Improved Interpretability of the Unified Distance Matrix with Connected Components

To Appear Proceedings of DMIN'11

Lutz Hamel and Chris W. Brown

***Abstract*— Self-organizing maps have been adopted in many fields as the data visualization method of choice. The unified distance matrix is the *de facto* standard for evaluating and interpreting self-organizing maps. In large, high-dimensional problems clusters can be difficult to identify in the plain unified distance matrix. Here we introduce an enhanced version of the unified distance matrix in which clusters are easier to see and interpret. In this enhanced version we view the self-organizing map as a planar graph where the clusters are connected components of this graph. Using the transitive properties of connectedness and exploiting the fact that each component has a minimal node where the gradient on the unified distance matrix is equal to zero we can transform these connected components into stars with the minimal node as the internal node. In order to avoid unnecessary fragmentation of the components we apply a kernel based smoothing algorithm to the unified distance matrix. Our enhanced unified distance matrix is then the smoothed original unified distance matrix with the star components overlaid. The result is an easily interpretable self-organizing map. We perform a number of experiments on synthetic as well as real-world data that highlight the increased visual power of this enhanced unified distance matrix.**

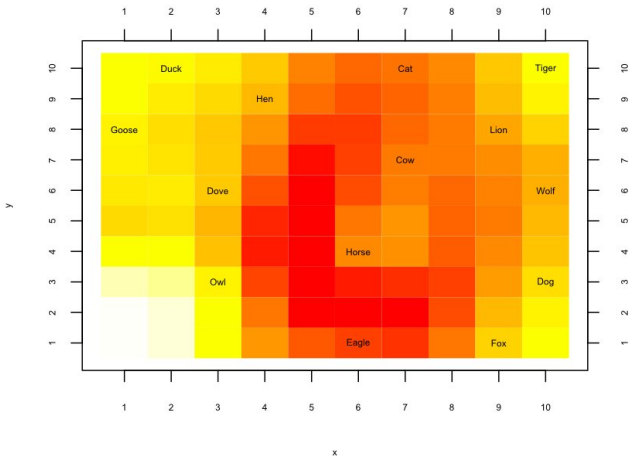

Fig. 1. A unified distance matrix.

Lutz Hamel is with the Department of Computer Science and Statistics, University of Rhode Island, Kingston, RI 02881, USA (email: hamel@cs.uri.edu).

Chris W. Brown is with the Department of Chemistry, University of Rhode Island, Kingston, RI 02881, USA (email: cbrown@chm.uri.edu).

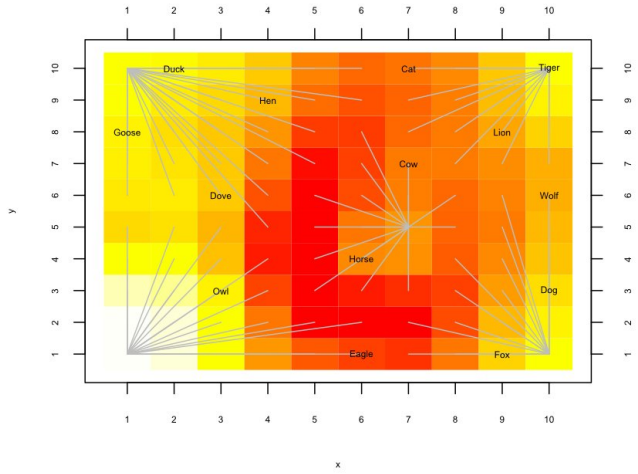

Fig. 2. An enhanced unified distance matrix.

I. Introduction

Data visualization is a powerful method to get an intuitive understanding of the data set at hand. Self-organizing maps (SOM) have been adopted as the method of choice for data visualization in many fields [1] [2]. The unified distance matrix (UMAT) is the de facto standard for evaluating and interpreting self-organizing maps. During the training phase of a SOM the values of its neural elements are computed in such a way that elements next two each other on the two-dimensional map are also close to each other in the space spanned by the attributes of the training data. In this way the two-dimensional map represents a topology preserving image of the high-dimensional training data. The UMAT visualizes the relative distances between the neural elements in training data space using a heat map: lighter colors represent neural elements that are close together in training data space and darker colors represents neural elements that are further apart. See Figure 1 for an example of an UMAT. The example is due to Kohonen [2] where each animal is described by a 10-dimensional feature vector with attributes such as the number of legs, if fur is present or not, and if the animal can swim. In Figure 1 we can clearly identify two strong clusters; one on the left side of the map clustering the birds and one on the right side of the map clustering the mammals. However, it turns out that there are more clusters that can

only be seen on the UMAT by a very careful inspection: the bottom left corner represents birds of prey whereas the top left corner represents birds that do not hunt. On the right side of the map we have clusters of hunters with fur and in the middle we have animals that are not hunters with hooves.

In this paper we introduce connected components as a way to improve the visibility of such clusters. Figure 2 is the UMAT appearing in Figure 1 with the connected components of the map overlaid. Notice that the clusters become immediately visible as the eye is guided toward seeing the clusters. These guides proved essential in our own work dealing with high dimensional spectroscopic data where data sets with hundreds of attributes are not uncommon [3], [4]. In order to obtain reasonable interpretations of these data via UMATs we had to construct large SOMs with thousands of neural elements; an approach not unlike Ultschs approach to constructing maps with emergent SOM [5]. The connected components proved extremely helpful when interpreting these large maps.

The paper is structured as follows. Section II provides a very brief overview of the standard SOM algorithm. In section III we develop connected components and describe how they are implemented in our current library. In section IV we describe a number of experiments. The first set of experiments is based on Ultsch's FCPS library and the second set of experiments is based on real world data sets we used in some of our work. We discuss related work in section V and we conclude with section VI providing some observations and pointers to further research.

II. Self-Organizing Maps

Self-organizing maps [2] were introduced by Kohonen in 1982 and can be viewed as tools to visualize structure in high-dimensional data. Self-organizing maps are considered members of the class of unsupervised machine learning algorithms, since they do not require a predefined concept but will learn the structure of a target domain without supervision.

Typically, a self-organizing map consists of a rectangular grid of neural elements. Multidimensional observations are represented as vectors. Each neural element in the self-organizing map also consists of a vector called a reference vector. The dimensions of the reference vectors on the map match the dimensionality of the observations. The goal of the map is to assign values to the reference vectors on the map in such a way that all observations can be represented on the map with the smallest possible error. However, the map is constructed under constraints in the sense that the reference vectors cannot take on arbitrary values but are subject to a smoothing function called the neighborhood function. During training the values of the reference vectors on the map become ordered so that similar reference vectors are close to each other on the map and dissimilar ones are further apart from each other. This implies that similar observations will be mapped to similar regions on the map. Often reference vectors are referred to as centroids, since they typically describe regions of observations with similarities.

The training of the map is carried out by a sequential regression process, where $t = 1, 2, \ldots$ is the step index. For each observation $\mathbf{x}(t)$ at time t, we first identify the index c of some reference vector which represents the best match in terms of Euclidean distance by the condition,

$$c = \underset{i}{\operatorname{argmin}} \|\mathbf{x}(t) - \mathbf{m}_i(t)\|. \tag{1}$$

Here, the index i ranges over all reference vectors on the map. The quantity $\mathbf{m}_i(t)$ refers to the reference vector at position i on the map at time step t. Next, all reference vectors on the map are updated according to the following rule where index c is the reference vector index as computed above,

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + h_{ci}[\mathbf{x}(t) - \mathbf{m}_i(t)]. \tag{2}$$

Here h_{ci} is the neighborhood function that is defined as follows,

$$h_{ci} = \begin{cases} 0 & \text{if } |c-i| > \beta \\ \eta & \text{if } |c-i| \leq \beta \end{cases} \tag{3}$$

where $|c - i|$ represents the distance on the map between the best matching reference vector at position c and some other reference vector at position i, β is the neighborhood distance and η is the learning rate. It is customary to express η and β also as functions of time t. The above computation is usually repeated over the available observations many times during the training phase of the map. Each iteration is called a training epoch.

One of the advantages of self-organizing maps is that they have an appealing visual representation as the 2-dimensional unified distance matrix as seen in Figure 1. Each square in the map represents a reference vector. As before, the colors on the map represent the relative distances between reference vectors: light colors indicate short distances and dark colors indicate long distances. Contiguous areas of light colors represent strong clusters.

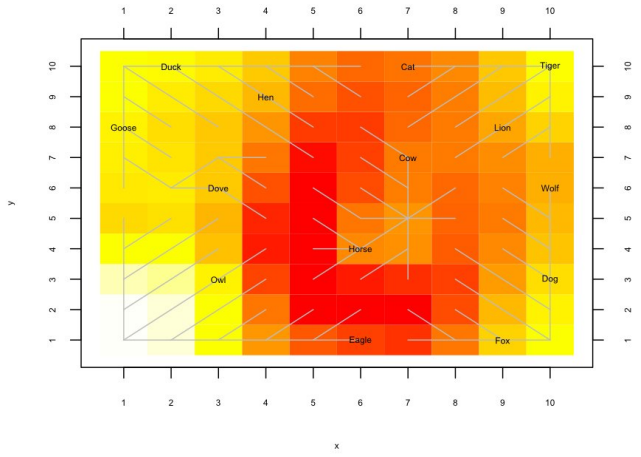

Fig. 3. Connected components of a SOM.

III. Connected Components

A. Development

Connected components are a graph theoretical concept and can be defined as follows,

> **Definition.** *A connected component of an undirected graph is a subgraph in which any two vertices are connected to each other by a path [6].*

In our case, we view the SOM neural elements as the vertices of an undirected planar graph. The edges in this graph are defined as follows based on the UMAT: a node in the graph is connected to a neighboring node along the maximum gradient on the UMAT. We do not add edges for nodes where the gradient is equal to zero. For the animal example given in the introduction this construction gives rise to a graph with five connected components shown in Figure 3.

It is easy to see that connectedness is a transitive relation. That is, if node A is connected to node B and node B is connected to node C, then node A is connected to node C via the path from A to B and from B to C. Exploiting this transitivity property and the fact that there exists a path from every node in a connected component to the node where the gradient is equal to zero, we can transform each component into a star [7] with the node where the gradient is equal to zero at the center. This transformation gives rise to the components as we had shown in Figure 2.

The visual power of this representation derives from the fact that clustering relationships between labeled neural elements can be directly read from the connected components by again exploiting the transitivity of connectedness. Consider for example the cluster in the lower left corner in the UMAT of Figure 1. It is visually difficult to see that owl and eagle indeed belong to the same cluster. Now compare this to the UMAT with the overlaid connected components shown in Figure 2. Here it is immediately evident that owl and eagle belong to the same cluster: owl and eagle are connected by a path in the connected component via the internal node of the star.

B. Implementation

We have implemented this enhanced version of the unified distance matrix as a package in R [8] (the code is available by request; we intend to release it as a publicly available package as part of CRAN). At the core of the implementation is the function `find.internal.node` which, given the map coordinates of a neural element, will find the corresponding internal node of the associated star. Table I shows the pseudo code for this function. Given a position on the map this function first searches the adjacent nodes for the minimal UMAT value using the function `find.min`. If an adjacent node with a smaller UMAT value than the value of our current node exists and if this UMAT value is smaller than the UMAT values of all other adjacent nodes, then that node lies along the maximum gradient of the surface and we make this node our new current position. If no such node exists, then the gradient at our current position is zero and we are at an internal node.

TABLE I

Pseudo code to find the internal node of a connected component.

```
function find.internal.node(int x,
                            int y,
                            real umat[xdim,ydim])

returns (int cx, int cy)

// x and y are the map coordinates of our current node
// umat is the unified distance matrix
// xdim and ydim are the dimensions of the map
// cx and cy are the map coordinates of the internal
// node of the star

begin
    // find the smallest value of umat in our immediate
    // neighborhood, including our current position.

    (minx,miny) = find.min(x,y,umat)

    // if minx and miny are our current position then
    // the gradient at our current position is zero
    // otherwise move to the new position along the
    // maximum gradient and call ourselves
    // recursively.

    if (minx == x and miny == y) then begin
        cx = minx
        cy = miny
        return
    end else begin
        (cx,cy) = find.internal.node(minx,miny,umat)
        return
    end
end
```

The remainder of the implementation is straightforward; we first draw the traditional UMAT, we then iterate over all the neural elements of the map and compute their corresponding internal nodes with the `find.internal.node` function. Once this is complete we draw the paths from each neural element to its corresponding internal node giving rise to enhanced UMAT representations as shown in Figure 2.

We found that smoothing the unified distance matrix before applying our algorithm to find the connected components results in larger, homogeneous components making the map easier to interpret. In our implementation we use a two-dimensional Gaussian kernel smoothing function where the bandwidth of the Gaussian controls the level of smoothing: the larger the bandwidth the more aggressive the smoothing. Picking the right level of smoothing then becomes a trade-off between the size of the connected components and their homogeneity.

IV. Experiments

We discuss four different experiments using our enhanced version of the UMAT. The first two experiments are based on artificial data sets from Ultsch's Fundamental Clustering Problem Suite (FCPS) [9] and the last two experiments are real world spectroscopy data sets we have analyzed using self-organizing maps. The hallmark of the latter data sets is that they are high-dimensional. Our petroleum data set has 257 attributes and our bacteria data set has 300 attributes. This high dimensionality forces us to consider large maps, in this case, maps with up to a thousand neural elements.

In each of these experiments we compare the original UMAT with our enhanced version of the UMAT and we point out that the enhanced version of the UMAT facilitates the discovery and evaluation of clusters. Unfortunately, as with many graphical artifacts, their comparative evaluation is highly subjective. However, it is our hope that this empirical study will convey some of the advantages of the enhanced UMAT representation, as we perceive them.

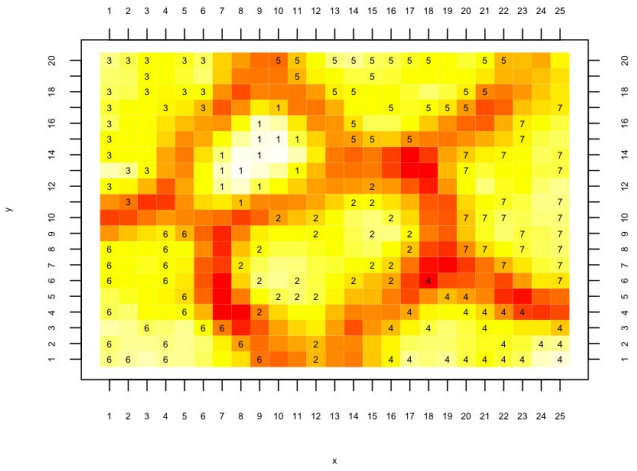

Fig. 4. UMAT for the Hepta data set.

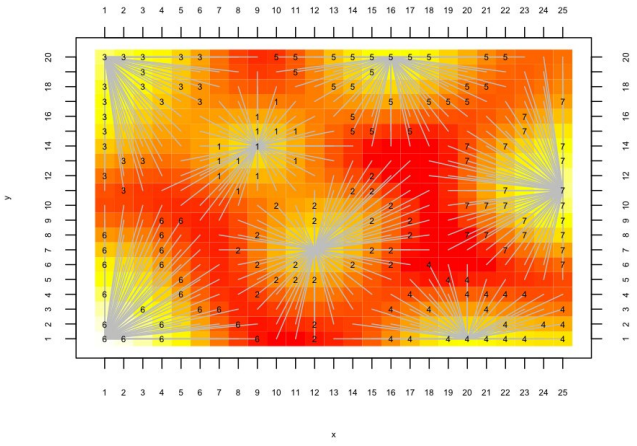

Fig. 5. Enhanced UMAT for the Hepta data set.

A. Experiment 1: Hepta

Hepta is the first FCPS data set. It is a three dimensional data set representing seven non-overlapping classes with different variances. The data set has 212 observations. Since the classes are non-overlapping we would expect that any clustering procedure, including SOM, would display seven individual clusters. Figure 4 shows the seven clusters in the traditional UMAT display and Figure 5 shows the clusters in our enhanced UMAT display. Since the seven clusters are completely separable, the traditional UMAT and our enhanced version show the clusters very nicely. The differences in the cluster layouts between UMAT and our enhanced version are due to the smoothing in our enhanced version.

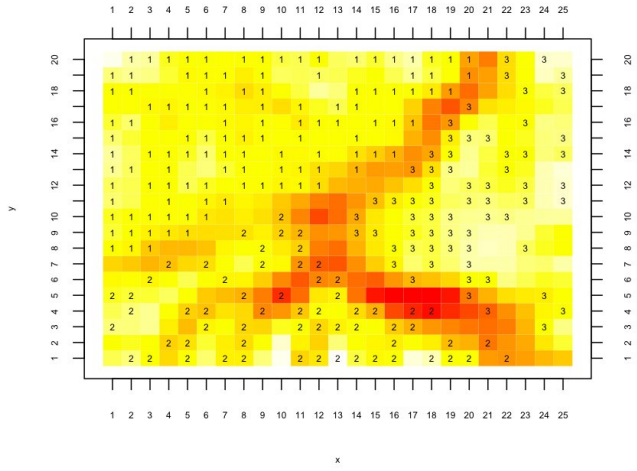

Fig. 6. UMAT for Lsun data set.

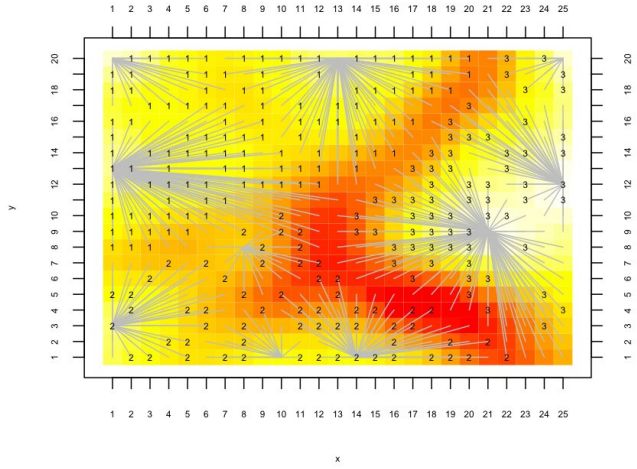

Fig. 7. Enhanced UMAT for the Lsun data set.

B. Experiment 2: Lsun

For our second experiment, we used the Lsun data set from Ultsch's problem suite. This is a two-dimensional data set with 400 observations containing three clusters. The hallmark of this data set is that the clusters have different variances and different inter-cluster distances. Figure 6 shows the UMAT for this data set. In this representation the border between

cluster 1 and cluster 2 in the bottom left corner of the map is difficult to detect due to the fact that they lie very close together in the data space. However, in our enhanced UMAT the border is easily seen due to the component structure. This can be seen in Figure 7 even though the clusters themselves are composed of multiple connected components.

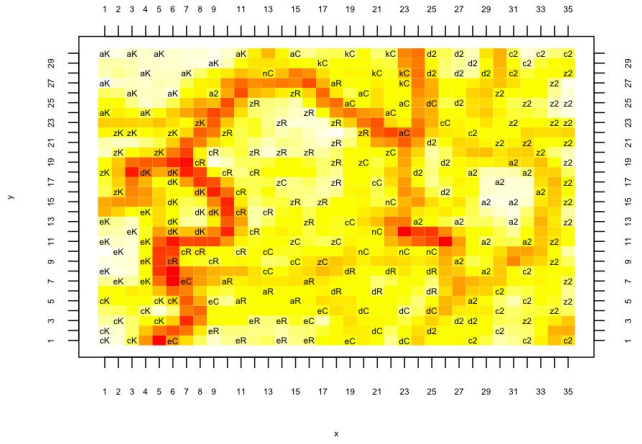

Fig. 8. UMAT for petroleum data set.

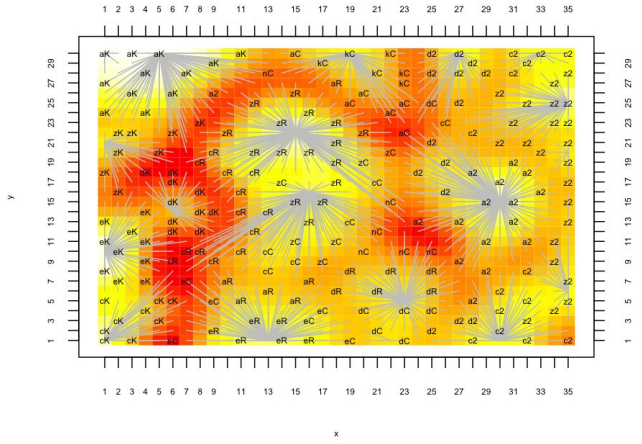

Fig. 9. Enhanced UMAT for the petroleum data set.

C. Experiment 3: Petroleum

Our petroleum data set consists of 235 observations of different petroleum products from various regions of the world. Each observation represents a spectrum in the infrared range and is labeled by a two letter label rP where r is a region identifier drawn from the set of region identifiers {a,c,d,e,k,n,z} and P is a product identifier: 2 – #2 fuel, K – kerosene, C – crude, and R – residue. Because of the large number of dimensions in the training data (257) we chose a map size with 1050 processing elements.

During our analysis we were interested to see if regions and petroleum products clustered based on the spectral information. Figure 8 shows the UMAT of our analysis. It turns out that there is significant clustering in this data set but it is difficult to see this here. Figure 9 shows the enhanced UMAT of the same data set. The clusters are easily visible. What this analysis shows is that both region of origin and fuel type carry significant identifying signatures in their corresponding spectra and this is easily seen in our enhanced UMAT.

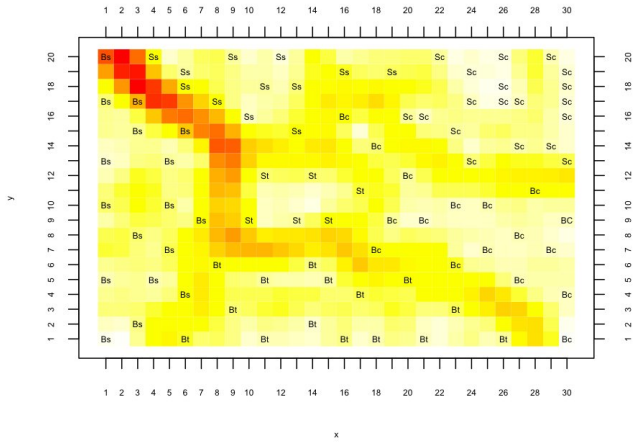

Fig. 10. UMAT for bacteria data set.

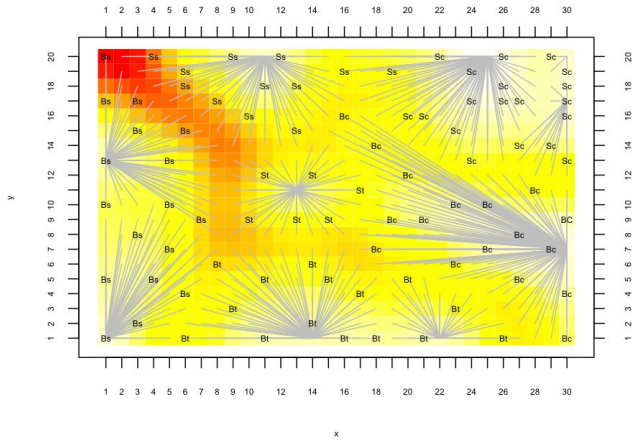

Fig. 11. Enhanced UMAT for the bacteria data set.

D. Experiment 4: Bacteria Spores vs. Vegetative States

Our final experiment concerns observations of three bacteria: Bacillus cereus, Bacillus thuringiensis, and Bacillus subtilis. For each bacterium we have spectral observations in its spore state as well in its vegetative state. Overall we have 89 observations. Each spectrum is composed of 300

wavelengths. The goal of this investigation was to see if the spectra of different bacteria are identifiable and if we can also distinguish between their spore and vegetative states. We labeled each observation with a two-letter label Qb where Q denotes the state of the bacterium, S for spore and B for the vegetative state, and b denotes the type of bacterium: c – cereus, t – thuringiensis, and s – subtilis. We analyzed the data with a 600 element SOM.

Figure 10 shows the traditional UMAT. Just as in the case of the petroleum data, the clusters are somewhat difficult to see even though the data clusters extremely well. Figure 11 is our enhanced UMAT and here the clusters are immediately visible due to the connected components. We are able to identify clusters of individual bacteria in the vegetative state along the bottom of the map and we can identify clusters of individual bacteria as spores in the top half of the map. This means that the spectra of the individual bacteria are characteristic with respect to whether they are spores or in the vegetative state as well as to their genus identity.

V. Related Work

The paper by Vesanto [10] provides a general overview of visualization techniques for self-organizing maps. Recently, a number of graph-based visualization techniques for self-organizing maps have been developed. From a visual perspective the work that is most closely related to our own is the work by Pölzlbauer *et. al* [11]. In their visualization they also plot gradient-based components on top of the SOM two-dimensional map, however, their gradients are derived from highly correlated groups of attributes in the training data. Thus, the meaning of the components is different from ours where a component simply connects all the neural elements that lie close together in data space. In [12] Tasdemir and Merenyi describe their graph-based visualization. In this visualization the graph edges overlaid on the SOM grid are color coded to convey more detailed information on the topology of the training data. What distinguishes our approach from these approaches is that we do not stray from the *de facto* standard interpretation of self-organizing maps via the unified distance matrix but instead enhance this interpretation.

VI. Conclusions and Further Work

We have shown that overlaying connected components based on the gradient information in the unified distance matrix improves the identification and interpretation of clusters on the self-organizing map. This was especially true in the case of our high-dimensional real world data.

We have found that smoothing the UMAT before applying our connected component identification algorithm results in less fragmented subgraphs. Given that the user can control the level of smoothing that is applied to the UMAT, constructing connected components then becomes a trade-off between the size of the components and their homogeneity. The more smoothing we apply, the larger the connected components, but this implies an increased likelihood that clusters will be merged that contain observations from different classes. Given that the connected components provide us with a tractable way to group observations into clusters, we envision that we can perform the optimization of map components via smoothing automatically. We can view this as a model fitting step. Highly fragmented components represent an overfit model responding to minor, perhaps random, nuances in the UMAT. Large, non-homogeneous components represent an underfit model; a model that is not refined enough to display the true clustering of the data. The optimization then is a model fitting step that attempts to find just the right trade-off between size and homogeneity of the components.

Furthermore we like to investigate why certain clusters consistently have multiple connected components regardless of the smoothing. We would like to understand if this is an artifact of the UMAT construction or if this in fact does describe topological features in the underlying data space.

References

[1] A. Ultsch, "Self-organizing neural networks for visualization and classification", *Proc. Conf. Soc. for Information and Classification*, 1993, pp. 307-313.

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

[3] K. Judge and C. W. Brown and L. Hamel, "Sensitivity of Raman Spectra to Chemical Functional Groups," *Appl. Spectrosc.*, Vol. 62, pp. 1221-1225, 2008.

[4] K. Judge and C. W. Brown and L. Hamel, "Sensitivity of Infrared Spectra to Chemical Functional Groups," *Anal. Chem.*, Vol. 80, pp. 4186-4192, 2008.

[5] A. Ultsch, "Data mining and knowledge discovery with emergent self-organizing feature maps for multivariate time series," *Kohonen Maps*, Elsevier Science, pp. 33-46, 1999.

[6] K. H. Rosen, *Discrete Mathematics and its Applications*, McGraw-Hill, New York, 2003.

[7] F. Harary, *Graph Theory*, Addison-Wesley, Reading, MA, 1994.

[8] R Development Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

[9] A. Ultsch, "Clustering with SOM: U*C", *Proc. Workshop on Self-Organizing Maps*, 2005, pp. 75-82.

[10] J. Vesanto, "SOM-based data visualization methods", it Intelligent Data Analysis, Vol. 3, pp. 111-126, 1999.

[11] G. Pölzlbauer and M. Dittenbach and A. Rauber, "Gradient visualization of grouped component planes on the SOM lattice", WSOM05, pp. 331-338, 2005.

[12] K. Tasdemir and E. Merenyi, "Exploiting data topology in visualization and clustering of self-organizing maps", *IEEE Trans. Neural Netw.*, Vol. 20, pp. 549-562, Apr. 2009.