Griff value 1. In the procedure, the distances between the unlabelled nodes and all the labeled ones were calculated and then, the node was assigned to the group of the label that was the nearest. Thus, all 642 nodes were divided into three groups that are either ver, gnc or set. The 3 data points, from a total of 9, were taken out from the ver, gnc and set data. Then, 51 data of 42 nodes of 42 face pieces on the sphere and the 9 iris data were used. Then, a polyhedron was transformed using the same procedure as in Fig. 1, i.e. with the Griff value 1. A dendrogram was constructed by the group average-method and shown in Fig. 2. The nodes that are in the neighborhood to the ver, gnc or set belong to these groups respectively. There are some nodes that are far from the labels. These are for example, nodes 31, 36, 38. There, when the dendrogram was traced, two gnc and three ver belong to these nodes 31, 36 and 38. In the same way, 35, 40, 20, 22, 23, 24, 41, 25, belong to set when the same dendrogram is traced. Also, all three data points of gnc, and all three of ver are connected through the root of the dendrogram. Here, we examined how the nine labels can be assigned to each of the 42 nodes by searching the nearest label data of each node. The method is named as All label algorithm. As shown in Fig. 2, by the method, the former nodes belonged to set, ver, set, respectively. Then, the latter nodes belonged to set, ver, set, gnc, set, set, ver, gnc, respectively. Thus, the nodes of 31 and 38 belong to set. They are different from those by the dendrogram. Also, the nodes of 40, 22, 41, and 25 belong to ver, gnc, ver and gnc, respectively. They are different from set group which would be expected from the dendrogram. The results are indicated in the bottom of Fig.2, either ver, gnc, or set.

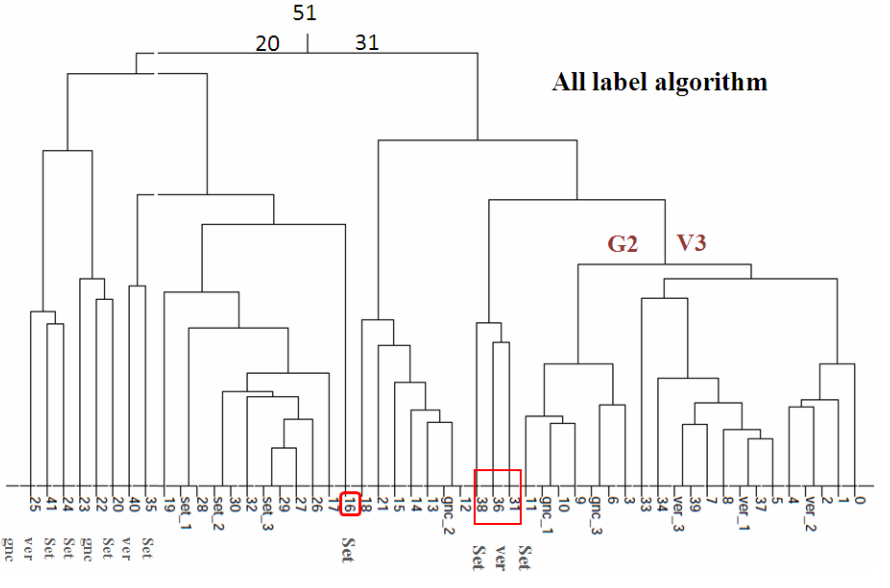


Fig. 2. The obtained dendrogram is shown with 42 nodes and 9 labels, where the nodes far from the labeled data are assigned by the All label algorithm and the group is shown in the bottom. Unlabelled nodes in the bottom are of course classified with the nearest label. For example, node 0 is classified as ver and the 19 is set and so on.

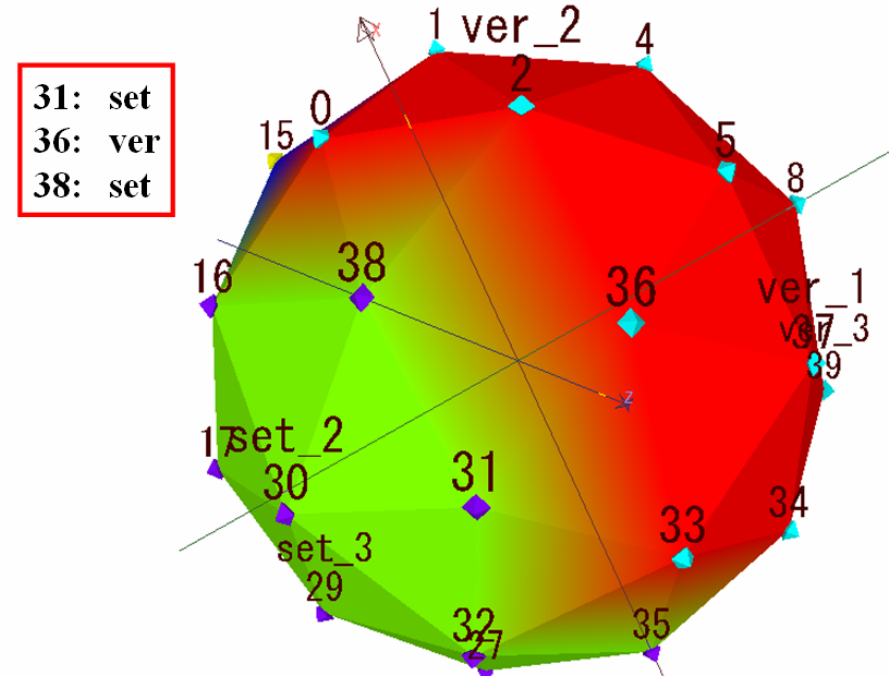


Fig. 3. A total 51 of iris 9 data and 42 nodes were pasted onto a 42 face map

As shown in the bottom of the dendrogram in Fig. 2, the nodes seem to be separately identified in the dendrogram by this All label algorithm. The label group of the identified node could be pasted onto 42 face map. It was possible to color like in Fig. 3 and then, a smooth boundary was obtained as shown. Because of this, the number of the nodes was increased to 642. The algorithm was verified on the data of the iris 150 data points.

The node groups (Nos. 31, 36, 38, and 35, 40, 20, 22, 23, 24, 41, 25 on the dendrogram), which were on the far left from labels in the dendrogram, as shown in Fig. 2, are judged to be close to a variety of labels. The reason is described below. First, a group is made with the group average-method among the nearest ones. The result is due to what makes the group bigger than another. All these nodes are near the boundaries. If theses are examined, they are judged different from the dendrogram, as they are near a variety of labels as shown in the bottom of Fig. 2.

3 Application to Iris Benchmark Data Set

The above-mentioned node coloring algorithm was applied to the iris benchmark data set [5] and [6]. This is the cluster sorting problem using all 150 data points taking each 50 data points from the virginica, setosa and vergicolor categories, respectively. In this problem, the boundary between setosa and the other two categories is very clear. However, it is an extremely difficult problem to find a boundary between

vergicolor and virginica. The result after 500 learning epocks for 642 nodes is shown in Fig. 4(a). In this example, the boundary between set(setosa) and ver(vergicolor) is very smooth as shown in Fig. 4(a). However, in this benchmark problem, it is difficult to distinguish between ver(vergicolor)_19 and gnc(virginica)_20. It is also difficult to distinguish ver_23 and ver_24 from gnc. As shown in Fig. 4(b), the boundary between ver_19 and gnc_20 is clearly separated. The boundary between two ver labels of ver_23 and ver_24, and gnc groups are clearly indicated by color in Fig. 4(b). It is a mathematically difficult problem to draw the boundary. When looking at the result of adding colors on the spherical surface, it will be possible to draw a more smooth boundary such as (a) between the set and the ver group, if ver_23 and ver_24 are read as belonging to the gnc group, thus different from the ver group. In Fig. 5, an unknown sample UK_3 was projected on the spherical surface. From Fig. 5(a) showing a usual SOM analysis, it can be understood that UK_3 is riding on the U-matrix, [7]. Under the present condition, it can't be decided whether UK_3 belongs to set or ver. However, Fig. 5(b), which is colored by our technique, shows that UK_3 is near set_44. Thus, it is in the group of ver based on the boundary between the colors.

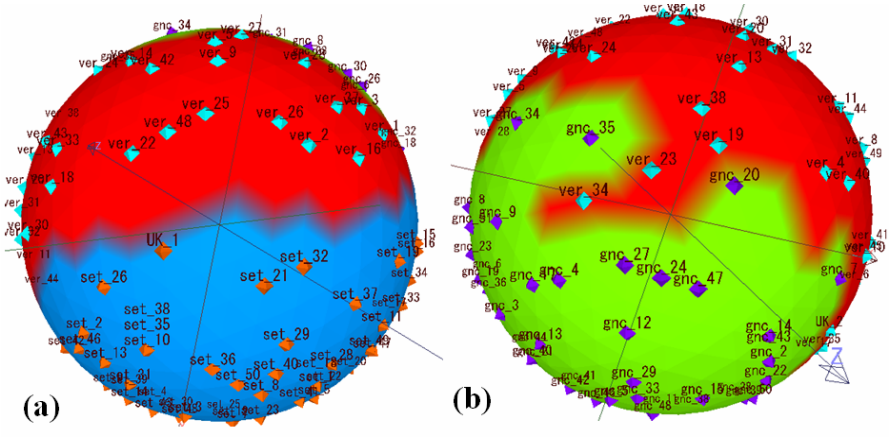


Fig. 4. (a) When the iris data is used for training, during 500 epochs, a spherical 642 node map, a smooth boundary can be drawn between set(setosa) and ver(vergicolor). (b) For the ver and gnc groups, as shown in (b), it was possible to draw the boundary between ver_19 and gnc_20 which are very close to each other. ver_23 and ver_24 are projected into the gnc group like a peninsular. There, a clear boundary was found as shown in (b).

Incidentally, in Fig. 6(a), SOM learning is performed during 50 epochs for a 642 spherical surface node map. In the iris benchmark problem, the boundary of ver and gnc is always arguable. Therefore, the set was excluded and only ver and gnc were used for the experiment. The samples of ver and gnc were projected on the spherical surface. Then, the boundary was drawn on the spherical surface [8]. At the same time, the boundary was also colored**,** as shown in Fig. 6. The boundary agrees almost with the coloring boundary. It should be understood that the coloring boundary is finer than the line boundary.

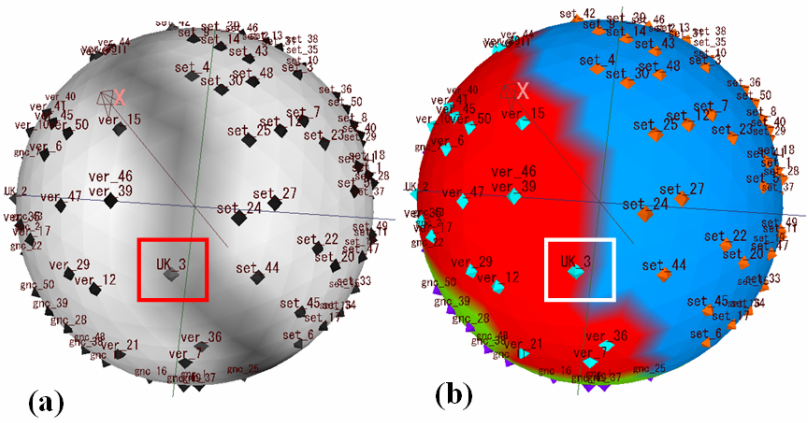


Fig. 5. (a) An unlabeled data point UK_3 (surrounded in the red square in (a) and in the white square in (b) was projected on the spherical surface. In ordinary SOM learning, UK_3 is on the line of the U-matrix [7] as shown in (a). There, it can't be found whether UK_3 belongs to ver or set. However, it can be found that UK_3 belongs clearly to ver from (b).

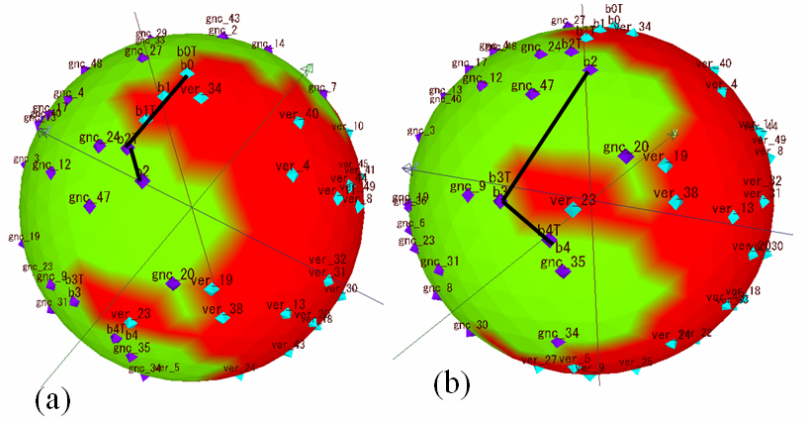


Fig. 6. The boundary between ver and gnc groups is shown by the line from b0T to b4. (a) The part of the line boundary from b0T to b2 . (b) The line boundary from b2 to b4 (ref. [8]).

4 Range Distance Calculation

In a spherical surface SOM, the phase can be displayed topologically more directly than with a planar SOM. The velocity pulse wave, which was obtained by the differentiation of the volume (original) plethysmogram pulse-wave [9], was used for the analysis. When the velocity pulse wave is differentiated, the wave is called the acceleration plethysmogram [9]. More than one corrugated velocity pulse waves are shown in Fig. 7(a). This bundle of waves is arranged for 1 period and they are divided into 100 divisions. However, the position where the height value of each corrugated wave first became 0 was cut down to see a pulse wave change. The cut down wave was also divided into 100 divisions. Then, the rate at which the magnitude became 0 within 1 period was added as another dimension.

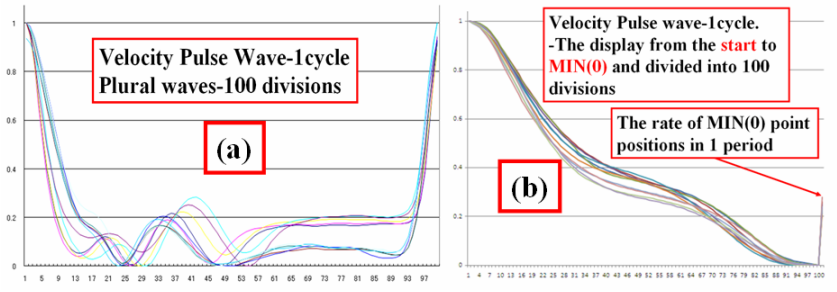


Fig. 7. (a) The bundle of corrugated velocity pulse waves in 1 period. (b) The position where each corrugated wave height value of the figure (a) became 0 first was cut out. Also, it was divided in 100 steps. Then, the rate data where the wave height became 0 in 1 period was added.

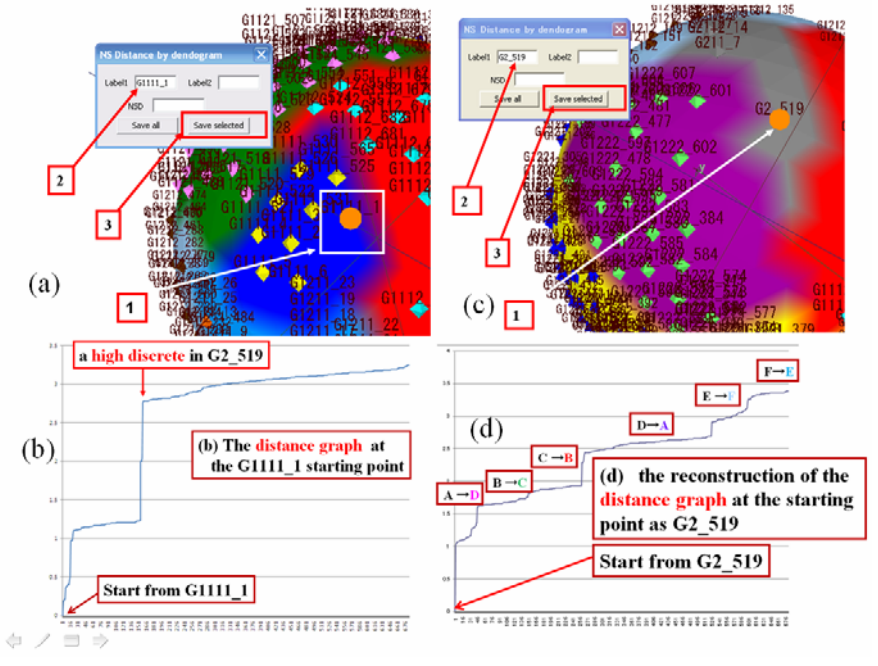


Fig. 8. (a) The distance among all labels from label G1111_1 of the first data in the dendrogram (No.1 enclosed with the square box) using the *blossom* tool [10] was measured and it is shown in (b). (b)The distances were arranged in ascending order. In (b), there is a big discontinuous point at label G2_519. (c) This time, all the distances which started from label G2_519 were measured and (d) they were shown in ascending order. As shown in (d), the distances are distributed with equal intervals. Then, the labels were put into all intervals. For (d), since it is obvious that there are much more people in the health region, the labels were then relabeled as indicated by the (arrowed) right hand-side labels.

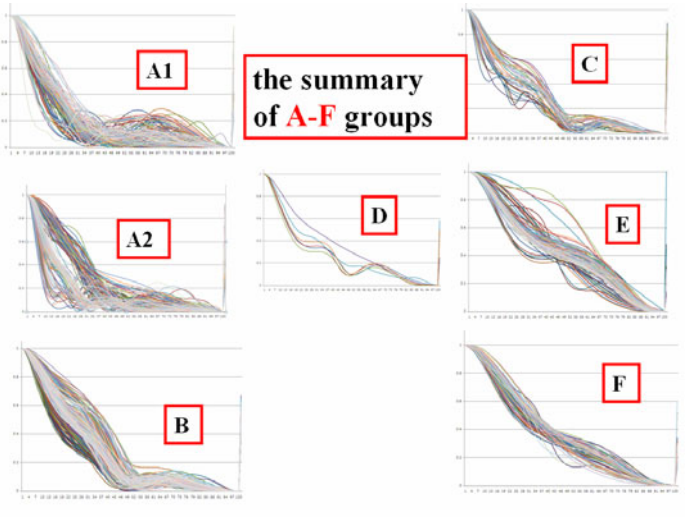


Fig. 9. The relabeled 6 groups**,** named A to F in Fig. 8(d)**,** are shown

The 683 data of 101 dimensions that was prepared were used for learning a spherical surface SOM. The data waves were classified into six groups from A to F, as shown in Fig. 9. The procedure is shown in Fig. 8. The classification from the A to F group is already carried out in (a) and (c) of Fig. 8. There, the distance measurement was accomplished by the procedure which was shown in [1] and [2]. The procedure supported by the tool is described in Fig. 8(a). The first label G1111_1 of the dendrogram that is searched in begins as No.1 (enclosed with the square box). Then, the label is put in Label1 as No.2 (enclosed with the square box). Then, "Save selected" in the Fig. 8(a) was chosen as No.3 (enclosed with the square box). All the distances among G1111_1, itself and the other 683 label data in total were measured and saved as a csv file. When the obtained distances were arranged in ascending order, a large discontinuous jump happens in G2_519 position (Fig. 8(b)). Therefore, the distances from the (G2_519) point to all other labels were recalculated and the procedure which is similar to Fig. 8(a) is shown in (c). The obtained distances can be arranged once again in ascending order and as a result Fig. 8(d) is obtained. A large discontinuous jump like Fig. 8(b) doesn't occur as shown in Fig. 8(d). Approximately all equal discontinuous steps are obtained and shown in Fig. 8(d). It is divided into 6 groups named as A-F starting from the bottom. However, the group with the largest number of people was considered temporarily as the waves coming from the health people. D was read as A. The A label was read in descending order starting from that point. The obtained corrugated wave group is shown in Fig. 9. Since the A group grew many corrugated waves, the display was divided into A1 and A2. As for the first A group, the waves are falling steeply. We also observe that the corrugated waves are falling more gently for the F group.

5 The Conclusions

The coloring of the nodes comprising a spherical surface was performed by measuring the distance between the node and all labels on the transformed polyhedron. The result was used as a way to decide on the boundary when considering the case of

cluster classification. The method is regarded as All label algorithm. The size of the node, which composes a spherical surface, in order to verify this algorithm, was reduced to 42. In this way, the boundary based on the coloring was confirmed. Next, the number of nodes was increased to 642. Our technique to decide on the boundary between the cluster groups, using the coloring of each node, was applied to and analyzed for the case of the iris benchmark problem [5] and [6]. For this problem, the number of epochs for learning the spherical maps was increased from the usual 50 to 500 epochs. In the past, the boundary between ver_19 and gnc_20 was not clear. By the present method, however, the boundary became clear. Then, there is a problem with the lower precision used in the past for this benchmark problem. For example, ver_23 and ver_24, overhang like a peninsular in Fig. 4(b) compared with the smooth boundary between set and ver of Fig. 4(a). If ver can be read as gnc, it should be possible to draw a smooth boundary like Fig. 4(a). Also, there is an unknown sample UK_3 which is near set_44 of setosa (set) in Fig. 5(a). It was found that this UK_3 belonged to the ver group by using the node coloring method.

With the usual planar SOM, the segregation of the clusters is in principle possible. However, when considering the distances between the labels, a sphere was transformed into a polyhedron. For this case, the distance calculation among the labels was performed and the result was displayed (see Fig. 8). We observed a distance discontinuity among the groups like Fig. 8(d). Large bundles of corrugated wave, which don't have any labels, were classified successfully. By the node coloring method, the cluster decision boundary became visible and was correctly estimated, as discussed above. Detailed information on the cluster classification was obtained by the distance measurement applied to the transformed polyhedron. Thus, a more quantitative evaluation became possible, as was demonstrated on the cluster groups. As other demonstration, the chain-link benchmark problem [11] of the three-dimensional data was also examined in the appendix. This is a very suitable problem for the *blossom* tool [10] of the three-dimensional visualization. Finally, we thank Prof. M. V. Hulle of K. U. Leuven for kind reading and correcting the manuscript and Prof. T. Kohonen of Academy of Finland for kind reading and giving useful comments to the manuscript.

References

1. Tokutaka, H., Fujimura, K., Ohkita, M.: Cluster Analysis using Spherical SOM (in Japanese). Journal of Biomedical Fuzzy Systems Association 8(1), 29–39 (2006)
2. Tokutaka, H., Fujimura, K., Ohkita, M.: Cluster Analysis using Spherical SOM. In: WSOM 2007, Bielefeld Germany, September 3-6 (2007)
3. Nakatsuka, D., Oyabu, M.: Application of Spherical SOM in Clustering. In: Proceedings of Workshop on Self-Organizing Maps (WSOM 2003), pp. 203–207 (2003)
4. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer, Heidelberg (2005)
5. Fisher, R.A.: The Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics 7, 179–188 (1936)
6. http://www.ics.uci.edu/~mlearn/databases/
7. Ultsch, A., Guimaraes, G., Korus, D., Li, H.: Knowledge Extraction from Artificial Neural Networks and Applications. In: Proc. TAT/WTC 1993, pp. 194–203. Springer, Heidelberg (1993)
8. Matsuda, M., Tokutaka, H.: Decision of class borders on a SOM (in Japanese). Journal of Biomedical Fuzzy Systems Association 10(2), 27–38 (2008)

9. Tokutaka, H., Maniwa, Y., Gonda, E., Yamamoto, M., Kakihara, T., Kurata, M., Fujimura, K., Shigang, L., Ohkita, M.: Construction of a general physical condition judgment system using acceleration plethysmogram pulse-wave analysis. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 307–315. Springer, Heidelberg (2009)
10. http://www.somj.com
11. 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)

Appendix

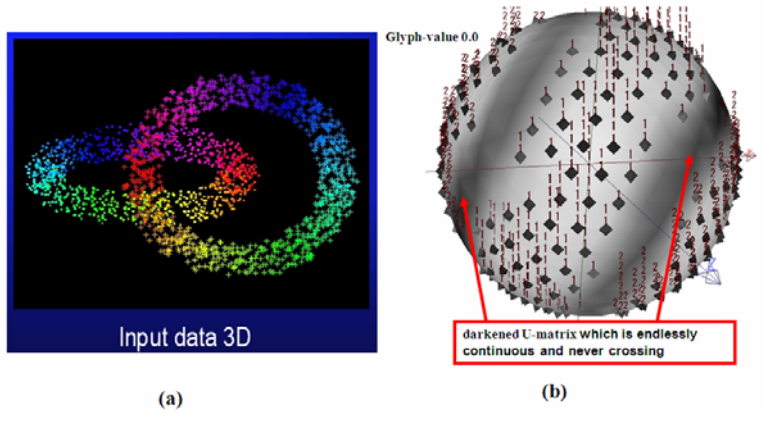


Fig. 10. (a) chain-link problem [11] where two links 1 and 2 are crossing perpendicularly in 3 dimensional spaces. (b) The input data 3D of (a) are learned by the spherical SOM of *blossom* [10]. The boundary between 1 and 2 are indicated by darkened U-matrix which is endlessly continuous and never crossing.

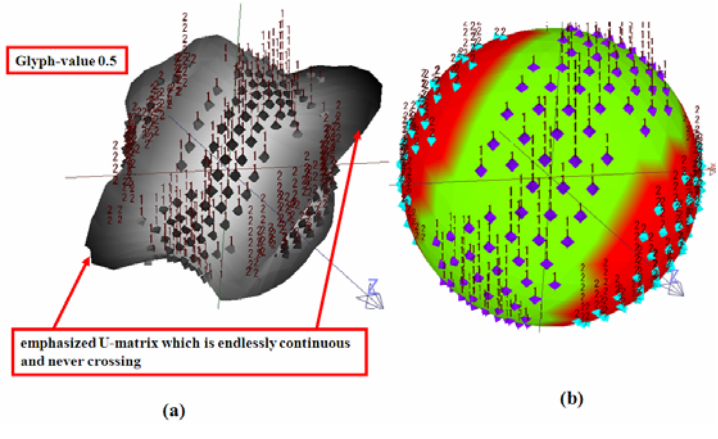


Fig. 11. (a) The Griff-value is increased from 0 (Fig.10(b)) to this 0.5, where U-matrix are emphasized and they are endlessly continuous and never crossing. (b) Fig.10(b) is colored by the All label algorithm. Then, it is easy to find the boundary of this two cluster problem. Fig.10(b) and Fig.11(a),(b) given by the cluster SSOM are displayed all in the same position.

Visualizing Patterns in the Air Quality in Mexico City with Self-Organizing Maps

Antonio Neme[1,2] and Leticia Hernández[2]

[1] Adaptive Informatics Research Centre. Aalto University, Konemiehentie 2 Espoo, FIN
aneme@cis.hut.fi
[2] Complex systems group, Autonomous University of Mexico City San Lorenzo 290, Mexico City, MEX

Abstract. Air pollution in big cities is a major health problem. Pollutants in the air may have severe consequences in humans, creating conditions for several illness and also affect tissues and organs, and also affect other animals and crop productivity. From several years now, the air quality has been monitored by stations distributed over major cities, and the concentration of several pollutants is measured. From these data sets, and applying the data visualization capabilities of the self-organized map, we analyzed the air quality in Mexico City. We were able to detect some hidden patterns regarding the pollutant concentration, as well as to study the evolution of air quality from 2003 to 2010.

1 Introduction

Pollutants are any substances that affect the normal cycle of any vital process or degrade infrastructure [1]. The sources of pollutants are several and well identified. Pollutants may be originated from human actions, but also from natural events. Among the former it can be listed the incomplete combustion of organic combustibles in cars, the end products from industrial reactors, and dust and minerals from construction sites [2].

Air pollution affects several regions all over the planet, and mainly impact major cities, in which has been reported to be a major problem of health for the last 30 years. Several respiratory illness have been reported to be a consequence of high levels of pollutants [3]. In Mexico City, during the years 2000 and 2008, more than 100,000 deaths were caused directly or indirectly from bad air conditions, and almost one million visits to the hospital were attributed to pollutants [4,5]. It is expected that if the tendency continues, more than four millions of related illness will be reported by 2020 [6] Also, air pollutants impact directly in other animals and in green areas and in harvest productivity [1].

Among the most dangerous pollutants is carbon monoxide (CO), that affects blood oxygenation as it reacts with hemoglobin and may lead to severe health problems and in many cases to death. Nitrogen oxides (NO) and dioxide (NO_2) are also dangerous air pollutants, as they decrease lung function and increase the risk of acute bronchitis. Ozone (O_3) is also a pollutant relevant for health issues,

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 318–327, 2011.

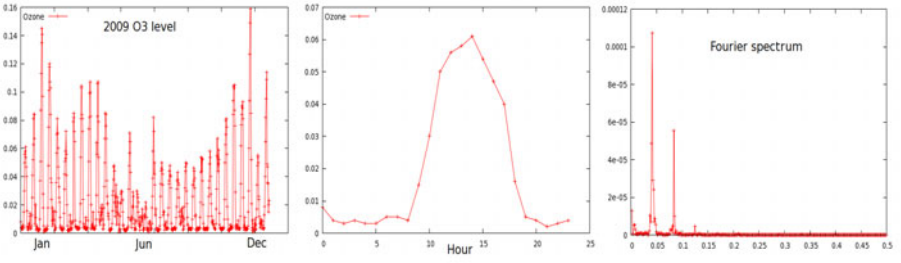


Fig. 1. Time series for O_3. It is shown the concentration for one station in Mexico City for year 2009, as well as the concentration for one day. Also, the Fourier spectrum is presented. Two frequencies are visible: a 24-hour period and a weekly period.

as it takes part in reactions that lead to the so-called smog. Sulfide dioxide (SO_2) is also constantly monitored as it is a precursor of acid rain which affects crops and some edifications. Lead (Pb) may impact nervous connections and cause blood and brain disorders [1]. Finally, particulate matter (PM), also called fine particles, lower than 10 micrometers are also considered as pollutant as they affect the lungs. In general, PM are identified as $PM_{2.5}$ for radius lower than 2.5 micrometers and in PM_{10} for radius less than 10 micrometers [7].

Air quality is affected by several variables, among the obvious the presence of pollutants, but also the wind and atmospheric conditions as well as traffic conditions. In particular, the case of the urban area of Mexico City and its surroundings is a complicated one, as the specific conditions of altitude, wind patterns, high population density, and traffic issues seem to be particularly adverse for air quality [7].

Pollutants concentration levels may vary in daily basis, but also may be subject of other influences. Fig. 1 shows the O_3 concentration for one day, one month and one year, as well as its Fourier spectrum. Trying to find patterns in just one time series may be achievable, but trying to make sense from several time series, as well as to try to identify patterns and correlations among them may require special tools.

As a part of a research group focused in studying the air quality in Mexico City, we have analyzed multidimensional data consisting of air pollutants measured every hour in several monitoring stations distributed over the most polluted areas in Mexico City. From these measures, we have been working in finding patterns in the data, and in this contribution, we describe some of them. In section 2 we briefly describe the self-organized map and its capabilities for data analysis. Then, in section 3 we present the application of SOM for visualization of air pollutants, and in section 4 we present some conclusions.

2 Data Visualization

We define the air quality vector for a given time as the average concentration of NO_2, NO, O_3, SO_2, CO, Pb, $PM25$, and $PM10$ and the number of times each of these pollutants exceeded the norm. Thus, the space has 16 dimensions

as there are eight pollutants and two statistics are considered for each one of them. Italic font refers to the average pollutant concentration (NO_2), whereas the pollutant with a bar over it refers to the number of times it exceeded the safe level over the specified period ($\overline{NO_2}$). Monitoring and visualization of all air quality data is not a straightforward task. Several pollutants have to be analyzed simultaneously and correlations and general patterns are to be found in data. The chemistry of air in polluted environments, although well studied, is still under development [2]. Thus, for environmental and health officials to make sense of data, automatic tools for data analysis and multidimensional data visualization are required.

As a support tool for data visualization, we applied the self-organizing map (SOM), as a consequence of its outstanding capabilities for high-dimensional data visualization, which are well above those presented by other techniques [8,9]. The SOM preserves neighborhood relationships during training through the learning equation (1), which establishes the effect that each best matching unit (BMU) has over any other neuron.

The SOM structure consists of a two-dimensional lattice of units referred to as the map space. Each unit n maintains a dynamic weight vector w_n which is the basic structure for the algorithm to lead to map formation. The dimension of the input space is considered in the SOM by allowing weight vectors to have as many components as features in the input space. Variables defining the input and weight spaces are continuous. Weight vectors are adapted accordingly to:

$$w_n(t+1) = w_n(t) + \alpha(t)h_n(g,t)(x_i - w_n(t)) \tag{1}$$

where $\alpha(t)$ is the learning rate at epoch t, $h_n(g,t)$ is the neighborhood function from BMU g to unit n at epoch t and x_i is the input vector. SOM has been widely applied as a data visualization tool as a consequence of its capabilities in computing high-order statistics [9], which is translated in a two-dimensional map that is a good approximation of the distribution observed in the high-dimensional space.

SOMs have been applied in the air quality domain as for example, [12] in which authors classify monitoring stations accordingly to pollutant levels. Here, we are interested in study the evolution of pollutant concentration in Mexico City, considering several levels of resolution.

3 Results

Since 1986, a governmental law established an agency focused on air quality monitoring. From that agency, several monitoring stations measure the concentration of air pollutants [7]. The full list of contaminants that are now monitored was started in 2003. Prior to that year, only a subset of pollutants was measured and not all pollutants were monitored in all stations. That is the reason we started our analysis only with data from 2003. Every year is represented by a 16-dimensional vector, as there are 8 major pollutants, and for each one of them, the average yearly concentration as well as the number of measures that the safe

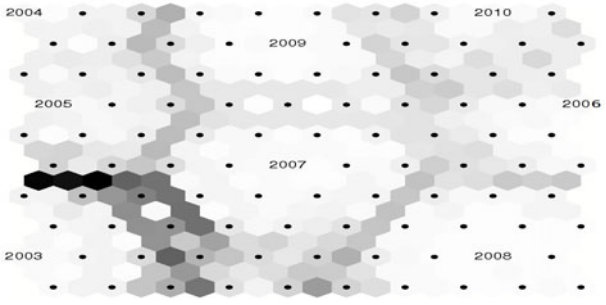


Fig. 2. U-matrix for air pollution map for years 2003 - 2010 in Mexico City. Each year is represented by a 16-dimensional vector, as there are eight pollutants. For every pollutant, its average for the whole year and all stations was considered, and also the number of times the measure exceeded the safe level.

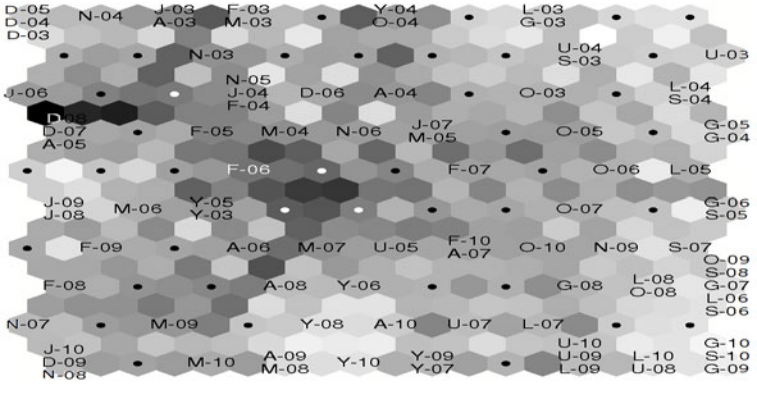


Fig. 3. Air pollution map for all months from January 2003 to November 2010. Each month is again represented by a 16-dimensional vector, and no geographic information in included. Codes are: J-January, F-February, M-March, A-April, Y-May, U-June, L-July, G-August, S-September, O-October, N-November, D-December, plus the year in two digit format.

level threshold was exceeded are considered. All variables are normalized and all are supposed to have the same relevance, so no additional preprocessing was considered. Fig. 2 shows the U-matrix [11] for the eight years analyzed. It is observed that year 2003 is in a well-defined cluster, and it makes sense, as that year was particularly polluted [7].

We start with a coarse-grain analysis, in which each year is represented by a single vector with 16 components, considering the average over all stations and months. Then, we increase the detail and each month is now defined by a vector, so we have $12 \times 8 - 1$ vectors (December 2010 is not considered). At the same time, we consider the case in which data is differentiated by monitoring station. Each one of the 10 stations is represented by the average pollutant concentration over

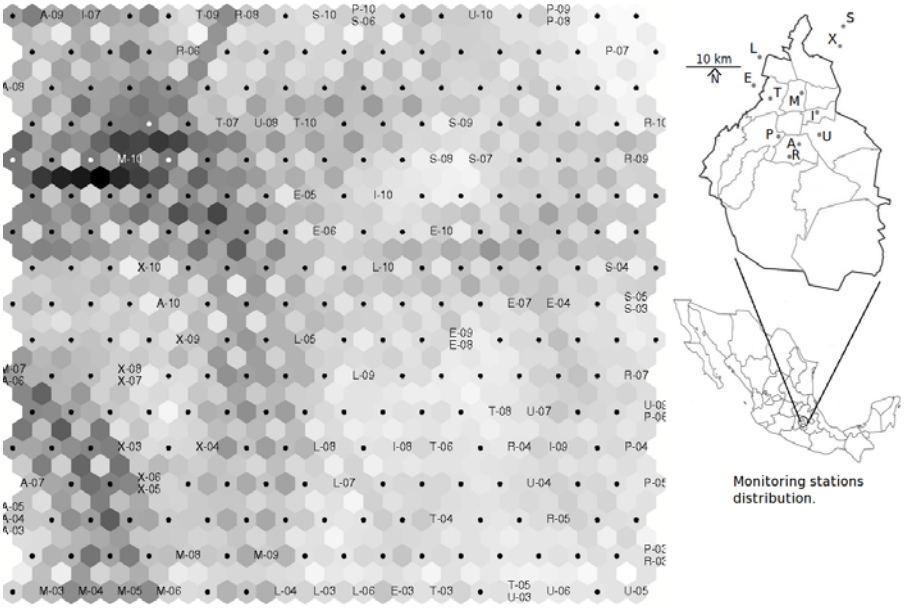


Fig. 4. Air pollution map for all stations for years 2003 to 2010. Each pair station-year is a 16-dimensional vector.

each year, so this time we have 10×8 vectors. At the lowest level, each hour of every day is represented by an 8-dimensional vector, in which each component is the concentration of each one of the pollutants measured at that hour, on average over all available monitoring stations. In all cases, maps were generated with the SOM PAK, available at http://www.cis.hut.fi/research/som_lvq_pak.shtml.

In fig. 3 it is presented the map for all months since January, 2003. It is observed some seasonality, as some months tends to be in the same cluster, as for example, December tends to be clustered at the upper left corner (D-03, D-04, D-05). Those three years presented bad air quality conditions in general, and the weather and traffic conditions of that month are proper for high concentration of pollutants. The cluster at the upper left corner contains the months with the highest concentrations and highest measures exceeding the norm. It is observed that other months in 2003 were also mapped to that area. Seasonality is also observed in other months, such as May (Y-06, Y-07, Y-08, Y-09, Y-10). The months with better air quality tend to be those in the late spring and in summer. Those months are mapped to the bottom right corner, and it is observed that June (U), July (L), August (G), and September are mainly located there.

In fig. 4 it is presented the u-matrix for the map for air pollution from year 2003 to 2010, but now, vectors are composed by measures for the specified monitoring station and years. In this scheme, we have intrinsic geographic information, which may be helpful to seek for patterns in data. For these experiment, only 11 stations were considered, as the rest of them (ten more) does not measure all considered pollutants. So we have 11×8 vectors. In this map, the vectors

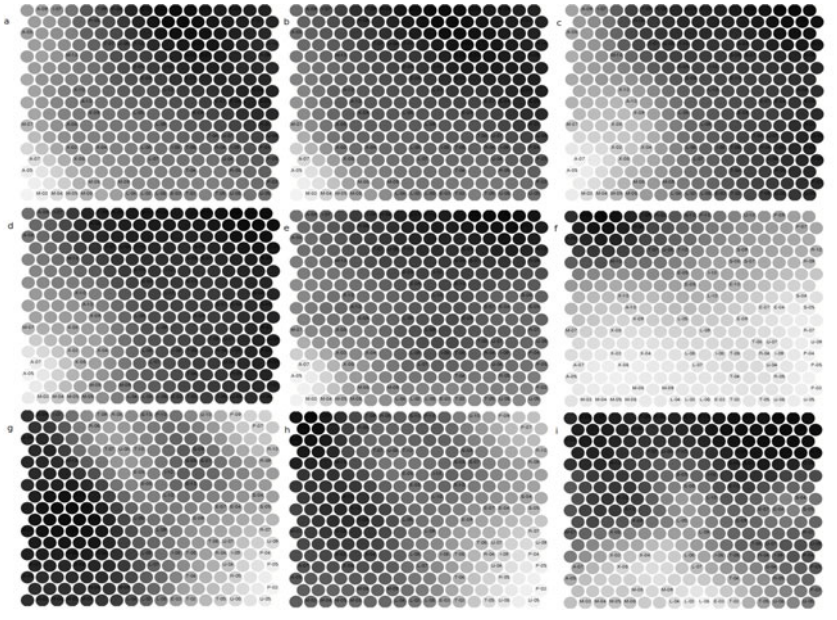


Fig. 5. Planes for NO_2 (a), $\overline{NO_2}$ (b), NO (c), $\overline{NO}$ (d), CO (e), $\overline{CO}$ (f), $O3$ (g), $\overline{O3}$ (h), and SO_2 (i) for the map shown in fig. 4. Gray level indicates the corresponding value of the plane. Light tones indicate higher values.

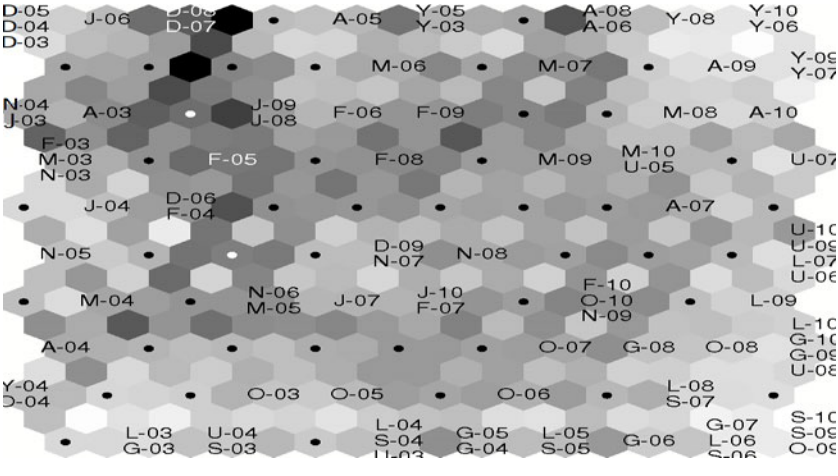


Fig. 6. Air pollution and weather conditions map for all months from January 2003 to November 2010. Each month is represented by a 19-dimensional vector, as the previously cited 16 variables (pollutants) are included, plus air temperature, wind speed, and wind direction. Codes are: J-January, F-February, M-March, A-April, Y-May, U-June, L-July, G-August, S-September, O-October, N-November, D-December.

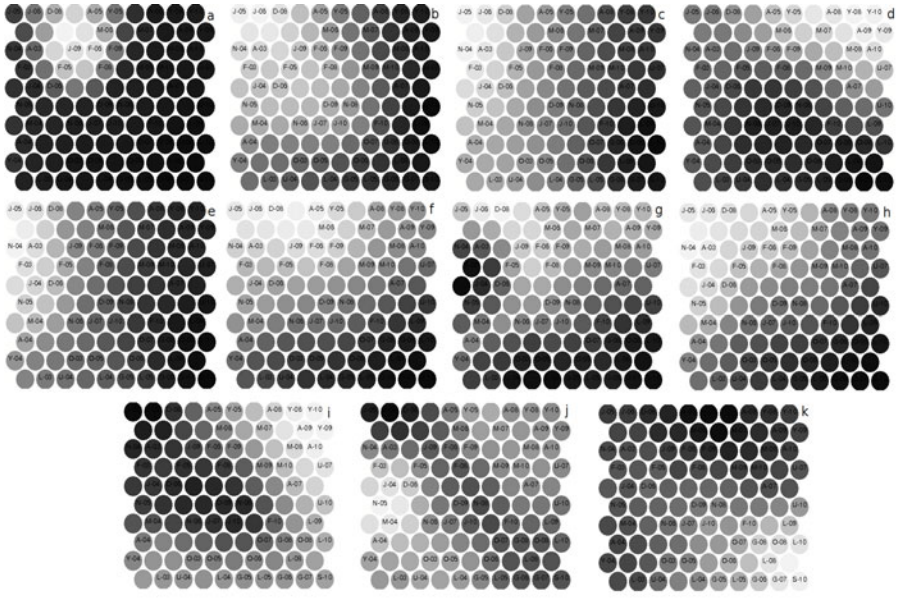


Fig. 7. Planes for NO_2 (a), NO (b), CO (c), $O3$ (d), SO_2 (e), PM_{10} (f), $PM_{2.5}$ (g), Pb (h), temperature (i), wind speed (j), and wind direction (k) for the map shown in fig. 6. Gray level indicates the corresponding value of the variable. Light tones indicate higher values.

corresponding to the poorest air quality are located in the cluster at the bottom left corner, and the gray levels surrounding that cluster show a heavy border. Stations A and M recorded the highest pollutant concentration during 2003 and 2004. Station A is situated in a residential area, with several avenues and streets, with a heavy traffic flow, whereas station M is situated in downtown, in an area with one of the highest population density in the city.

Interestingly, neither of the two stations A and M is located near to an industrial area, but stations L and E are. As SO_2 has as one of its sources some industrial processes, it is not a surprise that the concentration of that pollutant is very high in stations L and E, mainly for years 2003 and 2004. In 2002, a major modification in the environmental law urged industries to incorporate new air quality controls. As it was not an immediate process, 2003 and 2004 were still very high in some pollutant concentrations, as that of SO_2.

Fig. 5 show some of the variables (average pollutant concentration and number of measures higher to the safe level). It is observed that light areas tend to be located at the lower left corner, with the exception of O_3 and $\overline{O_3}$.

In fig. 6 we consider an additional variable in data. Besides the air pollutants, we included the wind conditions (speed and direction) as well as temperature. Now, each input vector has 19 components, as it contains the 16 mentioned variables plus the average air temperature, the average wind speed and the wind direction. As expected, the map is slightly different from map 3, as now some

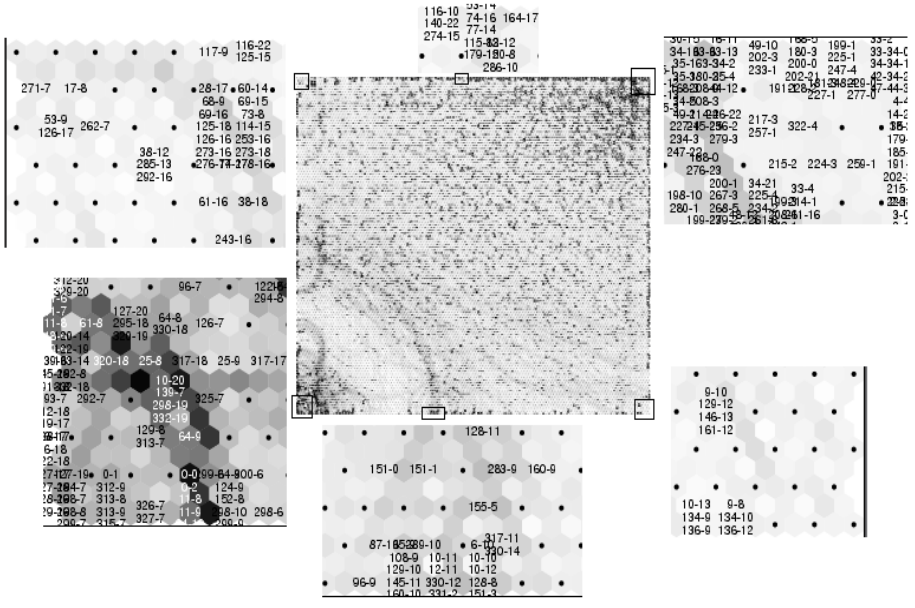


Fig. 8. U-matrix for the SOM for all hours in 2010 (December not included). Each hour of the year was represented by an 8−dimensional vector, with each component associated to the pollutant concentration measured during that hour, on average over all available monitoring stations. The label indicates the day number, starting with Jan, 1st, as day 0, and the hour of the day. Hours are numbered from 0 to 23.

weather conditions are considered. However, from the planes in fig. 6, some interesting patterns have been identified.

Temperature has been associated to pollutant levels [2]. In one hand, high temperatures favors the presence of O_3. This seems to be detected by the SOM, as shown in plane i in fig. 6 and plane d. Wind speed is also important for air quality, as it may disperse pollutants. Those months in which air speed is high tend to present lower concentrations of NO_2 and $PM_{2.5}$

At the highest level of detail, input vectors are related to pollutant measures for each hour. Under this scheme, we have at most 24×365 vectors, each with eight components, one for each pollutant concentration for that hour. Each component is then the average measure over all available stations during that hour. Fig. 8 shows the U-matrix for the SOM of of all hours for 2010, except for those of December. Fig. 9 is the U-matrix for all hours in 2009, but it is only shown the month for hours in July (7) and in December (11). Those two months were selected as, at least in 2009, they were very different regarding air quality. Hours with the highest concentrations are clustered in the lower left corner, and the U-matrix shows a heavy border. It is observed that several measures from December are placed there, whereas none of the measures of July are clustered in that zone.

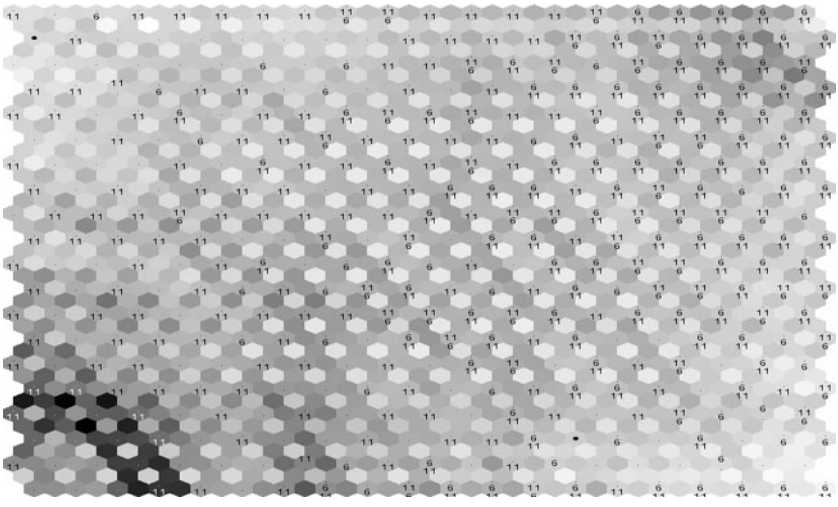


Fig. 9. U-matrix for SOM for all hours in 2009. Only labels for July (6) and December (11) are shown. In the bottom left corner, the hours with the highest pollutant measures are clustered and in fact, correspond the so called thermic inversion. It is observed that only measures of December are there, as the cold temperatures present in that month in Mexico City tend to favor pollutant concentrations. In contrast, almost all measures in July tend to be very low, and many of them are clustered at the upper right corner.

With the SOM capabilities for data visualization, it is possible to identify some patterns otherwise obscure.

4 Discussion and Conclusions

Air pollution is a multifactorial phenomenon, not entirely understood so far. Withe the help of visualization tools, such as the SOM, some hidden patterns may become evident. SOM is a very efficient tool for visualization of multidimensional data, as it is able to identify high-order statistical relations in data.

Here, we have presented the study of air pollutants in Mexico City through self-organizing maps. Due to the visualizing capabilities of SOM, we have been able to visually correlate some environmental variables, such as pollutant levels and weather conditions. Also, we identified the overall evolution patterns for air pollutants from years 2003 to 2010.

Acknowledgments

This research is derived from a project supported by Instituto de Ciencia y Tecnología del Distrito Federal (ICyTDF), under contract PICCT08-55.

References

1. Sportisse, B.: Fundamentals in air pollution. Springer, Heidelberg (2010)
2. Seinfeld, J., Pandis, S.: Atmospheric Chemistry and Physics: From Air Pollution to Climate Change, 2nd edn. Wiler (2006)
3. Knox, E.: Atmospheric pollutants and mortalities in English local authority areas. J Epidemiol Community Health 62, 442–447 (2008)
4. Ferrer-Carbonell, J., Escalante-Semerena, R.: Contaminación atmosférica y efectos sobre la salud en la Zona Metropolitana del Valle de México. Revista Economía Informa 360, 119–143 (2008)
5. http://www.bvsde.paho.org/bvsacd/eco/038267/038267-04.pdf
6. Bell, M., Davis, D., Guoveia, N., Borja, V., Cifuentes, L.: The avoidable health effects of air pollution in three Latin American cities: Santiago, São Paulo and Mexico City. Environmental Research 100, 431–440 (2006)
7. http://www.sma.df.gob.mx/simat2/
8. Kohonen, T.: Self-Organizing maps, 3rd edn. Springer, Heidelberg (2000)
9. Hujun, Y.: The self-organizing maps: Background, theories, extensions and applications. In: Computational Intelligence: A Compendium, pp. 715–762 (2008)
10. Kaski, S., Kohonen, T.: Exploratory data analysis by the self-organizing map: structures of welfare and poverty in the world. In: Apostolos-Paul, N.R. (ed.) Neural Nwteorks in Financial Engineering, pp. 498–507 (1996)
11. Ultsch, A.: Self organizied feature maps for monitoring and knowledge aquisition of a chemical process. In: Proc. of the Int. Conf. on Artificial Neural Networks, pp. 864–867 (1993)
12. Alvarez-Guerra, E.: A SOM-based methodology for classifying air quality monitoring stations (2010), doi:10.1002/ep.10474

Decision of Class Borders on a Spherical SOM with Non-equal Class Distributions

Nobuo Matsuda[1] and Heizo Tokutaka[2]

[1] Oshima College of Maritime Technology, Electronic and Mechanical Engineering,
1091-1 Komatsu, Suo-oshima-cho, Oshima-gun, Yamaguchi-ken, 742-2193, Japan
matsuda@oshima-k.ac.jp
[2] SOM JAPAN Co. Ltd.
680-0941 Tottori, Japan
tokutaka@somj.com

Abstract. We propose an approach to the determination of class borders on a SOM with non-equal class distributions. Our approach treats the class distribution as a variance-covariance matrix. The class distribution is expressed by a variance-covariance matrix and its decision border between the classes is determined from input data by using the eigenvalues and the corresponding eigenvectors of the matrices. Using the iris dataset of Fisher, it is shown that our approach allows the effect of non-equal class distributions on the decision borders to be successfully visualized in a qualitative and comprehensible manner.

Keywords: Decision border, Self-Organizing Map, Non-equal class distributions, Variance-covariance matrix, Visualization.

1 Introduction

The Self-Organizing Map (SOM) is a powerful tool for exploring huge amounts of multi-dimensional data. The SOM by Kohonen [1] is a kind of neural network algorithm that projects high dimensional data onto a low dimensional space. In the traditional SOM algorithm, however, the "border effect" problem have been pointed out, and several spherical SOMs based on a geodesicdome [2] or a toroidal SOM have been proposed as a remedy. To show its potential effectiveness, the spherical SOM has been applied to clustering. For instance, Tokutaka *et al.* [3] proposed a highly accurate cluster analysis using the spherical SOM.

On the other hand, there is a proposal [4] for the interpretation of information on the map. However, the U-matrix [5] has been mainly used in the traditional SOM and the spherical SOM. In the U-matrix, the Euclidean distance between nodes is expressed by a gray level. Therefore, it is difficult to decipher the information of the class distributions or the borders when the shading due to the U-matrix changes continuously. Tokutaka *et al.* [3] converted the shade of the U-matrix to the distance and obtained a dendrogram to perform a classification based on distances. They recommended analyzing the dendrogram and the graphical object on the polygon surface interactively to eliminate misclassifications. In the discussion of their cluster analysis, they used boundaries that were artificially drawn. When discussing the dendrogram,

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 328–337, 2011.

however, it is crucial that the class boundary is decided accurately because the precision of the computed class boundaries controls the accuracy of the cluster analysis. Generally, it is difficult to depict the class borders of multi-dimensional data, however the projection capability of the SOM of multi-dimensional data will allow the class borders to be successfully visualized. Therefore, there exists a necessity for a method accurately drawing the decision borders on the SOM.

To achieve the above goal, we have proposed two methods for determining the class decision borders [6]. The proposed methods mainly covered the case of equal class distributions, but it is vital that the class boundary is decided as accurately as possible. Therefore, we especially propose an approach to deal with the case of non-equal class distributions in this paper.

2 Methods and Procedures for Decision Borders on the SOM

2.1 Decision Border with Equal Class Distributions

Now, assume that there are two data points on the SOM. One of them is located at the center part of the class distribution, and the other is located near the class boundary. The latter will have an overwhelmingly larger influence than the former on the decision of the class border. Therefore, it is reasonable to approximate the class borders from the input data close to the boundary on the SOM [6].

Let us select a pair of feature vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$ on the SOM as shown in Fig.1. They are of a different class (class A and class B) and are placed at the nearest boundary. Symbols of open diamond stand for class A and those of open square stand for class B, and points of open circle should be selected on the border. The feature vectors are assumed to be m-dimensional and vector $\boldsymbol{X} = (x_1, x_2, \dots, x_m)$ belongs to class A and vector $\boldsymbol{Y} = (y_1, y_2, \dots, y_m)$ belongs to class B.

Let us consider the case of determining point $\boldsymbol{D} = (d_1, d_2, \dots, d_m)$ of an arbitrary feature vector on the boundary from the feature vectors *X* and *Y*.

Assume that the two class distributions are almost equal, the point $\boldsymbol{D}$ on the boundary should be selected to satisfy the condition that the summation *S* of k and l is the minimal value, where k is the square value of the distance between $\boldsymbol{D}$ and $\boldsymbol{X}$, and l is the square of the distance between $\boldsymbol{D}$ and $\boldsymbol{Y}$.

$$\begin{aligned} S = k + l = &(x_1 - d_1)^2 + (x_2 - d_2)^2 + \cdots + (x_m - d_m)^2 + \\ &(y_1 - d_1)^2 + (y_2 - d_2)^2 + \cdots + (y_m - d_m)^2 \end{aligned} \tag{1}$$

In order for *S* to be a minimal value, it is necessary to satisfy the simultaneous equations obtained by differentiating equation (1) with respect to $d_1, d_2, \dots, d_m$. Therefore, we get the solution:

$$2(x_i - d_i) + 2(y_i - d_i) = 0 \quad (i = 1, \cdots, m) \tag{2}$$

Finally, each components of point *D* is calculated as:

$$d_i = \frac{(x_i + y_i)}{2} \qquad (i = 1, \cdots, m) \tag{3}$$

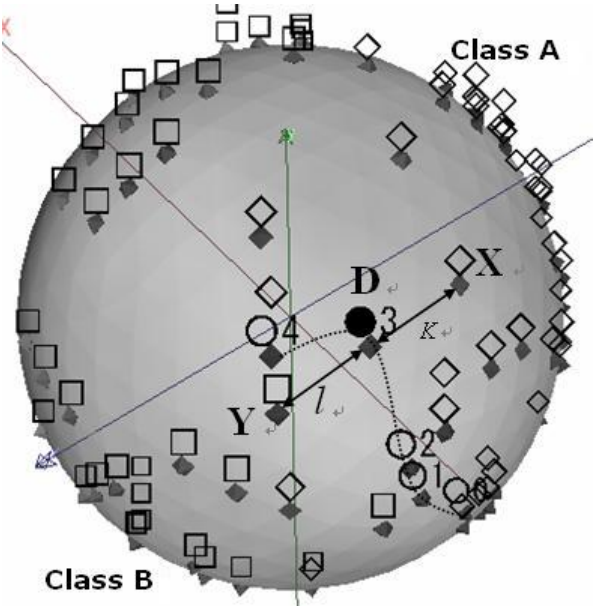


Fig. 1. Decision border and feature vectors X and Y on the spherical SOM: Open diamond nodes are class A and open square nodes are class B. Open circle nodes are the calculated ones on a border and a close circle node is the point D. The dotted line is a part of the borderline.

2.2 Decision Border with Non-equal Class Distributions

When a class distribution is considered, we assume that the class distribution can be expressed by using a variance-covariance matrix. When the two variance-covariance matrices **U** and **V** of the class A and class B, respectively, are defined as equation (4), then equation (1) is enhanced to equation (5) as follows.

$$\mathbf{U}=\begin{bmatrix} U_1^2 & U_{1,2} & \cdots & \cdots & U_{1,n} \\ U_{2,1} & U_2^2 & \cdots & \cdots & U_{2,n} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & U_{n-1}^2 & \vdots \\ U_{n,1} & U_{n,2} & \cdots & U_{n,n-1} & U_n^2 \end{bmatrix}, \mathbf{V}=\begin{bmatrix} V_1^2 & V_{1,2} & \cdots & \cdots & V_{1,n} \\ V_{2,1} & V_2^2 & \cdots & \cdots & V_{2,n} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & V_{n-1}^2 & \vdots \\ V_{n,1} & V_{n,2} & \cdots & V_{n,n-1} & V_n^2 \end{bmatrix} \tag{4}$$

$$M_1^2=[x_1-d_1,\cdots,x_m-d_m]\begin{bmatrix} U_1^2 & U_{1,2} & \cdots & \cdots & U_{1,n} \\ U_{2,1} & U_2^2 & \cdots & \cdots & U_{2,n} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & U_{n-1}^2 & \vdots \\ U_{n,1} & U_{n,2} & \cdots & U_{n,n-1} & U_n^2 \end{bmatrix}^{-1}[x_1-d_1,\cdots,x_m-d_m]^T$$

$$M_2^2 = [y_1 - d_1, \cdots, y_m - d_m] \begin{bmatrix} V_1^2 & V_{1,2} & \cdots & \cdots & V_{1,n} \\ V_{2,1} & V_2^2 & \cdots & \cdots & V_{2,n} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & V_{n-1}^2 & \vdots \\ V_{n,1} & V_{n,2} & \cdots & V_{n,n-1} & V_n^2 \end{bmatrix}^{-1} [y_1 - d_1, \cdots, y_m - d_m]^T \tag{5}$$

The point ***D*** on the boundary should be selected to satisfy the condition that the summation T of M_1^2 and M_2^2 is the minimal value, with M_1^2 is the square value of the distance between ***D*** and ***X***, and M_2^2 is similarly the square of the distance between ***D*** and ***Y***.

$$T = M_1^2 + M_2^2 = (\mathbf{X} - \mathbf{D})\mathbf{U}^{-1}(\mathbf{X} - \mathbf{D})^T + (\mathbf{Y} - \mathbf{D})\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{D})^T \tag{6}$$

In order for T to be a minimal value, it is necessary to satisfy the simultaneous equations obtained by differentiating equation (6) with respect to d_1, d_2, … , d_m. However directly differentiating equation (6) with respect to d_i complicates the handling of the expression. Thus the calculation is done after $\mathbf{U}^{-1}$ and $\mathbf{V}^{-1}$ in equation (6) are diagonalized by using the eigenvalues and the corresponding eigenvectors.

$$T = (\mathbf{X} - \mathbf{D})\mathbf{P}^{-1}\mathbf{A}\mathbf{P}(\mathbf{X} - \mathbf{D})^T + (\mathbf{Y} - \mathbf{D})\mathbf{Q}^{-1}\mathbf{B}\mathbf{Q}(\mathbf{Y} - \mathbf{D})^T \tag{7}$$

Here $\mathbf{A}$ and $\mathbf{B}$ are matrices whose elements on the diagonal are composed of the eigenvalues of $\mathbf{U}^{-1}$ and $\mathbf{V}^{-1}$, respectively, and $\mathbf{P}$ and $\mathbf{Q}$ are matrices whose elements are composed of the eigenvectors of $\mathbf{U}^{-1}$ and $\mathbf{V}^{-1}$ respectively:

$$\mathbf{A} = \begin{bmatrix} \lambda_1^A & 0 & \cdots & 0 \\ 0 & \lambda_2^A & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_n^A \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \lambda_1^B & 0 & \cdots & 0 \\ 0 & \lambda_2^A & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_n^B \end{bmatrix} \tag{8}$$

Differentiating T in equation (7) with respect to d_i', we conclusively get:

$$d_i' = \frac{(\lambda_i^A x_i' + \lambda_i^B y_i')}{\lambda_i^A + \lambda_i^B} \qquad (i = 1, \cdots, m) \tag{9}$$

Here d_i', x_i' and y_i' are defined respectively by the following equations:

$$\begin{bmatrix} x_1' & - & d_1' \\ \vdots & \vdots & \vdots \\ x_n' & - & d_n' \end{bmatrix} = \mathbf{P}^{-1} \begin{bmatrix} x_1 & - & d_i \\ \vdots & \vdots & \vdots \\ x_n & - & d_n \end{bmatrix} \tag{10}$$

$$\begin{bmatrix} y_1' & - & d_1' \\ \vdots & \vdots & \vdots \\ y_n' & - & d_n' \end{bmatrix} = \mathbf{Q}^{-1} \begin{bmatrix} y_1 & - & d_i \\ \vdots & \vdots & \vdots \\ y_n & - & d_n \end{bmatrix} \tag{11}$$

We can see that the equation coincides with equation (3), when two eigenvalues are equal in equation (9).

2.3 Procedure for Determining Decision Border

When the probabilistic relation between a feature vector and the class distribution is uncertain, it is difficult to obtain the decision borders. Then it is necessary to ensure that the map contains the information of the distribution of the feature vectors. The SOM is the easiest method to obtain such a map.

After obtaining the spherical SOM as shown in Fig.1, the following three steps are repeated for determining the decision borders with the equal class distributions.

Step 1
A number of candidates on the spherical SOM are selected from the data sets near the boundary.
Step 2
The distances between candidates are calculated and a pair of data points with the minimum distance is determined.
Step 3
Point D on the boundary is selected from the pair by equation (3).

After some points on the boundary are calculated by repeating step 1 through 3, a borderline is drawn. In Step 1, candidates in the class A and class B are selected according to equation (12) among the datasets.

$$\exp(-(\mathbf{x}-\boldsymbol{\mu}_A)^2 / R) \prec \varepsilon, \exp(-(\mathbf{y}-\boldsymbol{\mu}_B)^2 / R) \prec \varepsilon. \tag{12}$$

where $\boldsymbol{\mu}_A$ and $\boldsymbol{\mu}_B$ stand for vectorial reference points of each class, and R stands for a parameter (The value within the range from 0.01 to 0.1 is usually used as a value of R for the retrieval.). The vectorial reference points are chosen from the node located at the center part of the classes. ε is a threshold value. In Step 1 candidates for the decision borders are usually selected among the boundary dataset. If necessary, nodes of SOM or other dataset can be chosen. They are selected on the basis of the distance between candidates. Meanwhile, when the decision borders are determined with the non-equal the class distributions, then it is necessary to calculate the variance-covariance matrices of each class, their eigenvalues and the corresponding eigenvectors before beginning the step (1) of the procedure for the decision borders. In the step (3) point D on the boundary should be selected from the pair by using equation (9) instead of equation (3).

3 Experiments and Results

3.1 Dataset for Analysis

The benchmark dataset for analysis is described: the iris dataset of Fisher [7] is a well known benchmark data and consists of three classes (setosa, virginica,

vergicolor).The number of objects in each class is 50. The data is 4-dimensional and the attributes are the sepal length and width and the petal length and width. The label of each class data on the SOM or in the figures is shown with a brief symbol. Setosa, vergicolor and virginica classes are represented respectively by set_n, ver_n, and gnc_n: for example, the first data of virginica is described as "gnc_1".

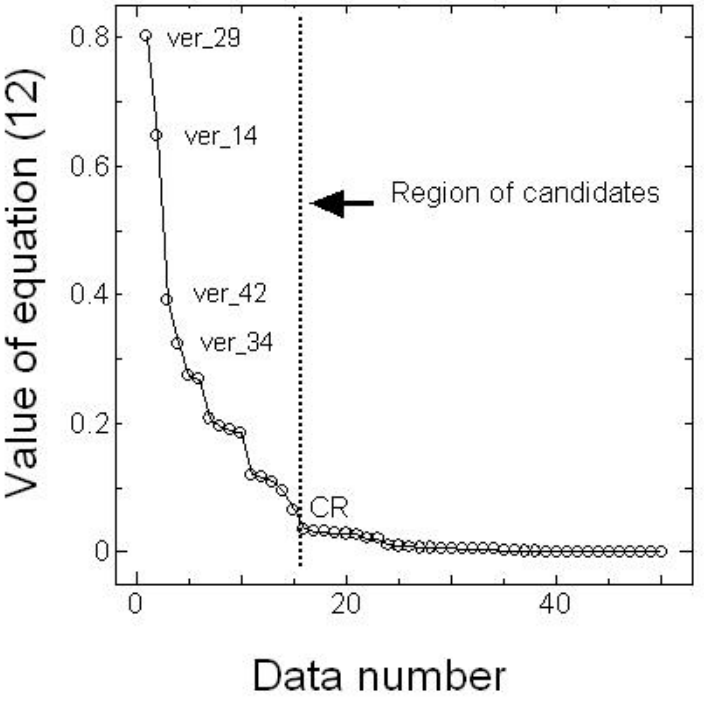


Fig. 2. Selection of candidates ($R = 0.01$, $\boldsymbol{\mu}_A = (0.4820, 0.3311, 0.6063, 0.5829)$, $\varepsilon = 0.0092$)

3.2 Result with Equal Class Distributions

We used the software of blossom [8] as the spherical SOM analysis tool to analyze the datasets. For the iris data, the points from b0 to b4 on the spherical SOM were calculated by repeating steps 1 through 3 for the combination of two versicolor datasets (ver_23, 34) and five virginica datasets (gnc_9, 17, 24, 27, 35). The calculated points form the borderline near the four data sets, versicolor19, 23, 38 and virginica20, were misclassified in the cluster analysis using the spherical SOM [3]. Fig.2 shows how the candidates were selected based on equation (12) when node b0 on the boundary was determined.

We adopted the candidates among the nodes that were contained within the left side region from the point CR. In this figure the slope of the value calculated by equation (12) changed significantly at this point CR. Fig.3 shows the results of the projection of the calculated values onto the spherical SOM. In this figure, gnc_20 was

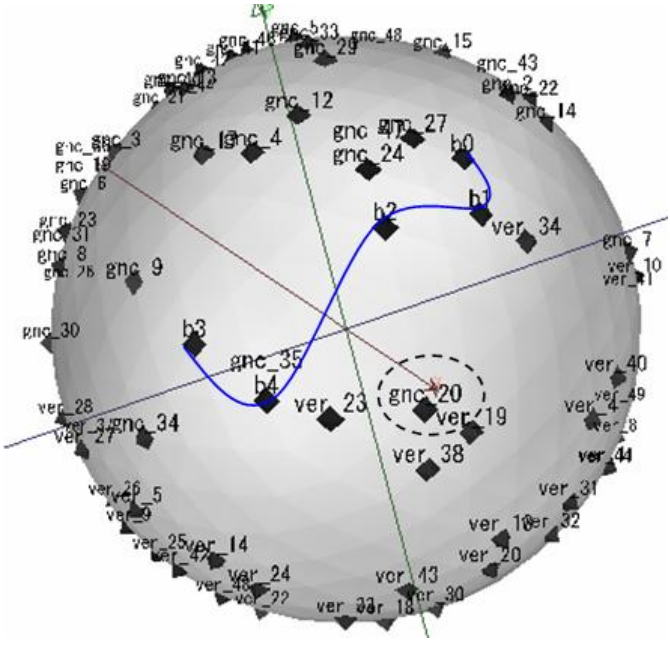


Fig. 3. A borderline pictured in the vicinity of gnc_20. A short dashed oval line is the line to facilitate the finding out gnc_20 and not the borderline.

misclassified as a vergicolor class by the cluster analysis using the spherical SOM. We can see from this figure that the borderline can be successfully expressed in the ambiguous shade region of the U-matrix.

3.3 Result with Non-equal Class Distributions

Table 1 lists the eigenvalues and the corresponding eigenvectors obtained from the variance-covariance matrices of the each class when the class distributions are expressed by using the variance-covariance matrices.

The maximum eigenvalue is that of setosa class and the minimum eigenvalue is that of vergicolor. From the result of this eigenvalues and equation (9), one can see that the magnitude of the effect of the non-equal class distributions on the decision borders depends on the eigenvalues and that the magnitude of the effect is the largest in the setosa class, the next largest in the vergica class and the least in the verginica class. Hence one can also see that the decision border between the vergica and vergicolor classes will be shifted toward the verginica side.

Fig.4 shows the decision borders determined with our approach on the polygon surface. The solid decision border with the nodes from b0 to b4 was determined by using equations (10) and (11) from the feature vectors which have been obtained with equation (3). The dotted decision border with nodes from b0T to b4T was determined by using equations (10) and (11) from the data which have been obtained based on the coordinate system proposed in [6]. The solid decision border in blue color with the nodes from Q0 to Q4 was determined obtained by using equations (9), (10) and (11)

from the feature vectors. When comparing the decision borders determined with the three methods, the values of three of the five nodes, b0T, b3T and b4T, on the decision border are a very good match with the corresponding values of b0, b2 and b3, but the other ones of the nodes, b1T and b4T, do not match with the correspond values of b1 and b4.

Table 1. Eigenvalues and the corresponding eigenvectors of each iris data class

	λ_1	λ_2	λ_3	λ_4
Setosa	22.441	10.414	4.904	0.047
Virginica	19.856	2.314	2.107	0.022
Vergicolor	6.677	1.734	1.084	0.011

Setosa (set)	$\boldsymbol{W}_1$	$\boldsymbol{W}_2$	$\boldsymbol{W}_3$	$\boldsymbol{W}_4$
	-0.011	-0.062	0.945	-0.322
	0.082	-0.119	-0.325	-0.935
	-0.920	0.372	-0.027	-0.119
	0.383	0.919	0.032	-0.094

virginica (ver)	$\boldsymbol{W}_1$	$\boldsymbol{W}_2$	$\boldsymbol{W}_3$	$\boldsymbol{W}_4$
	0.171	-0.852	0.025	0.493
	-0.139	0.148	-0.914	0.351
	-0.695	0.209	0.364	0.584
	0.685	0.455	0.177	0.540

vergicolor (gnc)	$\boldsymbol{W}_1$	$\boldsymbol{W}_2$	$\boldsymbol{W}_3$	$\boldsymbol{W}_4$
	0.444	-0.078	-0.753	0.479
	-0.243	0.909	0.145	0.306
	-0.726	0.370	-0.103	0.571
	0.465	0.176	0.633	0.593

On the other hand, the solid decision border with the nodes from Q0 to Q4 was mapped in a different region. The reason why the nodes from Q0 to Q4 were mapped in a different region is that some of input data remap out of phase on the polygon surface. Therefore some pairs of candidates with the non-equal class distributions should be reselected to decide the decision borders regardless to candidates selected in the equal class distributions.

Fig.5 shows two decision borders determined from some pairs of candidates with the non-equal class distributions by using our approach. As can be seen from this figure, the part of the decision border along the nodes from R0C to R4C shifts from the part of the decision border along the nodes from R0+ to R4+ even thought the part of the decision border along the nodes from R5C to R6C has slightly inverse tendency. This fact gives evidence in support of the fact that the magnitude of deviation from the decision border with equal class distributions depends on the magnitude of eigenvalues with the variance-covariance matrix of each class.

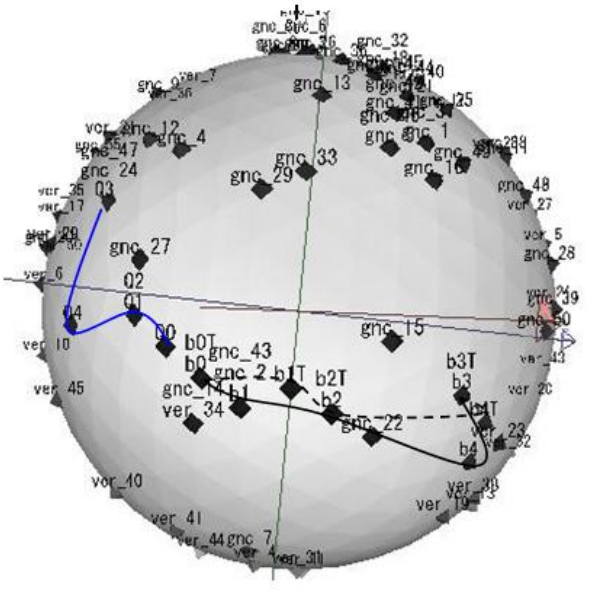


Fig. 4. A band of decision borders were obtained with equation (9)

The nodes from b0 to b4 on the polygon surface were calculated with the feature data, the nodes from b0T to b4T were calculated by using the coordinate system of [6] and the nodes from Q0 and Q4 were calculated by using equation (9).

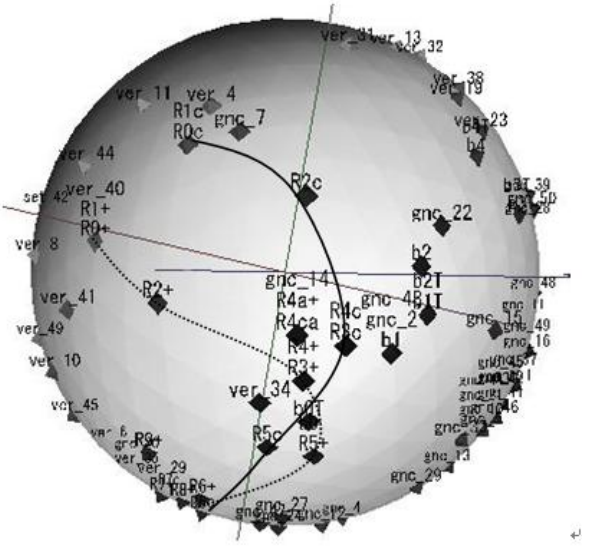


Fig. 5. Two decision borders were obtained with equation (9)

The solid decision border with the nodes from R0c to R6c was determined by using equation (9) and the dotted decision border with nodes from R0+ to R6+ was calculated when two eigenvalues are equal in the equation (9).

4 Conclusions

We have proposed an approach which approximates the decision borders on a spherical SOM with non-equal class distributions. The magnitude of the effect on the decision borders with the non-equal class distributions depends on the magnitude of the eigenvalues, especially maximum eigenvalue, of the variance-covariance matrices. Using the iris dataset of Fisher, we confirmed that our approach allows the magnitude of the effect on the decision borders to be successfully and qualitatively visualized.

References

1. Kohonen, T.: Self-Organizing Maps. Springer Series in Information Sciences, vol. 3. Springer, Heidelberg (2001)
2. Nakatsuka, D., Oyabu, M.: Usefulness of Spherical SOM for Clustering. In: Proceedings 19th Fuzzy System Symposium, pp. 67–70 (2003)
3. Tokutaka, H., Fujimura, K., Ohkita, M.: Cluster Analysis using Spherical SOM (in Japanese). Journal of Biomedical Fuzzy Systems Association 8(1), 29–39 (2006)
4. Ultsch, A., Mörchen, F.: ESOM-Maps: tools for clustering, visualization, classification with Emergent SOM, Depart. Of Computer Science University of Marburg, Research Report 46 (2005)
5. Ultsch, A., Guimaraes, G., Korus, D., Li, H.: Knowledge extraction from artificial neural networks and applications. In: Proceedings of TAT/ WTC 1993, pp. 194–203. Springer, Heidelberg (1993)
6. Matsuda, N., Tokutaka, H., Oyabu, M.: Decision of Class Borders on Spherical SOM and Its Visualization. In: Leung, C.S., Lee, M., Chan, J.H. (eds.) ICONIP 2009. LNCS, vol. 5864, pp. 802–811. Springer, Heidelberg (2009)
7. http://www.ics.uci.edu/~mlearn/databases/
8. http://www.somj.com/

Analysing the Structure of Semantic Concepts in Visual Databases

Mats Sjöberg and Jorma Laaksonen

Adaptive Informatics Research Centre,
Aalto University School of Science,
P.O. Box 15400, FI-00076 Aalto, Finland
firstname.lastname@aalto.fi

Abstract. In this paper we study how the Self-Organizing Map (SOM) can be used in analysing the structure of semantic concepts in visual data. We investigate two data sets with concept labels provided by humans, and unlabelled data for which we utilise automatically detected concept membership scores by using models trained on a labelled data set. By arranging the concept memberships of visual objects as components of a vector, they can be used as the feature space for training a SOM. A visual and qualitative analysis of the SOM distributions of different concepts is augmented with a quantitative analysis based on measuring the Earth Mover's Distance between the vector distributions on the 2D SOM surface. In particular we study the PASCAL VOC 2007 and TRECVID 2010 databases, which are two large image and video data sets.

Keywords: Self-Organizing Map, Earth Mover's Distance, concept detection, high-level features, image and video databases, visualisation.

1 Introduction

In this paper we study how the Self-Organizing Map (SOM) [2] can be used in analysing the structure of semantic concepts in visual data, in particular in the PASCAL VOC 2007 and TRECVID 2010 data sets. We compare how the *a priori* ground-truth concept data available in the training material and the *a posteriori* concept detections extracted from the testing material of the two databases behave in the mapping.

Early content-based image and video retrieval systems relied on measuring similarity solely using *low-level visual features* automatically extracted from the objects. However, such generic low-level features are often insufficient to discriminate content well on a higher conceptual level required by humans. This "semantic gap" is the fundamental problem in multimedia retrieval.

In recent years, *high-level features*, or *semantic concepts* have emerged as a partial answer to this problem. The main idea is to create semantic representations by extracting intermediate semantic levels (events, objects, locations, people, etc.) from low-level visual features using machine learning techniques.

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 338–347, 2011.

For example we can train detectors for semantic concepts such as "image containing a cat" or "video depicting an explosion or fire", which can then be used as building blocks for higher level querying into the database.

The accuracy of current state-of-the-art concept detectors can vary greatly depending on the particular concept and on the quality of the training data. Still, it has been observed that even concept detectors with relatively poor accuracy can be very useful in supporting high-level indexing and querying on multimedia data [1].

The rest of the paper is organised as follows. In Section 2 we explain the main idea of self-organising semantic concept vectors and how they can be analysed. Section 3 describes our experiments with two commonly used visual databases that are provided with semantic concepts and corresponding labelled examples. Conclusions are drawn in Section 4.

2 Self-Organisation of Semantic Concepts

In our experiments, each database is divided into a training set with a given ground truth of known concept members (labelled images or videos), and a test set for which concept membership is unknown. By training detectors on the known memberships we generate probability estimates of the corresponding concept membership of the test set objects. In this way the concept memberships are boolean in the training set (e.g. an image either depicts a cat or not), while the test set has probabilistic membership values (e.g. this image contains a cat with the probability 0.8). In this paper, we take the concept detectors as given. We use state-of-the-art detectors based on a fusion of Support Vector Machine (SVM) detectors on low-level visual features.

The ground truth labels and the concept detector outcomes give us two ways of analysing the semantic concepts of a database. A Self-Organizing Map can be trained either on the $\{0, 1\}$ values of the training set or the $[0, 1]$ probabilities from the test set. Given a set of objects (e.g. images) $x_1, \ldots, x_N$, and concepts $C_1, \ldots, C_K$, we can construct a concept vector for the object x_i:

$$\mathbf{c_i} = \begin{pmatrix} p_{i,1} \\ \vdots \\ p_{i,K} \end{pmatrix}, \tag{1}$$

where $p_{i,j} \in [0, 1]$ is the concept membership score of object x_i in concept C_j, often interpreted as the probability of the object belonging to the given concept. Thus, such a concept vector specifies concisely the concept membership of a given object to all concepts in the used concept vocabulary.

In the next step, we train Self-Organizing Maps using the concept vectors of the database objects as input. Such concept vectors will be contain only 1's and 0's in the training set where we have labelled data, but will be in the range $[0, 1]$ in the test set where we have detector outcomes. Because of the different types of input, we have trained two SOMs for each database, one for the training set, and one for the test set. Looking at the organisation of the training set can

give us insight in how concepts are correlated and the 2D relations can give us important clues to how the concepts group together into larger patterns.

A map of size $M \times M$ has model vectors $m_1, \ldots, m_{M^2}$, and if we take the j'th component of each model vector $m_{1,j}, \ldots, m_{M^2,j}$ we get a 2D distribution over the map surface for the concept C_j. Comparing such distributions between different concepts can provide insight into the semantic organisation of the database. While studying the organisation of the test set is less certain — since we only have estimated probabilities with quite varying accuracy — it may still be useful for analysing the overall organisation of the data set.

In addition to visually and qualitatively inspecting the concept distributions on the SOM surface we also wish to make a more quantitative analysis. In particular the "closeness" of concepts on the maps might be interesting. The different component distributions of the SOM model vectors represent different concepts, and thus the closeness of concepts could be estimated by calculating the distance between these distributions arranged as vectors — with short distances indicating semantically close concepts.

We first considered Euclidean distance between the component vectors, but this would not take into account the 2D organisation of the SOM map. Two distributions might be close by on the map but still be orthogonal. To take into account the 2D distribution of concepts on the map we instead decided to use the Earth Mover's Distance (EMD) [5] to calculate their dissimilarity. The EMD measures the minimum cost of turning one distribution into the other, where in this case the cost is the value that needs to be moved times the Euclidean distance over the 2D map surface. We used the C implementation for EMD[1] provided by the authors of [5].

3 Experiments

In the following subsections we present the resulting SOM maps for two different visual databases. For training the SOMs and processing the databases we have used the content-based retrieval and analysis framework PicSOM [4]. PicSOM by default uses Tree-Structured SOMs (TS-SOMs) [3] in which successively larger SOM layers are trained by fixing the previous layer and restricting the best-matching unit (BMU) search to the neighbourhood of the unit beneath the BMU in the previous layer. For our purpose, this gives us the advantage that we can visualise the SOM spatial surface at different levels of detail, by looking at different layers of the TS-SOM.

3.1 VOC 2007

The Pascal VOC 2007 database[2] contains almost 10,000 images with a training set of 2,500, evaluation set of 2,500 and test set of about 5,000 images. We used the training set and the test set to generate two Self-Organizing Maps. The

[1] http://ai.stanford.edu/~rubner/emd/default.htm

[2] http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html

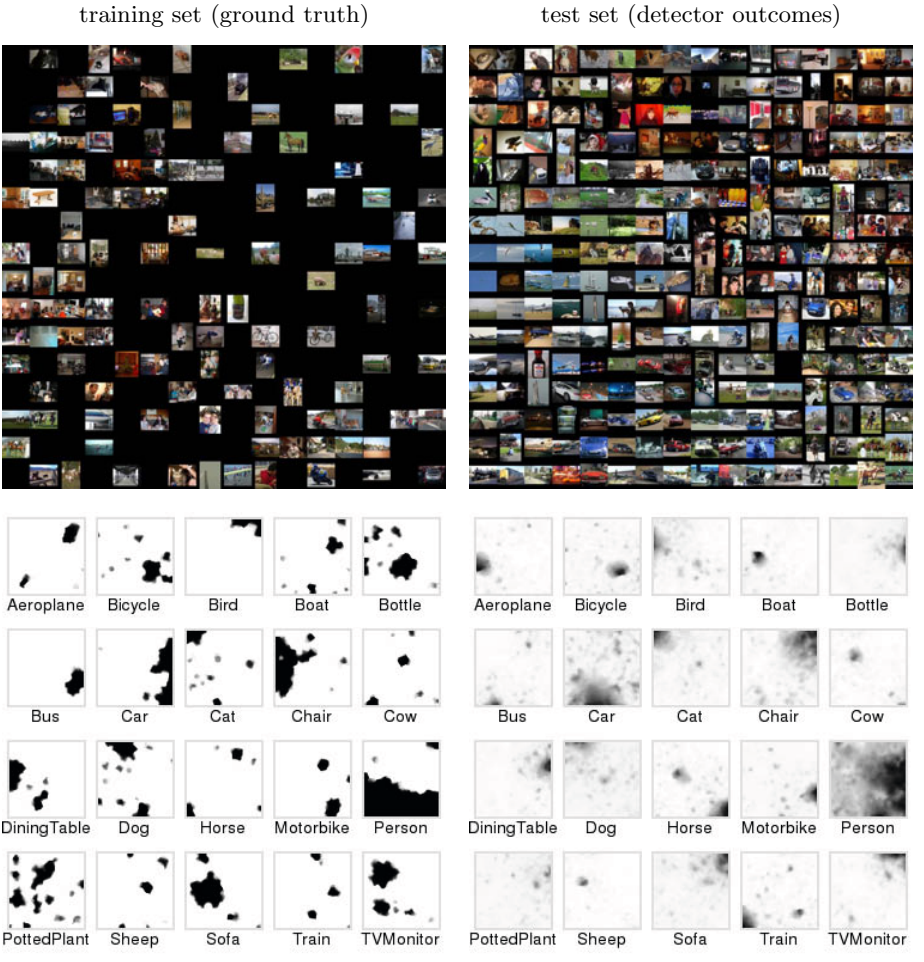


Fig. 1. SOM-based concept organisation of VOC 2007

training set is pre-labelled into 20 concepts; images may belong to 1 or more concepts. The concept detectors of the test set were simple SVM models on ColorSIFT (opponent colour space) histogram features [6] calculated from the images. We trained TS-SOMs with three layers of sizes 4×4, 16×16, and 64×64.

The trained SOMs are visualised in Figure 1, in the left column the training set (based on the labelled ground truth) and in the right column the test set (based on the detector outcomes). The top row shows the second surface layer (size 16×16), where each model vector is labelled with the image that has the closest concept vector. Not surprisingly the test set is much more evenly spread out, while the training set — with its boolean-valued vector components — is more sparse.

The concept distributions on the bottom-most TS-SOM layers are shown in the bottom row of Figure 1. The images are simply plots of the model vector

components, e.g. the first image, *Aeroplane* is a plot of the first component of each model vector. The gray scale values of the images are scaled so that white corresponds to 0.0 and black to 1.0. As can be expected, the training set concepts show mostly black and white, while the test set covers more of the gray scale.

At this point it should be noted that the two maps were trained separately and were initialised randomly. Thus the concepts are in general located at different regions in the training and test set SOMs. It can however be seen that the distributions are similar, e.g. the concept *Train* has three nodes in both data sets, but located at different points due to the random initialisation.

It is also interesting to note, especially while looking at the concept ground truth distributions (bottom left in Figure 1) how different concepts relate. E.g. the two classes of domestic pet animals, *Cat* and *Dog*, have a similar structure with two separate nodes in their distributions. The two distributions partially overlap over the two concepts which indicates some co-occurrence.

Other interesting phenomena can be found as well, e.g. that all forms of land-based transport have at least a partial presence in the lower right corner. *Bicycle*, *Bus*, *Car*, *Motorbike* and even *Train* have strong clusters there. Interestingly the *Bus* block seems to be like a "piece of a puzzle" that fits right into the larger distribution of the *Car* concept. On closer investigation it can be seen that they have a thin slice of common units which cause the areas to coincide on the map.

Also concepts, *Bottle* and *DiningTable* partially overlap, since a bottle can probably often be found on a dining table, and *Sofa* has overlaps with both of the two concepts. Also *TVMonitor* overlaps partially with *Sofa*. Furthermore *Bird* and *Aeroplane* seem to share the upper right corner, partially with *Boat*. Apparently these concepts sometimes co-occur.

The same phenomena are mostly repeated in the SOM based on the detector outcomes as well (bottom right in Figure 1), e.g. the two overlapping nodes of *Cat* and *Dog*. It is also clear that there is more of a visual organisation in the map based on the concept detectors, e.g. there are bluish images, often depicting the sky in the middle left border of the second layer map. This effect is because the concept detectors are trained on the visual features of the objects. The concept detectors are really indicating which images in the test set have the same visual features as those in the training set belonging to a particular class. It tries to learn which are those discriminating visual features in the training set. These may or may not generalise to the test set.

To get a more quantitative measure of the similarity of the concept distributions, we calculated the Earth Mover's Distance (EMD) between all concept pairs — excluding the concept *Person* which covers more than half of the map area (its *a priori* probability is 43%). Table 1 shows the 10 closest and 10 most distant concepts from the ground truth in the training set, while Table 2 shows the same, but calculated from the detector outcomes in the test set. The difference in EMD compared to the two closest concepts respectively the two farthest concepts are shown as percentages in the adjoining column.

The EMD ordering corresponds well with our intuitive understanding, and with the concept distributions shown in Figure 1. For the ground truth-based

Table 1. EMD comparison of concepts by the ground truth in VOC 2007

10 closest concepts		10 most distant concepts	
Sofa – TVMonitor		Bird – DiningTable	
Chair – TVMonitor	+9.2%	Bus – Chair	–5.1%
Chair – DiningTable	+18.2%	Bird – Chair	–6.1%
PottedPlant – Sofa	+20.9%	Bus – Cat	–6.3%
PottedPlant – TVMonitor	+21.8%	Car – Chair	–9.1%
Chair – Sofa	+29.0%	Bus – DiningTable	–10.3%
Bottle – PottedPlant	+30.3%	Car – DiningTable	–12.3%
DiningTable – Sofa	+31.2%	Car – Cat	–13.6%
Bus – Car	+33.7%	Bird – Sofa	–14.3%
Bottle – Sofa	+35.3%	Bird – Cow	–14.3%

Table 2. EMD comparison of concepts by detector outcomes in VOC 2007

10 closest concepts		10 most distant concepts	
Chair – Sofa		Train – TVMonitor	
Bottle – DiningTable	+51.8%	Aeroplane – DiningTable	–3.5%
Sofa – TVMonitor	+127.7%	Sofa – Train	–4.1%
DiningTable – PottedPlant	+179.6%	Aeroplane – Sofa	–4.5%
Chair – TVMonitor	+193.2%	Aeroplane – TVMonitor	–5.7%
Bottle – PottedPlant	+243.7%	Cat – Motorbike	–5.8%
Cat – Dog	+249.4%	DiningTable – Train	–6.3%
Chair – PottedPlant	+266.3%	Chair – Train	–6.8%
Chair – DiningTable	+325.5%	Aeroplane – Chair	–7.4%
Bird – Dog	+409.0%	Aeroplane – Bottle	–7.4%

results most concepts seem to be related to indoors domestic scenes, such as *Sofa*, *TVMonitor* and *PottedPlant* which are commonly found together in a living room, and *Chair* and *DiningTable* which are common in dining rooms. We also see *Bus* and *Car* which, as stated in the visual investigation of the maps, are close but overlap only a little. These are commonly seen in street scenes, but overlap only seldomly in the images of the database. However, they do overlap some times and the neighbourhood smoothing of the SOM algorithm causes the two distributions to appear nearby on the map.

The closest concepts based on the detector outcomes (Table 2) are similar, except for *Cat* and *Dog* and *Bird* and *Dog*. Especially the latter pair is quite interesting, and might be explained by the fact that both birds and dogs are often depicted outdoors with visually similar surroundings. Also it can be seen that for some reason the distance grows much steeper in the test set than in the training set, as shown by the percentage column.

The most distant concepts based on EMD are also not surprising, however here there are larger differences between the training and test sets. Especially *Aeroplane* seems to be distant from most other concepts, which might be explained by aeroplanes often being shown with a blue sky as a background, thus being visually very distinct from other images. This has an effect in the test

set, since it is based on concept detectors trained on low-level visual features, in particular a colour-based one in this case.

3.2 TRECVID 2010

One of the tasks in the annual TRECVID video retrieval evaluation [8] is to detect the presence of predefined *high-level features* (HLFs) [9] in broadcast videos that are already partitioned into shots. Our research group has participated since 2005, and in this paper we use the database from TRECVID 2010.

The TRECVID 2010 video data is taken from the Internet Archive collection[3]. A total of 130 concepts are provided, and the ground truth was specified by a collaborative annotation process among the participants[4]. The training data set contains about 120,000 video shots (200 hours) and the test set about 150,000 video shots (200 hours). Some videos did not belong to any concept and were dropped for these experiments.

As concept detectors we used our own developed for the TRECVID 2010 competition [7]. These are based on fusion of SVM detectors based on SIFT and ColorSIFT features calculated from a dense sampling of different spatial partitions of the key frame images extracted from the video shots. We trained TS-SOM with four layers of sizes $4 \times 4, 16 \times 16, 64 \times 64$ and 256×256. The SOM for the training set is visualised in Figure 2, the test set in Figure 3. Both figures show the second 16×16-sized layer with image labels representing the model vectors, and below that are the component distributions of selected concepts. The 10 closest and 10 most distant concept pairs measured by Earth Mover's Distance are shown in Table 3 for the ground truth and in Table 4 for the detector outcomes.

In the training set we can see that there are several larger clusters of concepts, e.g. suburban scenes form a large group close to the centre. Here we find e.g the concepts *Building*, *Car*, *Road*, *Streets*, *Suburban* and *Vehicle*. Also *Outdoor* overlaps this area. Pairs of these concepts also occur many times in the list of 10 closest concepts. The situation is similar with the detector outcomes (test set), but now these concepts occupy the lower right corner and are naturally more spread out. Another cluster covered by the *Outdoor* concept is in the upper left corner in the training set, e.g. *Landscape*, *Plant*, *Trees* and *Vegetation*. Again, in the test set, they are placed differently, in the middle of the bottom edge.

Looking at most distant concept pairs, it is not so strange that *Fem.Face-Closeup* (female human face closeup) is distant from other concepts, since a closeup image tends to fill the image with only one object excluding the possibility of finding other concepts. Again we find that the pair-wise distances grow more rapidly in the test set. Curiously, the concept *Canoe* is very distant from other concepts, probably because it is very rare – only 11 examples in the training set.

[3] http://www.archive.org/
[4] http://mrim.imag.fr/tvca/

Fig. 2. TRECVID2010, training set (ground truth)

Table 3. EMD comparison of concepts by the ground truth in TRECVID 2010

10 closest concepts		10 most distant concepts	
Road – Streets		Charts – Fem.FaceCloseup	
Walking – WalkingRunning	+21.9%	ComputerTVScreens – Fem.FaceCloseup	-3.2%
Anchorperson – Reporters	+32.4%	Fem.FaceCloseup – Maps	-4.0%
Car – GroundVehicles	+42.0%	Charts – Face	-9.2%
RoadwayJunction – Streets	+48.1%	Face – Maps	-9.9%
Car – Constr.Vehicles	+51.5%	Fem.FaceCloseup – OverlaidText	-10.1%
DemoProtest – PeopleMarching	+59.3%	Fem.FaceCloseup – Laboratory	-10.7%
Highway – RoadwayJunction	+70.8%	Beards – Landscape	-12.0%
Road – RoadwayJunction	+85.5%	Landscape – MalePerson	-12.3%
GroundVehicles – Motorcycle	+87.3%	Maps – SinglePerson	-12.6%

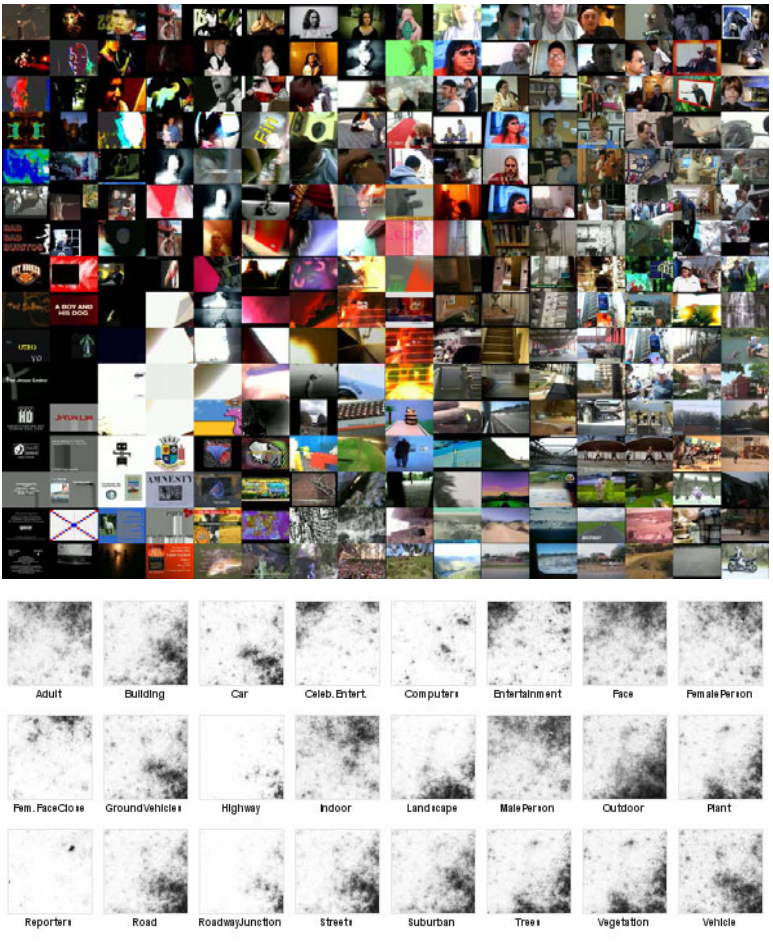


Fig. 3. TRECVID2010, test set (detector outcomes)

Table 4. EMD comparison of concepts by detector outcomes in TRECVID 2010

10 closest concepts		10 most distant concepts	
Car – Vehicle		Canoe – Celeb.Entert.	
GroundVehicles – Vehicle	+40.6%	Canoe – Singing	-0.5%
Outdoor – Swimming	+483.8%	Desert – Singing	-0.8%
Outdoor – Stadium	+651.7%	Canoe – Entertainment	-0.8%
Outdoor – Trees	+654.0%	Celeb.Entert. – Desert	-0.9%
Beards – Face	+684.9%	Desert – InstrumentalMusician	-1.0%
Constr.Vehicles – Vehicle	+817.7%	BoatShip – Celeb.Entert.	-1.2%
Adult – AsianPeople	+1018.7%	Desert – Entertainment	-1.3%
FemalePerson – Teenagers	+1375.6%	Canoe – InstrumentalMusician	-2.0%
Walking – WalkingRunning	+1886.3%	BoatShip – Singing	-2.1%

4 Conclusions

In this paper we have analysed the structure of semantic concepts in visual databases using Self-Organizing Maps. By constructing semantic vectors out of the labelled data, or using automatically extracted concept probabilities we can generate a self-organised 2D mapping of the concept space. In particular, we investigated two large-scale visual databases, the VOC 2007 image dataset and the video collection used in TRECVID 2010. We showed that interesting relationships can be found using visual analysis of the SOM surfaces, and also by measuring the similarity of the concept vector components. By using the Earth Mover's Distance the measure can also take into account the 2D organisation in the map.

While a simple correlation analysis of the concept labelling might yield partially similar results, we believe that the 2D organisation of the SOM provides a unique advantage. For example higher-level groupings of concepts are immediately visible on the map surface. The hierarchy of the TS-SOM might be used to further advantage here, by looking more closely at how the concepts merge when going upwards to smaller layers. This may be a fruitful topic for future research.

References

1. Hauptmann, A.G., Christel, M.G., Yan, R.: Video retrieval based on semantic concepts. Proceedings of the IEEE 96(4), 602–622 (2008)
2. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer Series in Information Sciences, vol. 30. Springer, Berlin (2001)
3. Koikkalainen, P.: Progress with the tree-structured self-organizing map. In: 11th European Conference on Artificial Intelligence, pp. 211–215 (1994)
4. Laaksonen, J., Koskela, M., Oja, E.: PicSOM—Self-organizing image retrieval with MPEG-7 content descriptions. IEEE Transactions on Neural Networks, Special Issue on Intelligent Multimedia Processing 13(4), 841–853 (2002)
5. Rubner, Y., Tomasi, C., Guibas, L.J.: The Earth Mover's Distance as a metric for image retrieval. Tech. Rep. CS-TN-98-86, Stanford University (1998)
6. van de Sande, K.E.A., Gevers, T., Snoek, C.G.M.: Evaluating color descriptors for object and scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(9), 1582–1596 (2010)
7. Sjöberg, M., Koskela, M., Chechev, M., Laaksonen, J.: PicSOM experiments in TRECVID 2010. In: Proceedings of the TRECVID 2010 Workshop, Gaithersburg, MD, USA (November 2010)
8. Smeaton, A.F., Over, P., Kraaij, W.: Evaluation campaigns and TRECVid. In: MIR 2006: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval, pp. 321–330. ACM Press, New York (2006)
9. Smeaton, A.F., Over, P., Kraaij, W.: High-Level Feature Detection from Video in TRECVid: a 5-Year Retrospective of Achievements. In: Divakaran, A. (ed.) Multimedia Content Analysis, Theory and Applications, pp. 151–174. Springer, Berlin (2009)

Mapping of the 3D Objects Using Computer Generated Hologram SOM

Hiroshi Dozono[1], Asami Tanaka[1], Shinya Nishijima[1], Hiroshi Tsukizi[1], and Masanori Nakakuni[2]

[1] Faculty of Science and Engineering, Saga University, 1-Honjyo Saga 840-8502 Japan
hiro@dna.ec.saga-u.ac.jp
[2] Information Technology Center, Fukuoka University, 8-19-1, Nanakuma, Jonan-ku, Fukuoka 814-0180 Japan
nak@fukuoka-u.ac.jp

Abstract. We propose the algorithm of CGH (Computer Generated Hologram)-SOM, in which SOM can organize the 3D information of objects on the map, using Fresnel hologram as memories of the units. The performance of CGH-SOM is also examined by experiments. Fresnel Hologram, which is conventionally implemented on photographic dry plates, can record the 3D information of an object, and can be used to recognize 3D objects. In the algorithm, we implemented Fresnel hologram as Computer Generated Hologram, which virtually simulates the photographic processing in the computer.

1 Introduction

Holograms are widely used for 3 dimensional(3D) data processing. For examples, 3D object displays, memories of 3D objects and hologram sheets are conventional applications[1][2]. In this paper, Computer Generated Hologram (CGH)[3] is used for the representations of 3D objects, and also applied to the recognition of 3D objects. In the conventional method of 3D object processing, the 3D objects are interpreted as the set of surfaces, edges and vertices, and the interpreted information is used for processing. Using CGH, the raw data of the points on the object, which can be obtained from 3D laser scanner or the results of the processing of 3D stereo camera, are directly used for processing. The computational costs of CGH may become large, and yet, it can be accelerated by SIMD processing in CPU or GPU, because the computation of CGH is simple numerical calculations. The recognition of 2D objects using CGH was reported in [4]. In our research, CGH is extended to the recognition of 3D objects.

In this paper, we propose a Self Organizing Map(SOM) which is composed of the CGH planes. Self Organizing map is the feed forward type neural network which consists of 2 layers, competitive layer and input layer without hidden layers. The learning method is unsupervised learning. After learning, SOM can map the multi-dimensional data on the 2 dimensional plane. CGH-SOM is a SOM which learns CGH of 3D objects in the units. After learning, the learned

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 348–356, 2011.

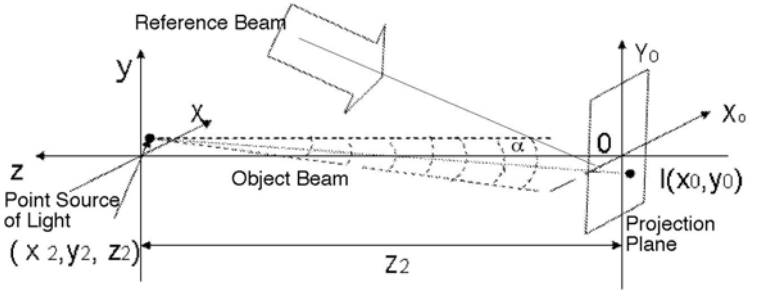


Fig. 1. Calculation of Fresnel CGH

3D objects can be mapped on the 2 dimensional plane and the resulting map can be used for the clustering and the recognition of 3D objects. The matching result between CGH and 3D objects is given as image data. The index to evaluate the image data is defined for introducing the metric among the objects. Some experimental results are shown using the artificial 3D objects and the actually measured data obtained from 3D laser scanner. Especially for the scanned data, sampling is required for hologram processing because too large number of points are obtained from the laser scanner and the difference of Z coordinates of scanned data affects the matching of hologram. For this purpose, the conventional SOM is applied to select the constant number of the points.

2 Hologram

Hologram is the record of the interference patterns of reflected beam of the object and reference beam. Hologram can record 3D information of the object, because it can record both amplitudes and phases. Additionally,the record of the hologram is highly redundant, so the recorded information can be retrieved even if some part of hologram is missed, and multiple holograms can be superposed in a hologram.

In the optical system, the hologram is recorded in the photographic dry plate. In optical information processing systems, hologram can be recorded in the high density liquid crystal panel. In this paper, we use the Computer Generated Hologram(CGH), which is virtually implemented in the computer. CGH can simulate all processes related to the holograms without using optical systems with changing the parameters.

2.1 Fresnel Hologram

To process the 2D information, Fourier Transform hologram can be applied. For processing 3D information, Fresnel Transform (FT) hologram [5] is required to handle depth information. The Fresnel CGH $I(x_0, y_0)$ projected on the plane $z = 0$ from point source of light located at (x_2, y_2, z_2) is calculated by the following equation where a_1 and a_2 are amplitudes of reference beam and object Beam respectively, and α is incidence angle of reference beam.

$$I(x_0, y_0, 0) = |R^2 + O^2|^2$$
$$= a_1^2 + \frac{a_2^2}{z_2^2} + \frac{2a_1a_2}{z_2} cos[k\{\frac{(x_0 - x_2)^2 + (y_0 - y_2)^2}{2z_2}$$
$$-(x_0 cos\alpha_{x1} + y_0 cos\alpha_{y1}) + z_2\}] \quad (1)$$

A 3D object is represented by the set of point source of light, and the summation of CGHs calculated for all points becomes CGH of the object.

2.2 Matched Filtering Using Fresnel Hologram

Fig.2 shows the schematic representation of matched filtering using Fresnel hologram. The object beam is Fourier transformed by the laser beam from lens, and projected on the Fresnel hologram. The projected beam is diffracted by Fresnel hologram. If the beam matches with the information on hologram, parallel beam is emitted as reference beam and converges to correlation spot by lens.

Computing in CGH, the beam projected on the hologram is computed as the Fresnel transform of the object beam, and calculated as follows. The object is sliced in Z-axis direction, and at each z_i, the distribution of the object beam is given as $g_i(x_0, y_o)$. Assume that the transfer function of point of light is given as $f_i(x_i, y_i)$ at (x_i, y_i) on hologram. Then, the Fresnel transform of $g_i(x_0, y_o)$ is given as follows where $\mathcal{F}$ denotes the Fourier transform.

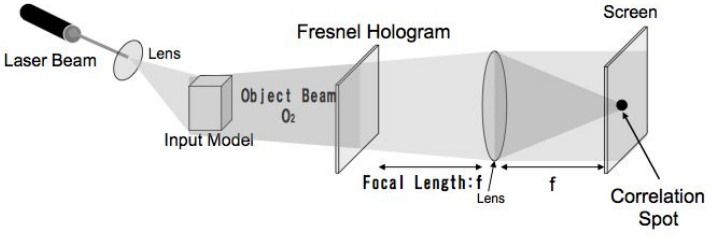


Fig. 2. Matched Filtering

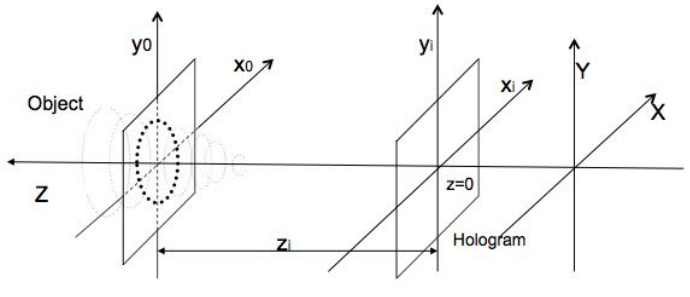


Fig. 3. Fresnel Transform of Input Object

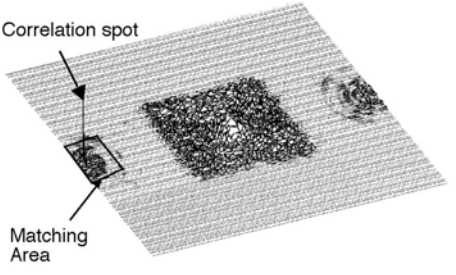


Fig. 4. Matching result for same 3D objects

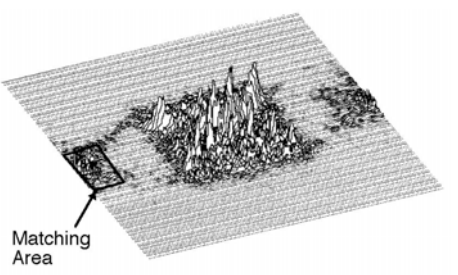


Fig. 5. Matching result for different 3D object

$$u_i(x_i, y_i) = g(x_0, y_0) \otimes f_{z_i}(x_i, y_i) \tag{2}$$
$$= \mathcal{F}^{-1}[\mathcal{F}[g(x_0, y_0)] \cdot \mathcal{F}[f_{z_i}(x_i, y_i)]] \tag{3}$$

The projected beam is calculated summing the u_is for all slices located at z_i. The diffracted beam is calculated by simply multiplying each pixel values of project beam and hologram, and the matching result projected on the screen is given as the inverse Fourier transform of the multiplied image. Fig.4 and Fig.5 show the results of CGH processing. Both figures show the pixel values as the hight. When input object matches with the object recorded in the hologram, correlation spot with extreme peak value is observed in the matching area. The position of the correlation spot depends on the parameters (beam angles, position of object, etc.) and shapes of object, and yet it will appear in a small area if the parameters are almost the same. The small values observed in the center of the image are transmitted beam. When the input object does not match, small peaks are observed in the matching area. To evaluate the matching results, the following two indexes are defined, where Ma is matching area.

– Maximum pixel value in matching area

$$Pmax = \max_{P \in Ma} P \tag{4}$$

– Difference of the maximum pixel value and average value

$$Pdif = Pmax - \overline{P}_{Ma} \tag{5}$$

$Pmax$ can not be absolutely evaluated, and $Pdif$ depends on the value of $Pmax$. Thus, these indexes are integrated as follows.

– Integrated index

$$Pm = Pmax/Pdif - 1 \tag{6}$$

The smaller value of Pm represents the better matching result.

3 CGH Self Organizing Map (CGH-SOM)

We are developing the 3D object database using holograms in our laboratory. As the extension of this system, CGH-SOM is proposed. CGH-SOM is the self organizing map which is composed of the units using CGH as memories. The learning algorithm is almost the same as that of conventional SOM with batch updates.

CGH-SOM Algorithm

Step1 : Preprocessing of 3D object data
For each Object Data D_i, calculate the Fresnel hologram H_i and Fresnel transform F_i used in matching process according to the equations (1) and (3) respectively.

Step 2 : Initialization of the map
For each unit U_{ij} on the map, generate the initial Fresnel hologram h_{ij} which is transformed from N random points of light sources.

Step 3: Matching
Match the holograms on the map and Fresnel transform of object data.
For each h_{ij}, for each F_k, generate the matching image $\mathcal{F}^(h_{ij} * F_k)$, and calculate $Pmax_k^{ij}$ and $Pdiff_k^{ij}$ using the equations (4) and (5) respectively.

Step 4: Finding winner
For each object D_k, find the winner unit W_k^{ij} using the matching results $Pmax_k^{ij}$, $Pdiff_k^{ij}$ and Pm_k^{ij}.

Step 5: Updating the units
For each winner W_k^{ij}, update the hologram h_{ij} and its neighbors using the following equation.

$$h_{ij} = h_{ij} + \eta fn(d)(H_k - hij) \tag{7}$$

where η is learning coefficient and $fn(d)$ is the neighboring function.

Repeat Step 3 - Step 5 by decreasing the size of neighbors and learning coefficient in pre-defined iterations

In step 1, the Fresnel hologram and Fresnel transform of the objects are computed before applied to SOM because the computational costs are very large. In step 2, because the random bitmap patterns do not matches any input data, the holograms which are generated from random point of light sources are used as initial patterns. In step 3, the input object data is matched with the holograms on the map based on the matching method of hologram processing. In step 4 and step 5,standard equation of SOM can be applied for updating holograms on the map, because the holograms can be superposed by simply summing the holograms.

4 Experimental Results

4.1 Experimental Results Using Artificial Objects

In this subsection, the experimental results using artificial objects are shown. At first, the objects which are shown in Fig. 6 are given as input data. The size

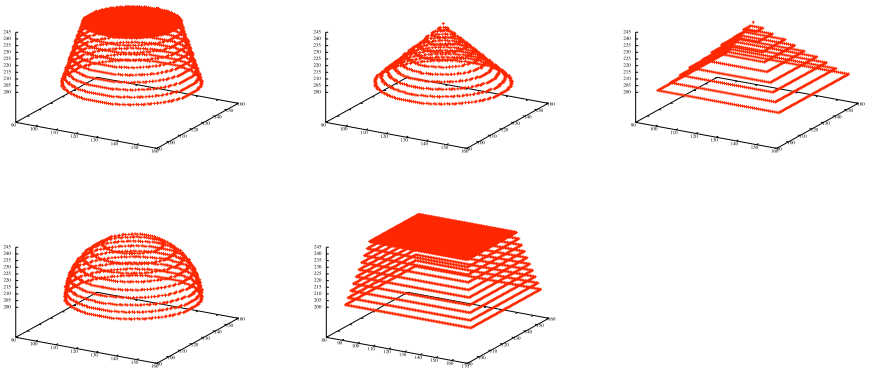


Fig. 6. Input data of artificial object 1-cylinder, 2-cone, 3-quadrangular pyramid, 4-hemisphere, 5-square column

1	1	5	2	2
1	5	5	5	2
3	5	5	2	2
3	3	5	4	4
3	3	3	4	4

Fig. 7. CGH-SOM for artificial object 1-cylinder, 2-cone, 3-quadrangular pyramid, 4-hemisphere, 5-square column

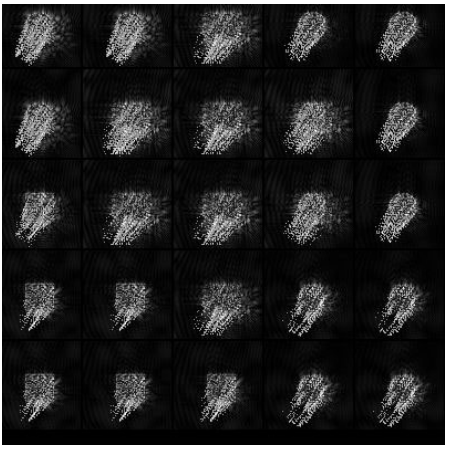


Fig. 8. Retrieved images from CGH-SOM for artificial object

of map is 5x5 and the number of iterations is 30. Fig.7 shows the map. Each number on the map denotes the number of the object which is the closest to the hologram associated to the unit. Each gray unit denotes the winner unit to which a object is mapped. Each object is clustered separately on the map. The map is organized using the metric in hologram space, so the similarity may not compatible with human sense. Fig.8 shows the images retrieved from the holograms learned on the map. The retrieved images are not clear because small amount of the superposes of the holograms may causes the big affection to retrieved images. However, the features of the objects, such as the shape of base plain, can be observed in the images and the images which are labeled same numbers look like similar. Fig. 9 shows the distribution of the Pm which is the

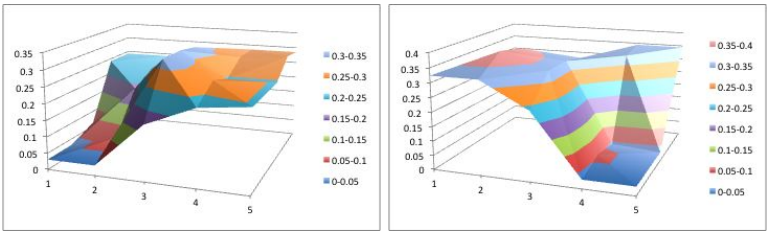


Fig. 9. Distribution of Pm on the map left-object 1, right-object 2

R23H26	R23H26	R26H23	R20H26	R20H26
R23H23	R23H23	R26H23	R26H23	R26H26
R23H23	R23H23	R20H20	R26H23	R26H26
R23H23	R20H20	R20H20	R20H20	R26H26
R26H20	R20H23	R20H23	R20H23	R23H20

Fig. 10. CGH-SOM for cones changing the radius(R) and hight(H)

similarity index between the holograms on the units and objects. These indexes become small for the units which are labeled as the object 1 and object 2 shown in Fig.7. The objects are clustered well based on the index Pm.

Next, the experiment was made changing the parameters of same object(cone). The radius of base plain and hight are changed to 20, 23 and 26. Fig.10 shows the results. All of the objects are clustered separately, and mapped on the different units.

4.2 Experimental Results Using Scanned Data

Next, we made experiments using scanned data form 3D laser scanner. 3D laser scanner generate 5000-9000 coordinates of surface points of the scanned object. The computational cost using all data for hologram processing becomes very large. For this problem, we used conventional SOM to reduce the number of points. Using the scanned data as input vectors, the coordinates data (x, y, z) are organized on the map. To eliminate the effect of randomness, the coordinates on the initial map are taken uniformly from x-y plane which includes the center of mass of the object, and batch learning algorithm is applied. Fig.11 shows the original scanned data and reduced data of the object cone, and Fig.12 shows the reduced data of other scanned objects. The map size is 25x25, so the number of points of reduced data is 645. The scanned data is uniformly reduced, and the feature of the object remains enough. Additionally, the matching result of CGH is heavily affected by changes of the values of Z coordinates. For this problem, the algorithm of SOM is modified as to generate discrete values of multiple number of $D_s = 2$ for Z coordinate.

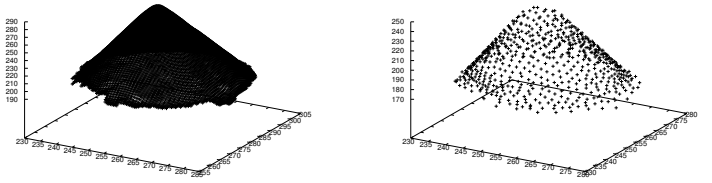


Fig. 11. The original scan data of cone and the reduced data by conventional SOM

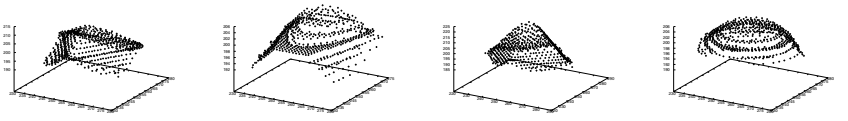


Fig. 12. The reduced data by conventional SOM (cube, halfpipe, pyramid, half spheral)

4	1	1	1	1
4	1	1	1	1
2	2	2	5	1
3	2	5	5	5
3	5	5	5	5

Fig. 13. CGH-SOM for scanned object 1-cone, 2-cube, 3-halfpipe, 4-pyramid, 5-half spheral

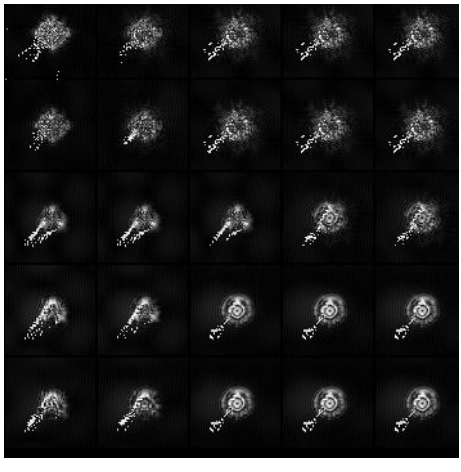


Fig. 14. Retrieved images from CGH-SOM for scanned object

Five Objects shown in Fig.11 and Fig.12 were used for training the map. Fig.13 and Fig.14 show the results. Using scanned data, the objects are also mapped separately on the map. Next, we made the experiment of object recognition. In this experiment, we used Supervised Pareto learning SOM[6]. Pareto learning SOM can integrate multiple objective functions in the learning process. As the objective functions $Pmax$ and $Pdif$ in equations (5) and (6) are used, and the category of each object is used for supervised learning. Two data are scanned for each object and the scanned data is different in each measurement. The 1st scanned data for each object are used for learning and the 2nd scan data are used for test. The experiments are made with changing the size of the map used

Table 1. Experimental results for object recognition O:success X:fail

	10x10	15x15	20x20	25x25	30x30	All(Raw)
cone	X	X	X	O	O	X
cube	O	O	O	O	O	X
halfpipe	O	O	O	O	O	O
pyramid	O	O	O	O	O	X
spheral	O	O	O	O	O	O

for reducing the scanned data. Table 1 shows the result. Using the map sized 10x10, 15x15 and 20x20 for reduction, the objects except the cone are recognized successfully, and yet cone can not be recognized. Using the map sized 25x25 and 30x30 for reduction, all objects are recognized successfully. Using the all raw data, only two objects are recognized because of the changes of Z coordinates in each scan data.

5 Conclusion

We proposed the Computer Generated Hologram SOM (CHG-SOM) for mapping 3D objects on the 2 dimensional plane. The 3D objects were successfully mapped using the metrics in the hologram space. Further research remains in order to make CGH-SOM applicable as a practical system in the following aspects. One, the matching method of CGH should be reconsidered because CGH matching is too rigid to guarantee its generalization ability. Two, the learning method of the units by CGH-SOM should be reconsidered. A superposed hologram, resulting from simply adding holograms, may match an incorrect object accidentally. Three, the computing cost of CGH SOM is high. If we use GPU computing in CGH-SOM, we can reduce the cost; we can do so significantly if we apply the optical computing method which uses laser beam and the hologram displayed on the liquid crystal display.

References

1. Yu, F.T.S., Lu, X.J.: A real-time programmable joint transform correlator. Opt. Commun. 52, 10–16 (2000)
2. Orihara, Y., Klaus, W., Fujino, M., Kodate, K.: Optimization and application of Hybrid level binary zone plates. Appl. Opt. 40(32), 5877–5885 (2001)
3. Dallas, W.J.: Computer-Generated Holograms. Digital Holography and Three Dimensional Display, pp. 1–49. Springer, Heidelberg (2006), doi:10.1007/0-387-31397-4_1
4. Yu, F.T.S., Jutammulia, S.: Optical pattern recognition. Cambridge Univ. Press, New York (1998)
5. Tudela, R., Mart N-Badosa, E., Badosa, A., Labastida, I., Vallmitjana, S., Juvells, I., Carnicer, A.: Full complex Fresnel holograms displayed on liquid crystal devices. Journal of Optics A: Pure and Applied Optics 5, 189–194 (2003)
6. Dozono, H., Nakakuni, M.: Application of Supervised Pareto Learning Self Organizing Maps to Multi-modal Biometric Authentication (in Japanese). IPSJ Journal 49(9), 3028–3037 (2008)

Analysing the Similarity of Album Art with Self-Organising Maps

Rudolf Mayer

Institute of Software Technology and Interactive Systems
Vienna University of Technology, Austria

Abstract. Digital audio has become an ubiquitously available medium, and for many consumers, it is the major distribution and storage form of music, accounting for a growing share of record sales. However, handling the ever growing size of both private and commercial collections becomes increasingly difficult. Users are often overwhelmed by the seemingly countless number of music tracks available. Computer algorithms that can understand and interpret characteristics of music, and organise and recommend them for and to their users can thus be of great assistance.

Therefore, a magnitude of research projects has been devoted in the last decade to automatically to make the sound characteristics of music machine interpretable, to e.g. allow for automatic categorisation of music, or to recommend track which are similar to the ones a user likes.

However, music is an inherently multi-modal type of data, and increasingly also other modalities of music have attracted interest from the community. The analysis of song lyrics and other textual data, such as websites or biographies associated with artists, together with social network data, has probably attracted most research in this area.

Album covers are another dimensionality characteristic to the music – they are often carefully designed by artists to convey a message consistent with the music and image of a band. Studies have shown that customers use album cover art as a visual cue when browsing music in regular record stores. We thus present a study on similarities in album covers, and their relations to certain styles and genres of bands. To this end, we employ Self-Organising Maps together with various visualisation techniques to automatically organise a music collection, and compare the results obtained when using both features from the music and the album covers.

1 Introduction and Related Work

Motivated by the vast spread of music in digital formats, Music Information Retrieval (MIR) has become a very important field to aid private and commercial users to organise their music collections. Important tasks are for instance automatic categorisation to organise music into predefined genres or moods, recommendation of music similar to a certain song (similarity retrieval), or the development of novel and intuitive interfaces to large music collections. A comprehensive overview of the research field is given in [12].

J. Laaksonen and T. Honkela (Eds.): WSOM 2011, LNCS 6731, pp. 357–366, 2011.

A strong focus on music's primary mode, the sound of a song, can be seen from the research in the last decade. A number of methods to extract descriptive features from the audio signal and to capture information such as rhythm, speed, amplitude or instrumentation have been proposed, ranging from low-level features describing the power spectrum to higher level ones. However, also other modalities associated with music have increasingly been employed for common MIR tasks.

Several research teams have been working on analysing textual information, often in the form of song lyrics and a vector representation of the term information contained in other text documents; an early example is a study on artist similarity via song lyrics [8]. Other cultural data is included in the retrieval process e.g. in the form of textual artist or album reviews [1].

The study in [2] suggests that 'an essential part of human psychology is the ability to identify music, text, images or other information based on associations provided by contextual information of different media'. It further suggests that a well-chosen cover of a book can reveal it's contents, or that lyrics of a familiar song can remind one of the song's melody. Album covers are generally carefully designed for specific target groups, as searching for music in a record shop is facilitated by browsing through album covers. There, album covers have to reveal very quickly the musical content of the album, and are thus used as strong visual clues [3]. Due to well-developed image recognition abilities of humans, this task can be performed very efficiently, much faster than listening to excerpts of the songs. This motivates and increased utilisation of this modality.

A multi-modal approach to query music, text, and images with a special focus on album covers is presented in [2]. In [5], a three-dimensional musical landscape via a Self-Organising Map is created and applied to small private music collections. Additional information like web data and album covers are used for labelling; album covers should facilitate the recognition of music known to the user. The covers are however not use in the SOM training itself.

The Self-Organising Map has also been applied to image data in the PicSOM project [6], for Information Retrieval in image databases, incorporating methods of relevance feedback.

In this paper, we want to empirically validate the hypothesis that album covers can provide cues to the type of music. We therefore organise a music collection with Self-Organising Maps using both music features and image features, and analysing the way the album covers are organised over the map. We investigate whether musical similarity and a similarity in the album cover art are correlated, and whether albums can really give a clue on the music they represent.

The remainder of this paper is structured as follows. Section 2 gives a brief outline over the SOM framework and visualisations employed, while Section 3 will introduce the feature sets employed to describe our music collection. Our experiments are then detailed in Section 5, before we conclude in Section 6.

2 SOM Framework

We employ the Java SOMToolbox framework[1], developed at the Vienna University of Technology, which provides methods for training SOMs. It further comprises an application for interactive, exploratory analysis of the map, allowing for zooming, panning and selection of single nodes and regions among the map. The application also allows to display digital images on top of the map grid, thus it can easily be used to visualise the album covers.

To facilitate the visual discovery of structures in the data, such as clusters, a wealth of approximatively 15 visualisations are provided, among them the U-Matrix [14] and Smoothed Data Histograms[13]. The former indicates distances between SOM nodes by colour-coding, and thus hints on cluster boundaries, while the latter visualises density in the data, also indicating clusters as nodes with high density. We also utilise the Thematic Classmap visualisation [9]. It which shows the distribution of meta-data labels or categories attached to the data vectors mapped on the SOM, by colouring the map in continuous regions, similar as e.g. a political map does for countries. To this end, it performs a Voronoi tessellation of the map space, and assigns colours to each Voronoi region to indicate how much a class contributes to the data items in that region.

To provide a partition of the map into separate clusters, the framework provides several clustering algorithms that can be applied on the vectors of the SOM nodes, such as Ward's linkage [4] algorithm.

3 Feature Sets

To obtain a vector representation of the music collection, we employ on the one hand methods that extract descriptive features from the audio snippets, as well as methods to extract features capturing information from the album covers.

3.1 Audio Features

The following descriptors are extracted from a spectral representation of an audio signal, partitioned into segments of 6 sec. Features are extracted segment-wise, and then aggregated for a piece of music computing the median (for RP and RH, see below) or mean (for SSD, see below) from multiple segments.

We describe the feature extraction algorithms very briefly, please refer to the references for further details.

Rhythm Patterns. The feature extraction process for a Rhythm Pattern (RP) is composed of two stages. First, the specific loudness sensation on 24 critical frequency bands is computed through a Short Time Fast Fourier Transform (FFT). The resulting frequency bands are grouped according to the Bark scale, and successive transformation into the Decibel, Phon and Sone scales takes place. This results in a psycho-acoustically modified Sonogram representation that reflects

[1] http://www.ifs.tuwien.ac.at/dm/somtoolbox/

human sound perception. In the second step, a Discrete Fourier Transform is applied to this Sonogram, resulting in a spectrum of loudness amplitude modulation per modulation frequency for each critical band. After additional weighting and smoothing steps, a Rhythm Pattern exhibits magnitude of modulation for 60 modulation frequencies on the 24 critical bands [7].

Rhythm Histogram. A Rhythm Histogram (RH) aggregates the modulation amplitude values of the critical bands computed in a Rhythm Pattern, and is thus a descriptor for general rhythmic characteristics in a piece of audio [7].

Statistical Spectrum Descriptor. The first part of the algorithm for computation of a **Statistical Spectrum Descriptor** (SSD), the computation of specific loudness sensation, is equal to the Rhythm Pattern algorithm. Subsequently at set of statistical values (mean, median, variance, skewness, kurtosis, min and max) are calculated for each individual critical band. SSDs therby describe fluctuations on the critical bands; they capture both timbral and rhythmic information. In a number of evaluation studies, SSD have often shown to be superior for musical genre classification tasks [7].

3.2 Image Features

Colour Histogram. This feature set computes the distribution of pixel values in the RGB colour space. For each colour channel, a histogram of values (from 0 to 255) is computed from all pixels in the image. To reduce the dimensionality, we employed binning of the values. 128 bins for each channels were determined as a good value through experimental evaluation in classification tasks. Thus the total dimensionality of such a feature vector is 384 dimensions.

Color Names. Colour names [16] are a level of abstraction on top of a colour histogram – the colour space is divided in the 11 basic colours black, blue, brown, gray, green, orange, pink, purple, red, white and yellow. Each pixel is associated with one of these colours, and then, as before, a histogram of values for the whole image is computed. This feature vector thus has eleven dimensions.

SIFT – Bag of Visual Words. Scale Invariant Feature Transform is a local feature descriptor which is invariant to certain transformations, such as scaling, rotation or brightness. The algorithm extracts interesting points in an image, which can then be used to identify similar objects. The points usually lie on high-contrast regions of the image, such as object edges. We utilise the algorithm presented in [15], which utilises a Harris corner detector and subsequently the Laplacian for scale selection. We created a 1024 dimensional codebook (Bag of Visual Words), capturing the relative distribution of the SIFT features.

4 Collection

Music information retrieval research in general suffers from a lack of standardised benchmark collections, being mainly attributable to copyright issues.

Nonetheless, some collections have been used frequently in the literature. These were howeber not usable for the study in this paper, as none of these collection comes with a complementary set of album covers, and additionally most collections either miss information about song title and artist, or are royalty free music from relatively unknown artists – for both cases, automated fetching album covers from the web is not feasible.

Therefore, we composed our own test collection containing both audio snippets and album covers, by crawling data from the webshop *amazon.com*, which provides rich information for their music shop. Considering the best-selling list from several different genres, for each album (or maxi-single) found, we downloaded the cover, and the 30 second audio snippet of the first song. We thereby skipped entries for which either the cover was of too poor quality (below 400×400 pixels), or the 30 second song snippets was missing. Amazon organises the contents of it its music shop into 25 top-level genres, with many sub-categories; songs may, and frequently are, assigned to multiple genres. We aimed at selecting rather diverse and non-overlapping genres, to achieve distinctive styles in the cover art, and thus chose genres such as 'Goth and Industrial Rock', 'Rap and Hip-Hop', 'Reggae', 'Country', 'Electronic', 'Classical music' and 'Blues'. Overall, the collection comprises more than 900 songs.

5 Experimental Analysis

We trained maps of the size 22×18 nodes, i.e. a total of 352 nodes, with each of the audio features. From a manual inspection, the map trained with SSD features seems to provide the best arrangement of music according to the authors perception, superior to RP and RH features.

This map is depicted in Figure 1, with the result of a clustering of the nodes superimposed on the map lattice.

It can be observed that the classical music (indicated by light-grey colour) is separated rather well from the other genres, being mostly located in the upper-right corner. This area also matches the boundary detected via the clustering of the map nodes using the Ward's linkage method. Gothic and Alternative rock music, indicated in green, is mostly located in the lower-left corner, though a few pieces are distributed on other areas as well. These pieces are mostly slow songs, using a lot of instrumentation found also in e.g. classical music, such as violins, and therefore most of these mapping patterns appear logical from a musical point of view. Reggae music (red) can be mostly found in the upper-left corner and upper-centre, often together with Hip-Hop (blue), with which it shares a lot of rhythmic and tempo characteristics. Jazz/Blues (dark-grey), which borrows many styles from other genres, is organised in a number of smaller, but in itself rather consistent, clusters. The distribution of these clusters all over the map is motivated by the nature of this genre, which is a confluence of several music traditions, and has incorporated many aspects of popular music. Electronic music (pink) shows no clear pattern, distributed in small groups all over the map.

In Figure 2, a Smoothed Data Histogram (SDH) visualisation of this map is depicted, with the 'Islands of Music' [13] metaphor, where islands represent

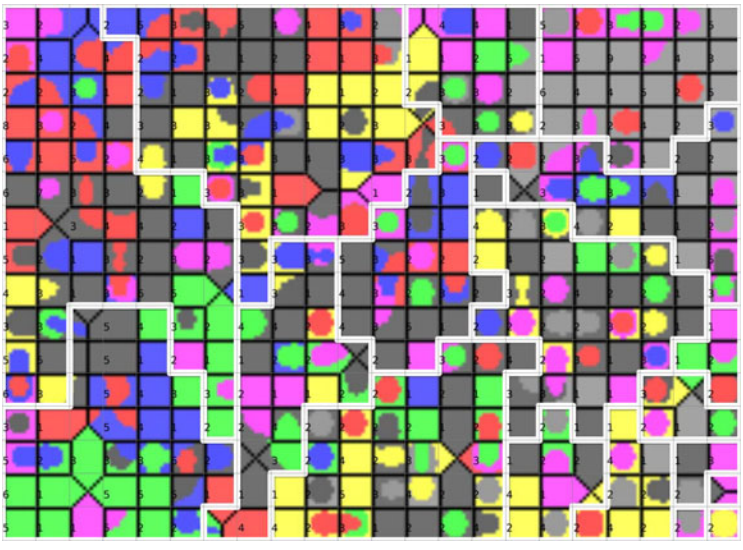


Fig. 1. Distribution of genres over the map with SSD audio features. Clusters obtained via Ward's linkage clustering of the nodes is indicated by white lines.

areas with high density. It can be seen that the arrangement of different genres correlates to some degree with the SDH, such as in the area of high-density in the upper right, which represents the cluster of classical music.

For a more detailed inspection, Figure 3 depicts 24 nodes in the upper-right corner of the map, the area containing mostly classical music; this section of the map contains a total of 64 songs. To indicate the genre, the class visualisation [9] from Figure 1 is also used in this illustration, using the same colours as background for the different genres as in Figure 1. On a first glance, there seems to be a certain coherence between the album covers. The most striking shared characteristics between the classical music album covers seems to be the frequent use of photos of people, in some cases the artists themselves, in other cases the musician interpreting the piece of music. These album covers generally follow a rather simple pattern for the background, consisting of few colours, and none or few objects. Many of the albums also simply feature a completely white background. The album covers on the top-edge of the figure mostly belong to the electronic genre; most of them share very similar instrumentation as the classical pieces, mostly the use of a piano or flutes. However, the album art seems to differ quite strongly, with a stronger use of dark colours, and more complex themes.

We can make similar observations for areas with Jazz and Country music, such as the area on the lower-right of the map, shown in Figure 4(a). Again, most of the covers feature portraits of the artists; however, there is a slightly different pattern in the background, using more darker colours, and thus allowing a subtle differentiation between the previous examples. Similar observations can be made for many Reggae songs.

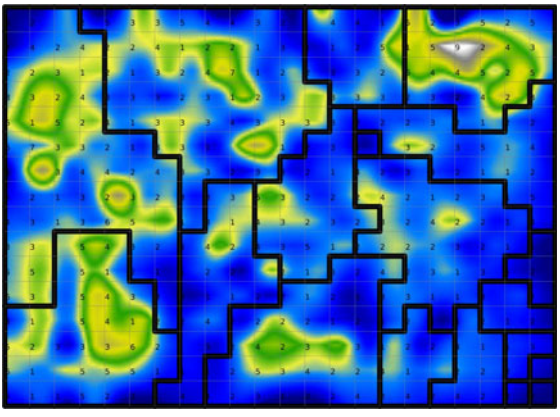


Fig. 2. Smoothed Data Histograms of the map with SSD audio features

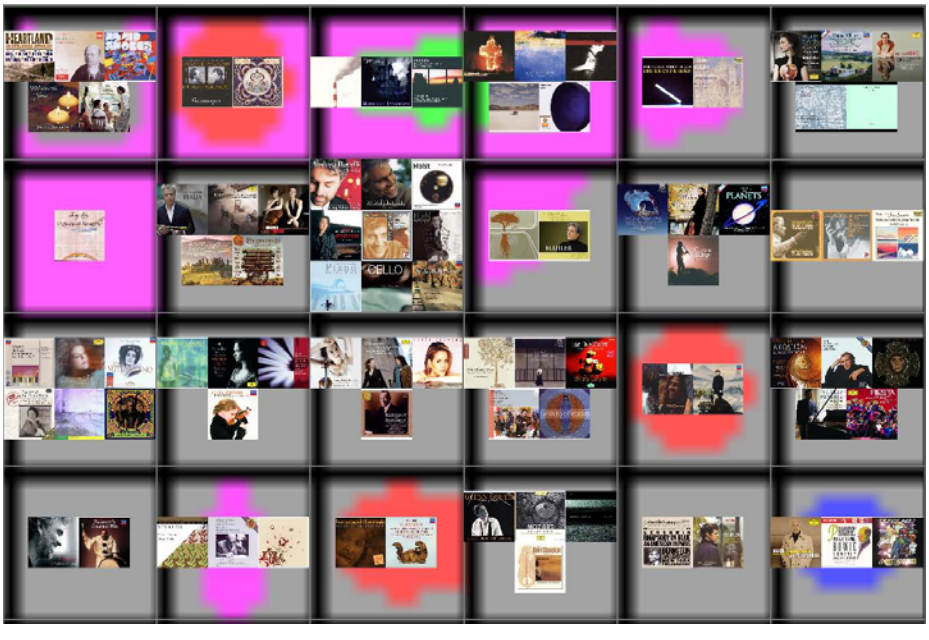


Fig. 3. Album covers in the cluster of classical music (SSD audio features, top-right corner)

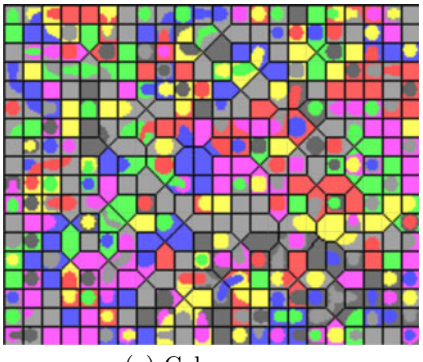
A cluster with songs from the Gothic and Alternative Rock genre is shown in Figure 4(b). Out of a total of 13 covers, only six show people, and in most cases, these portraits are heavily altered and appear more artificial. Noteworthy is also the use of many dark and flashy colours, which create a dark appearance.

While other areas of the map do not show that clear patterns, it can be concluded that at least to a certain degree, musical similarity as determined by

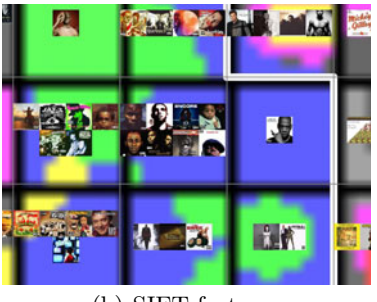
(a) Country, Jazz and Classical

(b) Gothic and Alternative Rock

Fig. 4. Album covers in the map with SSD audio features



(a) Color names



(b) SIFT features

Fig. 5. Music maps trained on the image features from the album covers

the SSD audio features and the vector projection of the SOM also coincides with some similarity in album cover art.

When organising the map with the image features, we build again on the assumption that album covers carry some clues about the music characteristics, and thus similar music should be located in neighbouring regions of the map. However, when using the simple features such as colour histograms or color names, the latter being depicted in Figure 5(a), this assumption is not fulfilled. While the organisation of album covers along the colour properties gives a nice overview, this arrangement does not match with the genres they belong to, as can be seen in Figure 5(a). There is basically no region in the map that shows a continuous area of similar music. We can thus conclude that for an interface to music, simple features such as the ones derived from colours are not sufficient.

Figure 5(b) depicts a section of a map trained with the SIFT BoV features. This section holds covers that, with a very few exceptions, depict people; further, most of the songs are from the Hip-Hop genre. It could thus be concluded that

SIFT BoV features can be useful to detect shapes of faces, which we identified earlier as an important aspect for several genres. We can also observe in some other areas that these features are very well working on depicting outliers, mostly albums with very complex cover art. However, similar observations as for the map with color names hold true – the features don't seem to be able to capture the complex similarities in the covers very well.

Finally, we applied the method described in [11], which allows for a analytical comparison of SOMs. It enables to identify differences in mappings obtained by different SOM trainings, by indicating which data items are mapped closely together in both maps. It can also be used to compare two maps trained on different features, for example on the music and song lyrics, as in [10]. Applying this method to the maps trained with the album cover features and the ones extracted from the music, we notice only a very small percentage of matches in the two different mappings – most of the songs that were mapped together in the music SOM are mapped to divergent areas in the album cover SOM.

6 Conclusions

We performed an analysis of the similarity of album art and the music they represent. To this end, we extracted audio features from the music, and image features from the album covers, and trained a set of SOMs with it. The SOM trained with the audio features revealed that in a number of cases, the musical similarity of the music is also reflected in the album covers, e.g. by the use of portraits or rather abstract objects, and also partly by the colours. The maps trained with the image features could, however, only reconfirm some of these similarities, when using the SIFT features to describe the visual content.

We thus conclude that while there is potential in using album covers for music information related tasks, there is a need for more powerful image feature descriptors. Such descriptors could be face detectors, more advanced use of points of interest features, and a combination of these features into a single descriptor.

References

1. Baumann, S., Pohle, T., Vembu, S.: Towards a socio-cultural compatibility of MIR systems. In: Proceedings of the 5th International Conference of Music Information Retrieval (ISMIR 2004), Barcelona, Spain, October 10-14, pp. 460–465 (2004)
2. Brochu, E., de Freitas, N., Bao, K.: The sound of an album cover: Probabilistic multimedia and IR. In: Bishop, C.M., Frey, B.J. (eds.) Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics, Key West, FL, USA, January 3-6 (2003)
3. Cunningham, S.J., Reeves, N., Britland, M.: An ethnographic study of music information seeking: implications for the design of a music digital library. In: Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 5–16. IEEE Computer Society, Washington, DC (2003)
4. Ward Jr., J.H.: Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association 58(301), 236–244 (1963)

5. Knees, P., Schedl, M., Pohle, T., Widmer, G.: An Innovative Three-Dimensional User Interface for Exploring Music Collections Enriched with Meta-Information from the Web. In: Proceedings of the ACM 14th International Conference on Multimedia (MM 2006), Santa Barbara, California, USA, October 23-26, pp. 17–24 (2006)
6. Laaksonen, J., Koskela, M., Laakso, S., Oja, E.: PicSOM–content-based image retrieval with self-organizing maps. Pattern Recogn. Lett. 21(13-14), 1199–1207 (2000)
7. Lidy, T., Rauber, A.: Evaluation of feature extractors and psycho-acoustic transformations for music genre classification. In: Proc. ISMIR, London, UK, September 11-15, 2005, pp. 34–41 (2005)
8. Logan, B., Kositsky, A., Moreno, P.: Semantic analysis of song lyrics. In: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME 2004), Taipei, Taiwan, June 27-30, 2004, pp. 827–830 (2004)
9. Mayer, R., Aziz, T.A., Rauber, A.: Visualising Class Distribution on Self-organising Maps. In: de Sá, J.M., Alexandre, L.A., Duch, W., Mandic, D.P. (eds.) ICANN 2007. LNCS, vol. 4669, pp. 359–368. Springer, Heidelberg (2007)
10. Mayer, R., Frank, J., Rauber, A.: Analytic comparison of audio feature sets using self-organising maps. In: Proceedings of the Workshop on Exploring Musical Information Spaces, in Conjunction with ECDL 2009, Corfu, Greece, pp. 62–67 (October 2009)
11. Mayer, R., Neumayer, R., Baum, D., Rauber, A.: Analytic comparison of self-organising maps. In: Príncipe, J.C., Miikkulainen, R. (eds.) WSOM 2009. LNCS, vol. 5629, pp. 182–190. Springer, Heidelberg (2009)
12. Orio, N.: Music retrieval: A tutorial and review. Foundations and Trends in Information Retrieval 1(1), 1–90 (2006)
13. Pampalk, E., Rauber, A., Merkl, D.: Using Smoothed Data Histograms for Cluster Visualization in Self-Organizing Maps. In: Dorronsoro, J.R. (ed.) ICANN 2002. LNCS, vol. 2415, pp. 871–876. Springer, Heidelberg (2002)
14. Ultsch, A., Siemon, H.P.: Kohonen's Self-Organizing Feature Maps for Exploratory Data Analysis. In: Proceedings of the International Neural Network Conference (INNC 1990), pp. 305–308. Kluwer Academic Press, Dordrecht (1990)
15. van de Sande, K.E.A., Gevers, T., Snoek, C.G.M.: Evaluating color descriptors for object and scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(9), 1582–1596 (2010)
16. Van De Weijer, J., Schmid, C.: Applying color names to image description. In: IEEE International Conference on Image Processing (ICIP 2007), vol. 3. IEEE, Los Alamitos (2007)

Author Index