A visualization technique for Self-Organizing Maps with vector fields to obtain the cluster structure at desired levels of detail

Georg Pölzlbauer
Department of Software Technology
Vienna University of Technology
Favoritenstr. 11-13, Vienna, Austria
E-mail:poelzlbauer@ifs.tuwien.ac.at

Michael Dittenbach
eCommerce Competence Center – ec3
Donau-City-Str. 1, Vienna, Austria
E-mail: michael.dittenbach@ec3.at

Andreas Rauber
Department of Software Technology
Vienna University of Technology
Favoritenstr. 11-13, Vienna, Austria
E-mail:rauber@ifs.tuwien.ac.at

***Abstract*— Self-Organizing Maps are a prominent tool for exploratory data analysis. In this paper, we propose a method of visualizing the cluster structure of the SOM based on the similarity of areas on the map, computed by aggregation of the distances of the underlying component planes of the codebook. The result can then be plotted on top of the map lattice with arrows that point to the closest cluster center, which is analogous to flow and vector field visualizations. A parameter is provided that allows fine-tuning of the granularity of the clustering, which can be adjusted according to whether a global or local view on the map is desired. We provide experimental results with a real-world data set where we discuss the effects of parametrization and the general applicability of our method, along with comparison to related techniques.**

I. Introduction

The Self-Organizing Map (SOM) [3] is a valuable tool in exploratory data analysis. It is a popular unsupervised neural network algorithm that has been used in a wide range of scientific and industrial applications [5]. In the research community, it has received much attention in the contexts of clustering, data mining, topology preserving vector projection from high dimensional input spaces (or feature spaces), and visualization of results. The SOM algorithm is computationally extremely light [1]. This projection can be visualized in numerous ways in order to reveal the characteristics of the underlying input data or to analyze the quality of the obtained mapping.

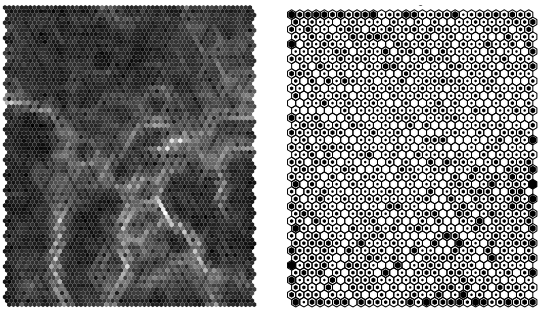
In this paper, we propose a novel visualization method that is visualized based upon vector field plotting. Another concept exploited by our method is the neighborhood kernel, which determines the mutual influence of nodes based on their distance on the map lattice. This kernel function is typically only used for training of the SOM. For each map node, we compute a vector that points towards the closest cluster center. We propose two methods of visualizing the results, a vector field plot, which can be seen analogous to flow visualization and gradient visualization, and we also derive a dual representation that emphasizes on the cluster structure of the map. The SOMs used for demonstration purposes and experiments are trained on the well-known Phonetic data set, consisting of 1962 samples in 20 variable dimensions that describe features of phonemes recorded from human speech.

The rest of this paper is organized as follows. Section 2 describes several visualization techniques for SOMs and related work, most notably U-Matrix and clustering algorithms. In Section 3, our novel visualization method is introduced, along with a description of its properties and interpretations. Section 4 presents experimental results, where the the influence of choices of neighborhood kernel, neighborhood radius and map size are investigated. Finally, Section 5 gives a short summary of the findings presented in this paper.

II. Related Work

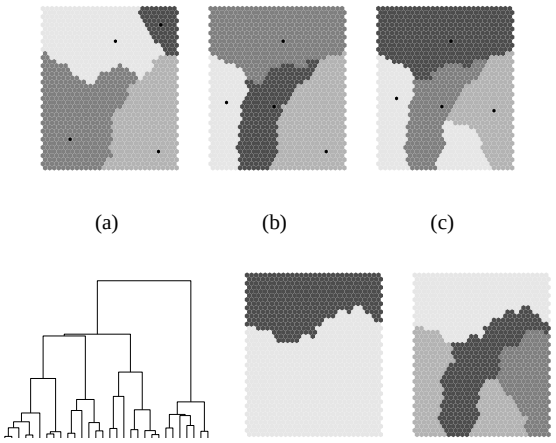
We briefly provide an overview of related visualization concepts for SOMs. Most commonly, component planes and the U-Matrix, which both take only the prototype vectors and not the data vectors into account, are applied to visualize the map. Component planes show projections of individual dimensions of the codebook vectors. If performed for each component, they are the most precise and complete representation available. However, cluster borders cannot be easily perceived, and high feature space dimensions result in lots of plots, a problem that many visualization methods in multivariate statistics, like scatterplots, suffer from. The U-Matrix technique [10] is a single plot that shows cluster borders according to dissimilarities between neighboring units. The distance between each map unit and its neighbors is computed and visualized on the map lattice, usually through color coding.

Recently, an extension to the U-Matrix has been proposed, the U*-Matrix [12], that relies on yet another visualization method, the P-Matrix [11]. Other than the original, it is computed by taking both the prototype vectors and the data vectors into account and is based on a concept of data density around the model vectors. It is designed for use with Emergent SOMs [10], which are SOMs trained with a high number of map units compared to the number of data samples. Interestingly, both the U*-Matrix and our novel method, among other goals, aim at smoothing the fine-structured clusters that make the U-Matrix visualization for these large SOMs



(a) (b)

Fig. 1. 30×40 SOM: (a) U-Matrix, (b) Hit histogram



(a) (b) (c)

(d) (e) (f)

Fig. 2. Clustering the 30×40 SOM: (a–c) k-means ($k = 4$), the black dots correspond to the map unit most similar to the cluster center; (d) Ward's Linkage Dendrogram for the top 30 merges, (e) Ward's Linkage: 2 Clusters, (f) Ward's Linkage: 4 Clusters

less comprehensible, although the techniques are conceptually totally different.

Methods that rely more heavily on the distribution of the data on the map are hit histograms and Smoothed Data Histograms [6], and recently proposed methods that directly show the density as graphs on top of the map [7]. Other visualization techniques include projections of the SOM codebook with concepts like PCA or Sammon's Mapping. For an in-depth discussion, see [13].

Hierarchical clustering [2], [9], [8] is related to our method with respect to the possibility to parameterize the desired level of detail. Our method also allows for fine-tuning to visually emphasize either a more local or global view on the clustering structure of the SOM.

Figure 1 shows hit histogram and U-Matrix visualizations for a map consisting of 30×40 map units that has been trained on the Phonetic data set. This SOM will serve as the main example throughout this paper. Before training, zero-mean-unit-variance scaling has been performed. It can be seen from the U-Matrix that the upper part of the map is very coherent, while it is difficult to observe a cluster structure from the lower two thirds. The hit histogram reveals that this SOM is almost evenly populated, because the number of map units (1200) is close to the number of data samples (1962). The U-Matrix is very helpful in providing an initial overview, but it is limited to comparing dissimilarities only between adjacent map nodes. We aim to extend this concept such that each map unit is compared to an area in its vicinity.

Clustering of the SOM codebook itself [14] by either partitional or hierarchical clustering with different numbers of clusters along with a graphical representation of the results can be very beneficial for understanding the map. Figure 2 shows results for clustering algorithms applied to the prototype vectors of the SOM. Three results are depicted for k-means performed with $k = 4$, along with the dendrogram and partitioning at levels 2 and 4 obtained from Ward's Linkage. The k-means visualizations show considerably different results due to the non-deterministic nature of this algorithm, which sometimes converges to sub-optimal local minima. However, some of the regions are within the same cluster in all of the three figures which indicates a strong contingency in these areas, such as in the upper left corner. The dendrogram shows that according to this linkage metric, the choice of either 2 or 4 clusters seems plausible because of the large margin to the next level of the hierarchy. In some cases, such as Ward's Linkage at level 4, the clusters contain non-adjacent areas of the map resulting from folding of the map during the training process. The borders obtained from the discussed clustering methods vary considerably, and in crisp clustering, the borders do not indicate whether the clusters are very distant or not. It is one of the aims of our method to visualize the extent of similarity between regions, such that the coarse cluster structure can be perceived as well as the fine differences.

Neighborhood kernel functions, which are an integral part of any SOM training process, are incorporated in our method. An example for an application outside the learning process is the SOM Distortion [4], that has been shown to be the energy function of the SOM in certain cases.

III. SOM Visualization by Vector Fields

In this section, we describe the computation algorithm and basic properties of our visualization technique. In the previous section, we have hinted at some of the aims: To obtain a visualization that allows fine-tuning between a local and global clustering, and that is comparable to the U-Matrix family of methods, but taking more than just the adjacent neighbors into account, aggregating over a large region. Further, we wish to obtain a pointer to the most similar units seen from each individual map node. Drawing these arrows on top of the map lattice results in a visualization analogous to gradient vector fields where units are repelled from or attracted to each other.

We begin by defining the formal framework. A two-dimensional SOM consists of a number M of map units ξ_i arranged in an equidistant manner, where the index i lies between 1 and M. Each of the map units is linked to a model vector m_i of input dimension N. Each of the m_i

is linked to the output space by its position on the map. To distinguish between feature space and map lattice, we explicitly write ξ_i for the position vector of map unit that represents prototype vector m_i; the index i connects input and output space representation. We denote the horizontal and vertical coordinates of the map unit as ξ_i^u and ξ_i^v, respectively. Thus, the distance between two prototype vectors m_i and m_j, or ξ_i and ξ_j, can be determined both in input and output space:

$$d_{feature}(m_i, m_j) = ||m_i - m_j||_{feature} \tag{1}$$

where $||.||_{feature}$ is a suitable distance metric for input space, and

$$d_{map}(\xi_i, \xi_j) = \sqrt{(\xi_i^u - \xi_j^u)^2 + (\xi_i^v - \xi_j^v)^2} \tag{2}$$

is the Euclidean Distance between nodes ξ_i and ξ_j, i.e. roughly the number of nodes that lie between them. Note that $d_{feature}$ can be used to calculate distances between prototype vectors and input samples, while d_{map} can only be used to measure the distance between map nodes.

One of the key differences between the SOM and other prototype-based unsupervised learning methods such as k-means is the concept of adjacency in output space. The usually two-dimensional topology of the map is fixed, and close regions of the map are expected to represent similar data samples. This concept of adjacency is introduced through the neighborhood kernel that determines the influence of the prototype vectors among each other. Our visualization technique heavily depends on this kernel as a weighting factor, which is a parameterized function that takes the distance between two map units on the lattice as input and returns a scaling factor that determines by which amount the map unit is updated for each iteration during training. The parameter the kernel depends on is the neighborhood radius σ which controls the width of the kernel function, with high values leading to flat stretched-out kernels and low values resulting in sharply peaked functions. It only returns non-negative real numbers and is monotonically decreasing. We denote the kernel function as $h_\sigma(d_{map}(\xi_i, \xi_j))$. The Gaussian kernel, resembling the well-known bell-shaped curve, is probably the most frequently used kernel for the SOM:

$$h_\sigma^{\text{Gauss}}(d_{map}) = \exp\left(-\frac{d_{map}^2}{2\sigma}\right) \tag{3}$$

We will use the kernel function as a weighting factor that allows us to compute the similarity in terms of input space distance of map units that are close to each other on the map. Our technique plots arrows for each map unit like in gradient field visualizations. A unit's arrow points to the region where the most similar prototype vectors are located on the map. The length of this arrow reflects the ratio of how much the area it is pointing to is more similar to it than the opposite direction.

For each of the M nodes ξ_i, the two-dimensional vector a_i is computed. As with the coordinates of the map nodes, a_i can be decomposed into u and v components, denoted as a_i^u and a_i^v, respectively. For both axes, we compute the amount of dissimilarity along positive and negative directions. Our method determines these vectors in a two-step process: First, the computations for each map unit are performed separately for positive and negative directions of axes u and v, and finally, these components are aggregated by a weighting scheme to gather the coordinates of a_i.

In the following, we adopt the notation that the i-th vector a_i will be computed, and formulas requiring two input vectors, i.e. ξ_i and ξ_j always refers to the vector or number with subscript i to be the one for which the computation is performed. In the first step, we have to obtain the angle α that identifies the direction of ξ_j seen from ξ_i on the map lattice. This is defined in basic trigonometry as

$$\alpha(\xi_i, \xi_j) = \arctan\left(\frac{\xi_j^v - \xi_i^v}{\xi_j^u - \xi_i^u}\right) \tag{4}$$

Since not only the angle, but also the distance between ξ_i and ξ_j is of interest, this distance is projected onto the u and v axes, after the neighborhood kernel has been applied to it to weight the influence of distant units accordingly:

$$\omega^u(\xi_i, \xi_j) = \cos(\alpha(\xi_i, \xi_j)) \cdot h_\sigma(d_{map}(\xi_i, \xi_j)) \tag{5}$$

and

$$\omega^v(\xi_i, \xi_j) = \sin(\alpha(\xi_i, \xi_j)) \cdot h_\sigma(d_{map}(\xi_i, \xi_j)) \tag{6}$$

This results in a distribution of the neighborhood kernel values among the two axes according to the position of ξ_i and ξ_j on the map and serves as a weighting factor in the following steps. To illustrate the meaning of ω, consider that the ξ_j is located directly below ξ_i. ω^u will then be zero, while ω^v will absorb the whole neighborhood kernel value, measured negatively. The neighborhood kernel relies on the width parameter σ, which determines the influence of far-away map units. The importance of this parameter will be investigated in the next section.

In the next step, the distance in input space is taken into account. It is decomposed in positive and negative directions for both axes for each pair of map units ξ_i, ξ_j, resulting in splitting the dissimilarity of m_j from m_i into four quadrants:

$$\delta_+^u(\xi_i, \xi_j) = \begin{cases} d(m_i, m_j) \cdot w^u(\xi_i, \xi_j) & \text{if } \omega^u(\xi_i, \xi_j) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

and

$$\delta_-^u(\xi_i, \xi_j) = \begin{cases} -d(m_i, m_j) \cdot w^u(\xi_i, \xi_j) & \text{if } \omega^u(\xi_i, \xi_j) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

For the sake of compactness, d was written instead of $d_{feature}$; δ_+^u denotes the contribution of map unit ξ_j's dissimilarity in positive direction along u, and δ_-^u in negative direction. The definition of δ_+^v and δ_-^v follows analogously. For example, a map unit ξ_j that lies to the lower right of ξ_i results in $\delta_-^u(\xi_i, \xi_j) = \delta_+^v(\xi_i, \xi_j) = 0$, and positive values for $\delta_+^u(\xi_i, \xi_j)$ and $\delta_-^v(\xi_i, \xi_j)$ according to the distance in output space, weighted by the neighborhood kernel, and also its distance in feature space, which is directly measured by the factor $d_{feature}$.

For all of the four quadrants, the sum of contributions δ in both directions is computed for each node ξ_i

$$\rho_+^u(\xi_i) = \sum_{j=1\ldots M, j\neq i} \delta_+^u(\xi_i, \xi_j) \tag{9}$$

and

$$\rho_-^u(\xi_i) = \sum_{j=1\ldots M, j\neq i} \delta_-^u(\xi_i, \xi_j) \tag{10}$$

Again, ρ_+^v and ρ_-^v are defined analogously. The variable $\rho_+^u(\xi_i)$ indicates how much m_i is dissimilar from its neighbors on the side in the positive u direction. In a gradient field analogy, this value shows how much it is repelled from the area on the right side. If, for example, ρ_+^u is high compared to ρ_-^u, the arrow will ultimately be more likely to point to the left, since a high value ρ_+^u indicates that the model vectors in positive u direction are strongly different from m_i.

In the next step, the u and v coordinates of vector a_i are determined by aggregating negative and positive components. This is performed by computing the ratio between ρ_+ and ρ_-. But before this can be done, a non-trivial normalization scheme has to be performed, because units at the borders of the map lattice would have components pointing outside of the map equal to zero, which is not desired. The sums of the neighborhood kernel weights ω_i pointing in positive and negative directions are

$$\omega_+^u(\xi_i) = \sum_{j=1\ldots M, j\neq i} \begin{cases} \omega^u(\xi_i, \xi_j) & \text{if } \omega^u(\xi_i, \xi_j) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

and

$$\omega_-^u(\xi_i) = \sum_{j=1\ldots M, j\neq i} \begin{cases} -\omega^u(\xi_i, \xi_j) & \text{if } \omega^u(\xi_i, \xi_j) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

At last, the u component of the gradient vector a is computed as the normalized ratio

$$a_i^u = \frac{\rho_-^u(\xi_i) \cdot \omega_+^u(\xi_i) - \rho_+^u(\xi_i) \cdot \omega_-^u(\xi_i)}{\rho_+^u(\xi_i) + \rho_-^u(\xi_i)} \tag{13}$$

and analogously for the v direction. The weighting factor ω_+^u is multiplied with the component in the other direction to negate the effects of units close to the border in which case the sum of the neighborhood kernel is greater on one side. If this normalization would be omitted, the vector a would be biased towards pointing to the side where units are missing, always preferring to point outside of the map where no dissimilarity can come from. For map units in the center of the map's u-axis, where ω_+^u and ω_-^u are approximately equal, (13) can be approximated by this simpler formula

$$a_i^u \approx \mu \cdot \frac{\rho_-^u(\xi_i) - \rho_+^u(\xi_i)}{\rho_+^u(\xi_i) + \rho_-^u(\xi_i)} \tag{14}$$

where μ is a constant factor equal to $\frac{\omega_+^u + \omega_-^u}{2}$ and is approximately the same for all units in the middle of an axis.

The key calculations are performed in 9 and 10. Here, we briefly discuss different scenarios of how ratios and proportions of ρ_+ and ρ_- influence a_i.

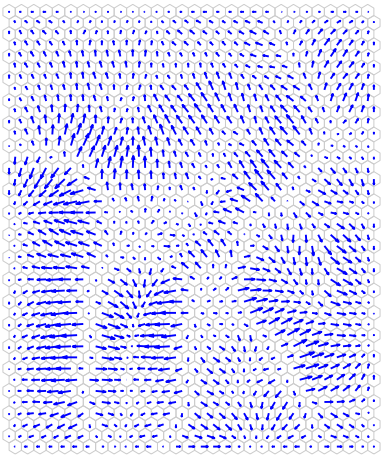
- If ρ_+ and ρ_-, i.e. negative and positive dissimilarity contributions are roughly equal, the resulting component of a_i will be close to zero, no matter how large their absolute values are. The vector is equally repelled from both sides, resulting in a state of equilibrium.
- If the $\rho_+ > \rho_-$, a_i will point into the negative direction. The reason for this is that the prototype vectors on the negative side of the axis are more similar to the current map unit than on the positive side.
- If one side dominates, but the second side still has a high absolute value, the normalization performed in the denominator of (13) decreases the length of the vector.
- Ultimately, the ratio of ρ_+ and ρ_- decides the length of the arrow. If the dissimilarity is distributed to 50% in each direction, it would be in an equilibrium state; in the hypothetical example that the codebook vectors in the positive direction are identical to m_i, this would result in the longest possible arrow in positive direction.

In the previous section, the 30×40 SOM trained on the Phonetic data set has been introduced. Figure 3(a) shows our visualization technique with a Gaussian kernel with $\sigma = 5$. If compared to the U-Matrix in Figure 1(a), it can be seen that the longest arrows are observed near the cluster borders, pointing to the interior of their cluster and away from these borders. It can be seen that adjacent units, for which the arrow points in different directions, are clearly along a cluster border. The length of the arrows indicates the sharpness of a border. In the middle of these transitions, arrows are sometimes drawn with almost no distinguishable length or direction. The corresponding prototype vectors are likely to be very far away from either cluster, and are referred to as interpolating units, since they do not represent any data vectors in a vector quantization sense, but are only a link connecting two distant data clouds. Cluster centers also have small dot-like arrows pointing in no distinguishable direction, but the difference is that the surrounding arrows are pointing in their direction, and not away from them as is the case with interpolating units.

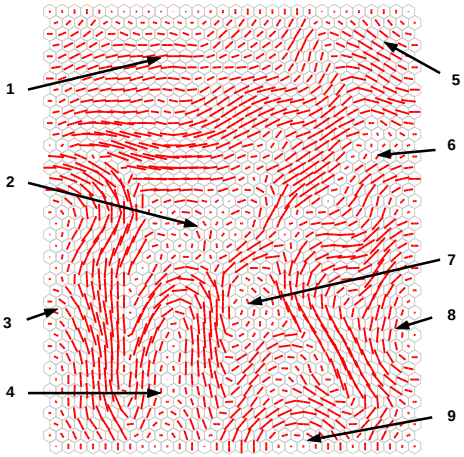
Another property of this visualization is that the units on the edges of the map never point outside of it, which is desired and stems from the normalization performed in 13.

The two-dimensional vector representation of a_i allows for a similar visualization with little modification. Instead of the arrows, the orthogonal hyperplane, which in this case is again a line, can be computed and visualized. So instead of plotting arrows pointing towards the closest cluster center, the borders that separate adjacent clusters can be shown. The length of the arrow is maintained and corresponds to the length of the border. The result is depicted in Figure 3(b). The emphasis of this dual representation is stressing cluster borders, while information on directions is omitted.

Computationally, the method is more expensive than the U-Matrix. It relies on pairwise distance calculation of all the map units, resulting in $O(M^2)$ complexity, which is the same as hierarchical clustering, compared to $O(M)$ for U-Matrix. In our experience, this has not been a problem, since the number of map units in the maps we use lies in the magnitude



(a)



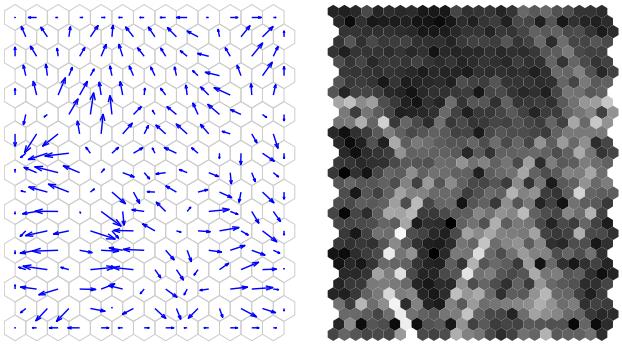
(b)

Fig. 3. 30×40 SOM trained on Phonetic data, depicted with Gaussian neighborhood kernel and width $\sigma = 5$: (a) Arrow representation that shows directed similarities, (b) Dual representation that shows cluster borders, with indicators for likely cluster centers

of 1000, which is computationally reasonable with modern computers, and visualization of much larger vector fields is limited by displaying capabilities of monitors and printers anyway. For reducing the number of distance calculations, the cut-off Gaussian kernel function could be considered. Since the Gaussian kernel is exponentially decreasing with higher distance, distance calculations could be omitted outside a certain radius due to negligible influence on the overall result, leading to better efficiency.

IV. Experiments

The empirical findings from experiments with our method are presented in this section. We show the effects of applying it to SOMs of varying size, but the same underlying data set. Next, we investigate how the neighborhood kernel parameter σ influences the results. Finally, we compare our method to related techniques that have been described in Section II.



(a) (b)

Fig. 4. 13×17 SOM trained on Phonetic data (a) Arrow representation with parameter $\sigma = 2$ and Gaussian kernel, (b) U-Matrix

Our first experiment concerns the number of codebook vectors, i.e. the size of the map. The data vectors remain the same for both maps, and we use the Phonetic data for this experiment. The smaller version of the SOM consists of 13×17 units, and the larger one of 30×40 units. In the former case, the number of data vectors (1962) is much larger than the number of map units (221), thus the vector quantization properties of the SOM are emphasized. The visualizations for the smaller version are depicted in Figure 4. The U-Matrix and vector field plots for the larger map are shown in Figures 1(a) and 4(a), respectively. In the smaller SOM the gap between the upper part and the lower part of the map can clearly be distinguished, as in the bigger SOM. Also, the lower part is also clearly more heterogeneous. However, the larger version of the SOM gives more insight into the structure of the data. Transitions and gradual changes in directions and length can be distinguished more easily at this higher granularity.

Next, we examine the effects of tuning parameter σ. In Figures 5 and 3, the large Phonetic SOM is visualized with three different values for σ. Figures 5(a), (b) show the two methods for $\sigma = 1$. The visualization with this width is the one most closely related to the U-Matrix technique, since only direct neighbors are emphasized, while the influence of slightly more distant units is neglected. Of all the visualizations shown here, these two are chiseled the most and are least smooth. The frequent changes in direction of neighboring arrows is due to the very local nature of this kernel. In Figure 3 the visualization is shown for $\sigma = 5$, where the increased neighborhood radius produces a smoothing effect over the vector field. Here, changes in direction between close arrows can be better distinguished and result in a visually more comprehensible picture. The set of arrows is perceived as a whole and as less chaotic. It gives the impression of visualizing a somewhat more global structure. Finally, the visualization for $\sigma = 15$ is depicted in Figures 5(c) and (d), where only big clusters can be distinguished. The effect of the width parameter σ can be summarized as follows: For a value of 1, the cluster representation is very similar to the U-Matrix, which is the method relying mostly on local differences. With higher values

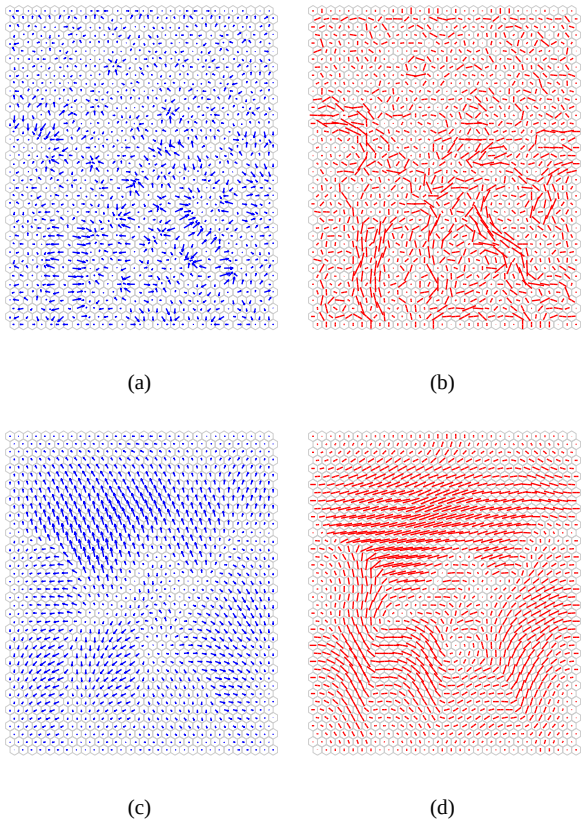


Fig. 5. 30×40 SOM trained on Phonetic data (a) Vector field representation with $\sigma = 1$, (b) Border representation with $\sigma = 1$, (e) Vector field representation with $\sigma = 15$, (f) Border representation with $\sigma = 15$

of σ, the kinds of perceived cluster structures gradually shift from local to global. The choice of σ has a very deep impact on this visualization method and is dependant on the map size. Further experiments have shown that good choices are close to one tenth of the number of map units in the axis of the map lattice with fewer map units, but it also depends on the desired level of granularity.

Finally, we compare our method to clustering methods performed on top of the SOM. In Figure 2, three results for k-means with $k = 4$ are given, along with results for Ward's Linkage at levels 2 and 4. The border representation in Figure 3(b) shows the most probable cluster centers obtained by our method. The border outlines are strongest where k-means and Ward's Linkage also have bordering clusters. Higher choices of σ correspond to clustering with fewer clusters, which can be seen by comparing Figure 2(e) with Figure 5(d) for an example of coarse clustering. What our method can not achieve, however, is identifying similar regions that are outside of the range of the neighborhood radius, i.e. map units that are close in input space, but mapped to far-away areas on the map. For example, the labels "3" and "9" in Figure 3(b) are merged in many of the clustering examples, for example Figure 2(c). The reason for this is that clustering of the codebook does not take the neighborhood on the map into account, and frequently finds clusters that are not adjacent, while our method aims at finding dissimilar neighboring regions. The phonemes mapped to region "3" are mostly "A"s, the area labeled with "9" is occupied primarily by "U"s, both vowels, with cluster "4" in between populated by "N"s.

V. Conclusion

In this paper, we have proposed and demonstrated a novel method of visualizing the cluster structure of Self-Organizing Maps. Our method is distantly related to hierarchical clustering methods and the U-Matrix. It is based on the neighborhood kernel function and on aggregation of distances in the proximity of each codebook vector. It requires a parameter σ that determines the smoothness and the level of detail of the visualization. There are two choices for depicting it, either as gradient field where arrows point towards the closest cluster center, or as border visualization that indicates how grave a transition is between neighboring regions. Our experiments have shown that this method is especially useful for maps with high numbers of units, and that the neighborhood radius σ has a major impact on the outcome.

Acknowledgements

Part of this work was supported by the European Union in the IST 6. Framework Program, MUSCLE NoE on Multimedia Understanding through Semantics, Computation and Learning, contract 507752.

References

[1] E. Cuadros-Vargas, R. Francelin Romero, and K. Obermayer. Speeding up algorithms of SOM Family for Large and High Dimensional Databases. In *Workshop on Self organizing Maps*, 2003.

[2] M. Dittenbach, D. Merkl, and A. Rauber. The growing hierarchical self-organizing map. In *Intl. Joint Conf. on Neural Networks 2000*, Como, Italy, 2000.

[3] T. Kohonen. *Self-Organizing Maps, 3rd edition*. Springer, 2001.

[4] J. Lampinen and E. Oja. Clustering properties of hierarchical self-organizing maps. *Journal of Mathematical Imaging and Vision*, 2(2–3):261–272, 1992.

[5] M. Oja, S. Kaski, and T. Kohonen. Bibliography of self-organizing map (SOM) papers: 1998-2001 addendum. *Neural Computing Surveys*, 3:1–156, 2001.

[6] E. Pampalk, A. Rauber, and D. Merkl. Using smoothed data histograms for cluster visualization in self-organizing maps. In *Intl. Conf. on Artifical Neural Networks*, 2002.

[7] G. Pölzlbauer, A. Rauber, and M. Dittenbach. Advanced visualization techniques for self-organizing maps with graph-based methods. In *Intl. Symp. on Neural Networks (ISSN'05)*, 2005.

[8] A. Rauber, D. Merkl, and M. Dittenbach. The growing hierarchical self-organizing map: Exploratory analysis of high-dimensional data. *IEEE Transactions on Neural Networks*, 13(6):1331–1341, 2002.

[9] A. Rauber, E. Pampalk, and J. Paralic. Empirical evaluation of clustering algorithms. *Journal of Information and Organizational Sciences (JIOS)*, 24(2):195–209, 2000.

[10] A. Ultsch. Data mining and knowledge discovery with emergent self-organizing feature maps for multivariate time series. In *Kohonen Maps*. Elsevier, 1999.

[11] A. Ultsch. Maps for the visualization of high-dimensional data spaces. In *Workshop on Self Organizing Maps*, 2003.

[12] A. Ultsch. U*-matrix: a tool to visualize clusters in high dimensional data. Technical report, Dept. of Mathematics and Computer Science, Philipps-University Marburg, 2003.

[13] J. Vesanto. *Data Exploration Process Based on the Self-Organizing Map*. PhD thesis, Helsinki University of Technology, 2002.

[14] J. Vesanto and E. Alhoniemi. Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, 11(3):586–600, 2000.