Rule Visualization based on Multi-Dimensional Scaling

Thomas R. Gabriel, Kilian Thiel, and Michael R. Berthold

ALTANA Chair for Bioinformatics and Information Mining
Department of Computer and Information Science
Konstanz University, Box M 712, 78457 Konstanz, Germany
{gabriel,thiel,berthold}@inf.uni-konstanz.de

***Abstract*— This paper presents an approach to visualizing and exploring high-dimensional rules in two-dimensional views. The proposed method uses multi-dimensional scaling to place the rule centers and subsequently extends the rules' regions to depict their overlap. This results not only in a visualization of the rules' distribution but also enables the relationship to their immediate neighbors to be judged. The proposed technique is illustrated and discussed on a number of well-known benchmark data sets.**

I. Introduction

Extracting rules from data is not a new area of research. In [15] and [19], to name just two examples, algorithms were described that construct hyperrectangles in feature space. The resulting set of rules encapsulates regions in feature space that contain patterns of the same class. Other approaches, which construct fuzzy rules instead of crisp rules, were presented, for example, in [1], [12], [17] and [18].

What all of these approaches have in common is that they tend to build very complex rule systems for large data sets originating from a complicated underlying system. In addition, high-dimensional feature spaces result in even more complex rules relying on many attributes and increase the number of required rules to cover the solution space. An approach that aims to reduce the number of constraints on each individual rule was recently presented in [3]. The generated fuzzy rules only constrain a few of the available attributes and hence remain readable even in the case of high-dimensional spaces. However, this algorithm also tends to produce many rules for large, complicated data sets.

Multi-dimensional scaling methods [13] are widely used to map high-dimensional data onto lower dimensions. However, models—especially rule models—are much harder to visualize in lower dimensions. The main challenge is the mapping of the rules' antecedences, which are usually not only points but hyperrectangles in the original higher-dimensional feature space. One approach [5] that deals with this problem maps fuzzy points using fuzzy distance measures onto two dimensions using the MDS algorithm. Here, the rules' coverage is displayed by a square in two dimensions giving only a limited feeling for the original rule-spread in higher dimensions. Another idea visualizes fuzzy points as bands in Parallel Coordinates [4] for all features, but in the case of highly overlapping rules, exploration is hardly possible. Therefore, in order to obtain a useful view it is essential that the spread of rules is shown independently and in addition to their location, while the gap between adjacent rules is preserved in the original space. The method proposed here makes use of this information, and keeps track of the relationship between neighboring rules. The resulting two-dimensional view can then be used for further analysis.

The paper is organized as follows: in the next section we briefly introduce the used fuzzy rule induction algorithm[1]. We subsequently describe the multi-dimensional scaling method and how it is extended to visualize rule systems. We go on to illustrate the proposed method using the Iris data set and show how larger rule sets can be visualized and explored for a number of other benchmark data sets.

II. Fuzzy Rule Induction

The used fuzzy rule learning algorithm [3] constructs a set of fuzzy rules from given labeled training data and generates individual fuzzy rules defined by independent membership functions in the feature space. The training algorithm usually only constrains a few attributes, i. e., most support regions remain unconstrained, leaving the rules interpretable even in the case of high-dimensional input spaces. The final set of fuzzy rules can be described as an n-dimensional `IF` clause as antecedence and one assigned $class$ in the rule's conclusion:

$$\begin{array}{lllllll}
\mathcal{R}^1_1 & : \texttt{IF} & x_1 \texttt{ IS } \mu^1_{1,1} & \wedge\cdots\wedge & x_n \texttt{ IS } \mu^1_{n,1} & \texttt{THEN class } 1 \\
 & & \vdots & & \vdots & \\
\mathcal{R}^1_{r_1} & : \texttt{IF} & x_1 \texttt{ IS } \mu^1_{1,r_1} & \wedge\cdots\wedge & x_n \texttt{ IS } \mu^1_{n,r_1} & \texttt{THEN class } 1 \\
 & & \vdots & & \vdots & \\
\mathcal{R}^k_j & : \texttt{IF} & x_1 \texttt{ IS } \mu^k_{1,j} & \wedge\cdots\wedge & x_n \texttt{ IS } \mu^k_{n,j} & \texttt{THEN class } k \\
 & & \vdots & & \vdots & \\
\mathcal{R}^c_{r_c} & : \texttt{IF} & x_1 \texttt{ IS } \mu^c_{1,r_c} & \wedge\cdots\wedge & x_n \texttt{ IS } \mu^c_{n,r_c} & \texttt{THEN class } c
\end{array} \tag{1}$$

where $\mathcal{R}^k_j$ represents rule j for class k. The rule base contains rules for c classes and r_k indicates the number of rules for class k ($1 \leq j \leq r_k$ and $1 \leq k \leq c$). The fuzzy sets $\mu^k_{i,j} : \mathbb{R} \mapsto [0,1]$ are defined for every feature i ($1 \leq i \leq n$), but in cases of unconstrained features the membership degree is constant 1. The overall degree

[1]Note, however, that any algorithm generating rules from data can be used as a basis for the visualization method presented here.

of fulfillment of a specific rule for an input pattern $\vec{x} = (x_1, \ldots, x_n)$ can be computed using the fuzzy set operator for conjunction, called T-norm ($\top$):

$$\mu_j^k(\vec{x}) = \top_{i=1,\cdots,n} \left\{ \mu_{i,j}^k(x_i) \right\}. \quad (2)$$

The combined degree of membership for all rules of class k can be calculated using the fuzzy set operator for disjunction, called S-norm ($\perp$):

$$\mu^k(\vec{x}) = \perp_{j=1,\cdots,r_k} \left\{ \mu_j^k(\vec{x}) \right\}. \quad (3)$$

From these membership values the predicted class k_{best} for an input pattern $\vec{x}$ is derived as:

$$k_{\text{best}}(\vec{x}) = \arg\max_{k=1,\ldots,c} \left\{ \mu^k(\vec{x}) \right\}. \quad (4)$$

The algorithm uses trapezoidal membership functions, which can be described with four parameters $\langle a_i, b_i, c_i, d_i \rangle$, where a_i and d_i define the fuzzy rule's support-, and b_i and c_i its core-region for each attribute i of the input dimension. The core-region is defined as a rectangular area with an activation of 1, whereby the support-region decreases linearly to its boundaries with a degree of fulfillment of 0.

The fuzzy rule induction method is based on an iterative algorithm. During each learning epoch, i. e. presentation of all training patterns, new fuzzy rules are introduced when necessary and existing ones are adjusted whenever a conflict occurs. For each pattern three main steps are executed:

- **Cover**: If a new training pattern lies inside the support-region of an already existing fuzzy rule of the correct class, its core-region is extended to cover the new pattern. In addition, the weight of this rule is incremented.
- **Commit**: If the new pattern is not yet covered, a new fuzzy rule belonging to the corresponding class is created. The new example is assigned to its core-region, whereas the overall rule's support-region is initialized "infinite", that is, the new fuzzy rule is unconstrained on all features and covers the entire domain.
- **Shrink**: If a new pattern is incorrectly covered by an existing fuzzy rule of conflicting class, this fuzzy rule's core- and/or support-region is reduced, so that the conflict with the new pattern is avoided. The underlying heuristic of this step aims to minimize the loss in volume.

The algorithm usually terminates after only a few iterations over the training data. For details see [10].

In the next section, we show how the underlying multi-dimensional scaling method can be used to map the generated fuzzy rules' core regions onto two dimensions.

III. Multi-Dimensional Scaling

Multi-dimensional scaling methods [7] are widely used to map objects of high-dimensional spaces to lower dimensions, usually onto two or three dimensions. These methods try to preserve the pairwise distances between objects by minimizing an appropriate error function. The dimension reduction enables visualizations of high-dimensional points in lower-dimensional space. One technique to compute such a mapping is known as the Sammon algorithm [16] which finds a spatial representation $\vec{x}_i \in \mathbb{R}^L$ in the lower-dimensional space (usually $L = 2$) for each object $\vec{X}_i \in \mathbb{R}^H$ of the high-dimensional space ($1 \leq i \leq N$). The distances between two objects $\vec{X}_i, \vec{X}_j$ in the high- ($D_{ij} = D(\vec{X}_i, \vec{X}_j)$) and low-dimensional ($d_{ij} = d(\vec{x}_i, \vec{x}_j)$) space have to be approximated, that is:

$$\forall_{i \neq j} : D_{ij} \approx d_{ij},\ 1 \leq i, j \leq N \quad (5)$$

where the used distances are usually measured by the Euclidean distance:

$$D_{ij}^2 = \sum_{q=1}^{H} (X_{i,q} - X_{j,q})^2 \quad (6)$$

$$d_{ij}^2 = \sum_{k=1}^{L} (x_{i,k} - x_{j,k})^2. \quad (7)$$

To define a global cost function, the differences of the distance values need to be aggregated:

$$E = \sum_{i=1}^{N} \sum_{j>i}^{N} w_{ij} (d_{ij} - D_{ij})^2, \quad (8)$$

where the factors w_{ij} are introduced to weight the distances individually and to normalize the stress function E in order to be independent from the absolute values D_{ij}. Sammon suggests using an intermediate normalization [16]:

$$\omega_{ij} = \frac{1}{\sum_{i'=1}^{N} \sum_{j'>i'}^{N} D_{i'j'}} \frac{1}{D_{ij}}. \quad (9)$$

The algorithm then applies a steepest gradient method to iteratively minimize this stress function E for each point $\vec{x}_i$ at each step. The position for each point $\vec{x}_i$ in the lower-dimensional space is updated as follows:

$$\vec{x}_i^{(new)} = \vec{x}_i^{(old)} + \eta \vec{\Delta}_i \quad (10)$$

using the steepest gradient method to derive:

$$\forall k \in L : \Delta_{i,k} = -\frac{\partial E}{\partial x_{i,k}} \Big/ \left| \frac{\partial^2 E}{\partial x_{i,k}^2} \right|. \quad (11)$$

The update vector $\vec{\Delta}$ is computed based on the step length η (decreasing learning rate usually between 0.3 and 0.4) and is repeated for a given number of cycles. The algorithm usually converges after a few epochs to a (local) cost function minimum.

A. Mapping the Rule Centers

To map high-dimensional rules of the form given in (1), the center point $\vec{z}$ of the core-region's interval $[b, c]$ is computed as $\vec{z} = 1/2 \cdot (\vec{b} + \vec{c})$. The set of all centers $\vec{z}_i$ is then mapped onto two dimensions using the MDS method described above. For each high-dimensional point, a two-dimensional point is initialized either randomly, or by its first two principal components. The algorithm then re-positions the points in two dimensions to approximate the distances in the original space.

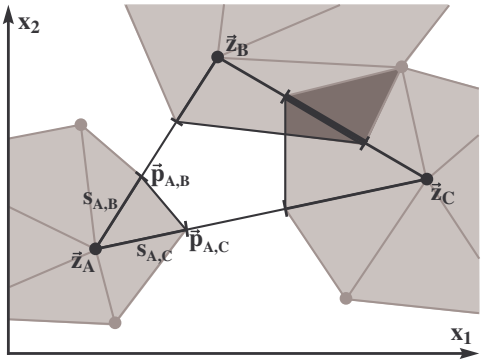

Fig. 1. Example with three rules, A, B, and C, mapped onto two dimensions. The figure shows the Delaunay triangulation between the rules' center points and the core coverage for rule A towards B and C. The points $\vec{p}_{A,B}$ and $\vec{p}_{A,C}$, as well as a number of virtual core boundary points are connected to sketch the rules' coverage areas, as well as overlap between rule B and C.

B. Visualizing the Rule Spreads

After the rule center points are positioned onto the lower-dimensional space, in addition, the rules' spread is mapped to provide better insight into the rule system by visualizing rule distributions and overlapping rule areas. For this, the axes parallel areas of coverage, i. e. hyperrectangles in high-dimensional feature space need to be placed onto lower dimensions. Note that we can not expect these mappings to also generate rectangles in the low-dimensional space. Based on the original rule coverage, the rule spread between the center of gravity and the border points is computed for all rules in the original space. These distances are then used to display an approximation of the rules' coverage in the target space, only considering rules in the immediate neighborhood. To find all direct neighbors the Delaunay triangulation [8] (geometric dual of the Voronoi diagram) is applied. In this graph each rule has a well-defined number of (at least two) neighbors, where each triplet of neighboring points spans a triangle in the two-dimensional space.

Figure 1 shows an example with three neighboring rules, A, B, and C, spanning such a triangle. Starting from rule A towards B and C, the spreads $s_{A,B}$ and $s_{A,C}$ are computed based on the Euclidean distance between the center and core boundary point in the original space. These values can be determined by the fraction between the distances in the higher- and lower-dimensional space. All boundary points compose the outer envelope and the resulting polygon illustrates the rule's spread.

In the special case of rule center points on the convex hull of the rule graph, the partial polygon is also mirrored to the opposite side of the hull assuming the same spread of the core regions towards all dimensions. This is necessary to equally model coverages to rules in the inner part of the hull and convey a better impression of the actual rule coverage.

More specifically, given two rules R_A and R_B, and their corresponding center points $\vec{Z}_A$ and $\vec{Z}_B$, and points on the core boundary $\vec{P}_A$ and $\vec{P}_B$ in the high-dimensional space, the spread $s_{A,B}$ is computed as:

$$s_{A,B} = d(\vec{z}_A, \vec{z}_B) \cdot D(\vec{Z}_A, \vec{P}_A)/D(\vec{Z}_A, \vec{Z}_B) \qquad (12)$$

using the distance between the center points $\vec{z}_A$ and $\vec{z}_B$ in the lower-dimensional space, and the fraction of the core spread in the original space normalized by the distance between the high-dimensional center points. The new point $\vec{p}_{A,B}$ on the core boundary of the lower-dimensional space can be determined as the offset of $\vec{z}_A$ towards $\vec{z}_B$:

$$\vec{p}_{A,B} = \vec{z}_A + (\vec{z}_B - \vec{z}_A)s_{A,B}. \qquad (13)$$

To compute all these boundary points between neighboring rules, three cases of rule arrangements need to be considered:

(1) A rule overlaps with another rule if all core regions overlap. Figure 2 shows two rules that overlap in all dimensions. Here, the distance to the core boundary point from the center of the rule under consideration is used to determine the overlapping value s.

(2) Rules which do not overlap in any dimension are non-overlapping, see Figure 2 center. The overlapping spread is computed as in (1).

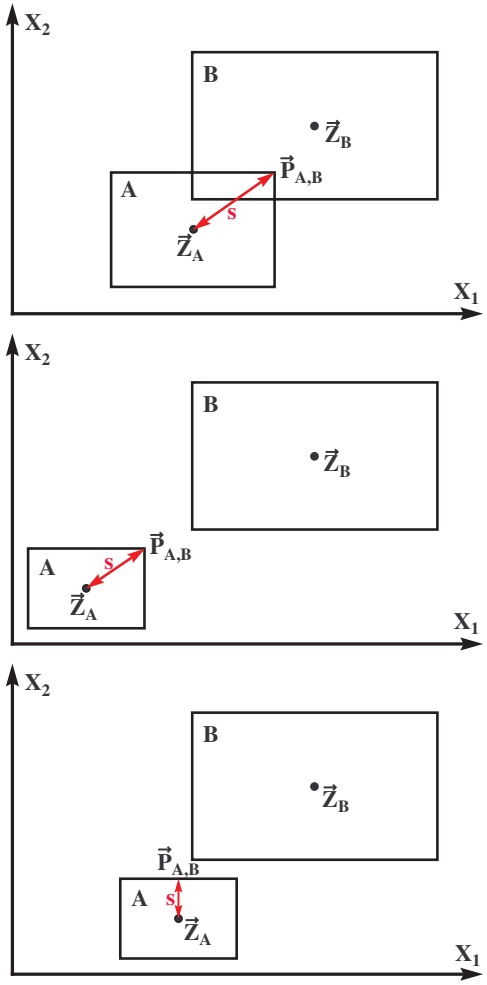

Fig. 2. Shows the three cases of overlapping (top), non-overlapping (center), and obstructing rules (bottom).

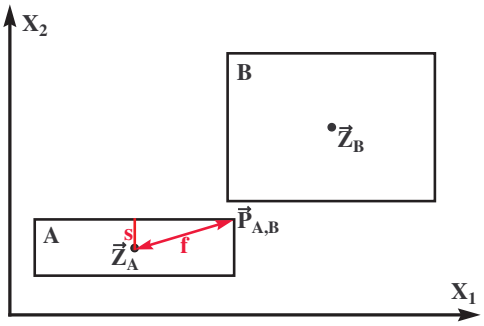

Fig. 3. Two unbalanced two-dimensional (X_1, X_2) rules only overlap in one dimension. The spread s is used to map rule R_A towards R_B, but f would be the better intuitive approximation.

(3) Obstructing rules may overlap either in one or more dimensions, but not in all dimensions. The spread s for obstructing rules is computed based on all non-overlapping dimensions, see Figure 2. All overlapping dimensions are ignored in this case and the distance functions in equation (12) need to be modified using only the non-overlapping dimensions.

This heuristic approach has one short-coming: in cases where the overlap is marginal in one dimension, this dimension is not taken into account when approximating the spread value. Figure 3 illustrates this problem, where f would be the better approximation, but s is computed as spread value due to the small overlap in the X_1 dimension.

C. Rule Visualization: An Example

Before showing results on real benchmarks we demonstrate the proposed visualization scheme on the well-known Iris data set [9]. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant: Iris Setosa (red), Iris Versicolor (green), and Iris Virginica (blue). The four dimensions consist of measurements for length and width of petal and sepal for each instance. One class (Iris Setosa) is linearly separable from the other two along axes-parallel lines in two of those dimensions.

The mapping of the rules' center points (with 14 rules generated by the fuzzy rule algorithm) is shown in Figure 4. In addition, for each rule the direct neighbors as determined by the Delaunay triangulation are connected by lines. It can already be seen that the red class is covered by only one rule whose center is relatively far away from all others. As expected, for the other two classes a higher number of rules was introduced. Still, this picture does not reveal any information about possible overlaps between rules or even a hint about the rules' spread.

To gain more insight into the rule model, it is desirable to see how each rule's spread can be displayed in relationship to their direct neighbors. Figure 5 illustrates how the proposed method displays this property for the rules generated for the Iris data. Here, we clearly see covered areas and rule overlapping. Also the red class of Iris Setosa is separated

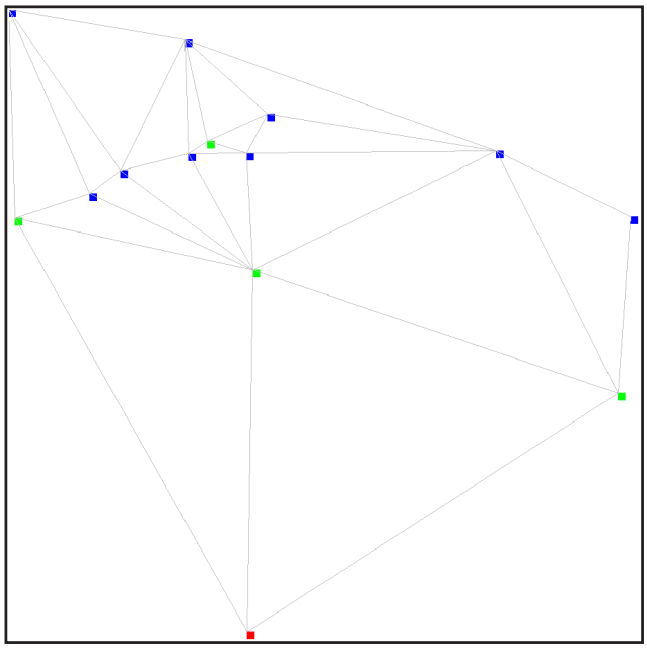

Fig. 4. Visualization of the rule set (center points only) generated for the Iris data. The Delaunay triangulation indicates direct neighbors for each rule.

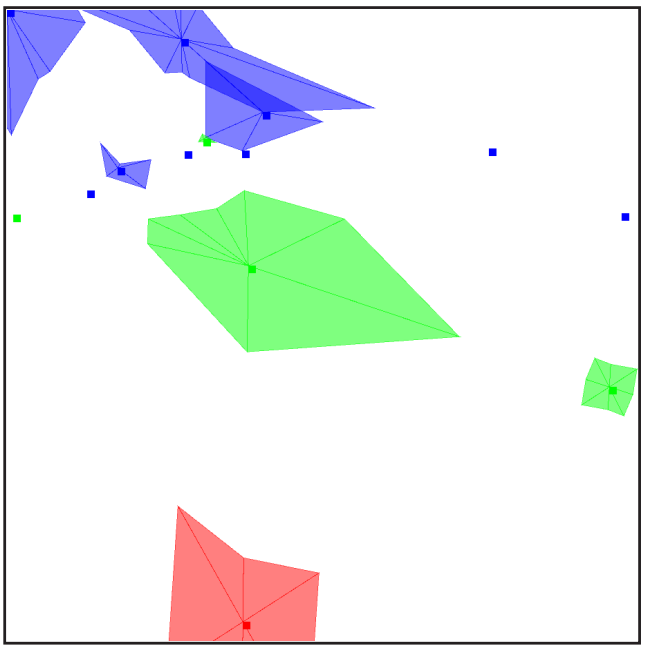

Fig. 5. Visualization of the Iris rule set, including its spread and possible overlaps.

relatively far apart from the other two. It becomes clearer from this view how the blue and green classes are modeled using rules that partially overlap and often only cover very small areas. The underlying algorithm had to commit highly specialized rules for these regions to model areas where patterns of these two classes are very close to each other.

Obviously, such an easy example is only suitable to

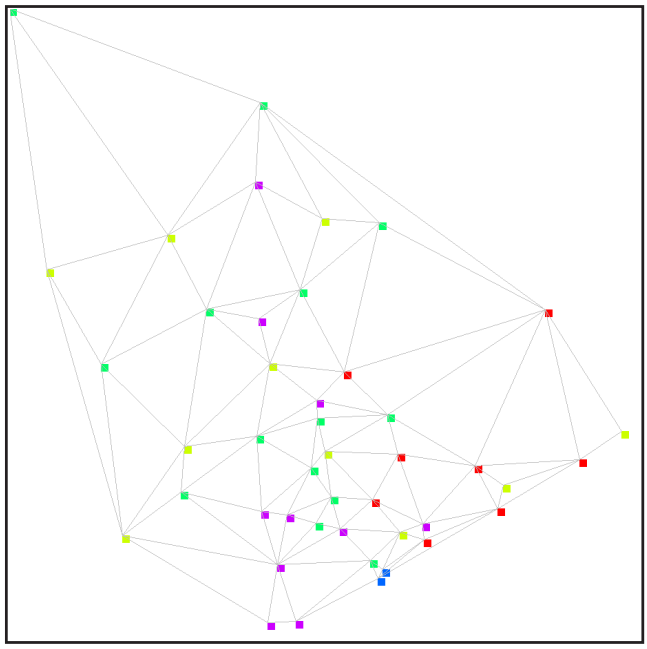

Fig. 6. Visualization of the rule set (center points only) generated on the Ocean Satellite Image data.

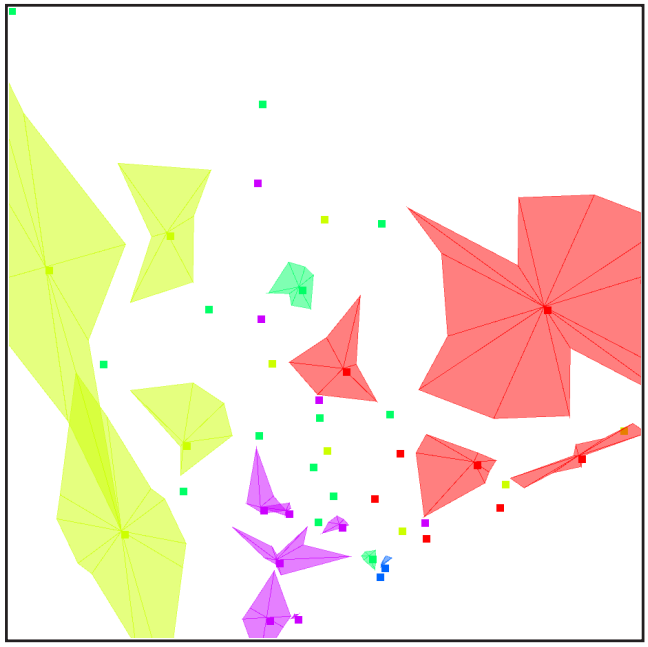

Fig. 7. Visualization of the rule set generated on the Ocean Satellite Image data, including its spread and possible overlaps.

demonstrate the methodology's operation. In the following section we show how this method works on two larger benchmark data sets.

IV. Experimental Results on other Data

Experiments were conducted on two benchmark data sets, namely Ocean Satellite Images [20], [22] and the Shuttle Control Database from the StatLog–Project [14].

A. Ocean Satellite Images

The first data set stems from satellite images from the Coastal Zone Color Scanner (CZCS) and are of the West Florida shelf. The CZCS was a scanning radiometer aboard the Nimbus-7 satellite, which viewed the ocean in six co-registered spectral bands 443, 520, 550, 670, 750 nm, and a thermal IR band. It operated from 1979-1986.

The features used were the 443, 520, 550, 670 nm bands; the pigment concentration value was derived from the lowest 3 bands. Atmospheric correction was applied to each image [11] before the features were extracted. A fast fuzzy clustering algorithm, mrFCM [6], was applied to obtain 12 clusters per image. There were five regions of interest in each image. These consist of red tide, green river, other phytoplankton blooms, case I (deep) water and case II (shallow) water. Twenty-five images were ground-truthed by oceanographers [21] and eighteen of these were used for training. The eighteen training images were clustered into 12 classes. Each class or cluster was labeled by the ground-truth image as its majority class.

The rules' center vectors generated from the training images are depicted in Figure 6. These 44 rules together with their core areas are shown in Figure 7. In this picture the low distortion of red and yellow rules becomes visible, e. g. these classes almost form one coherent cluster. All other classes are distributed across the two-dimensional domain.

The mapping of the rules gives more detailed insight into the structure of the rule system by showing larger inter-related areas. For instance, it is clearly visible that a number of rules of class yellow cover bigger areas of the space. Furthermore, a number of smaller rules, almost randomly distributed, only cover one or a small number of instances, which can be an indication for outliers or artifacts in the data. The larger areas of interest reflect the distribution of the input patterns. These rules can easily be identified and used for further investigation.

B. Shuttle Control Database

This data originated from NASA and concerns the position of radiators within the Space Shuttle [14]. The data was divided into a training set with 43,500 and a test set with 14,500 examples. The shuttle data has 9 attributes which are assigned to three classes[2]. This data set can be perfectly described with only 15 rules, see Figures 8 and 9.

In this example the relationships between rules and their organization can be seen clearly. For example, rules for class blue are split due to intermediate instances of conflicting classes. Other rules cover bigger areas and are separated from the remaining rules. A number of smaller rules, red and green, can be found in the middle of the plot, which again are caused by areas with low evidence in the data, but which may, for some application, be exactly what the user is interested in.

[2]The remaining classes of the original data set occur less than 1% and were left out for these experiments.

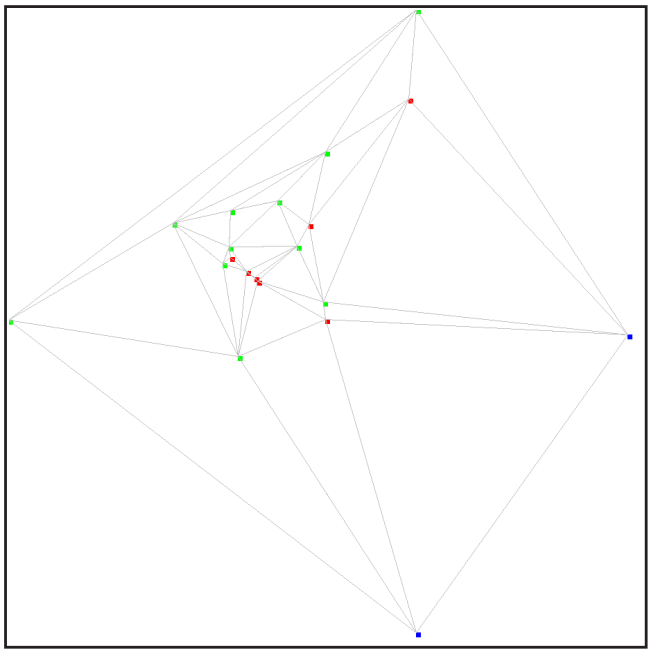

Fig. 8. Visualization of the rule set (center points only) generated on the Shuttle Control Database.

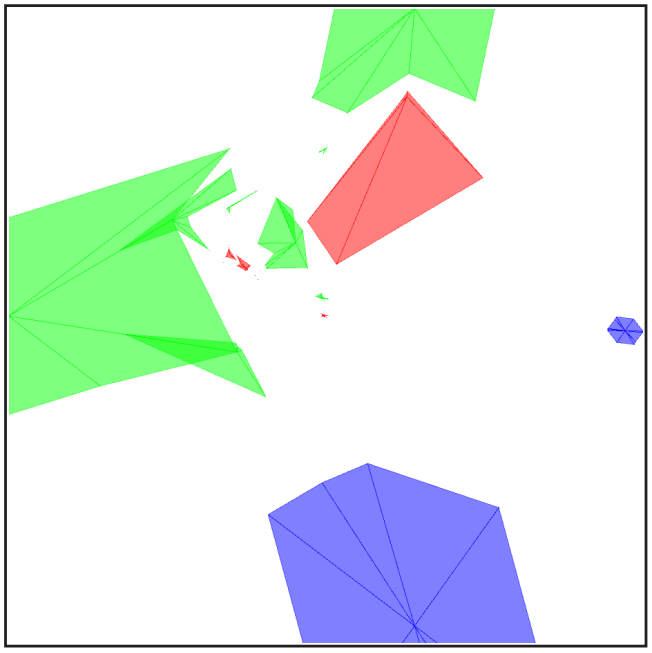

Fig. 9. Visualization of the rule set generated on the Shuttle Control Database, including its spread and possible overlaps.

V. Conclusions

In this paper a methodology was presented to visualize a set of rules built in a high-dimensional feature space. The mapping onto the two-dimensional space maintains the pairwise distances between the rules as much as possible and additionally displays an approximation of the rules' spread and overlap in the original feature space. The presented methodology not only provides a way to visualize the model but also shows potential for user interaction with the model and hence offers a promising addition to intelligent data analysis [2] in the area of model exploration.

Acknowledgments

This work was partially supported by the DFG Research Training Group GK-1042 "Explorative Analysis and Visualization of Large Information Spaces".

References

[1] S. Abe and M.-S. Lan. A method for fuzzy rules extraction directly from numerical data and its application to pattern classification. *IEEE Transactions on Fuzzy Systems*, 3(1):18–28, 1995.

[2] M. R. Berthold and D. J. Hand, editors. *Intelligent Data Analysis: An Introduction*. Springer Verlag, 2003.

[3] M. R. Berthold. Mixed fuzzy rule formation. *International Journal of Approximate Reasoning (IJAR)*, 32:67–84, 2003.

[4] M. R. Berthold and L. O. Hall. Visualizing Fuzzy Points in Parallel Coordinates. *IEEE Transactions on Fuzzy Systems*, 11:369–374, 2003.

[5] M. R. Berthold and R. Holve. Visualizing high dimensional fuzzy rules. In *Proceedings of NAFIPS*, pages 64–68. IEEE Press, 2000.

[6] T. W. Cheng, D. B. Goldgof, and L. O. Hall. Fast fuzzy clustering. *Fuzzy Sets and Systems*, 93:49–56, 1998.

[7] T. F. Cox and M. A. Cox. Multidimensional Scaling. *Monographs on Statistics and Applied Probability*, Chapman and Hall, 1994.

[8] B. Delaunay. Sur la sphere vide. *Bulletin of Academy of Sciences of the USSR*, pages 793–800, 1934.

[9] R. A. Fisher. The use of multiple measurements in taxonomic problems. In *Annual Eugenics, II*, 7, pages 179–188. John Wiley, NY, 1950.

[10] T. R. Gabriel and M. R. Berthold. Influence of fuzzy norms and other heuristics on "mixed fuzzy rule formation". *International Journal of Approximate Reasoning (IJAR)*, 35:195–202, 2004.

[11] H. R. Gordon, D. K. Clark, J. L. Mueller, and W. A. Hovis. Phytoplankton pigments derived from the nimbus-7 czcs: comparisons with surface measurements. *Science*, 210:63–66, 1980.

[12] C. M. Higgins and R. M. Goodman. Learning fuzzy rule-based neural networks for control. In *Advances in Neural Information Processing Systems*, 5, pages 350–357, Morgan Kaufmann, California, 1993.

[13] J. Meulman. *A distance approach to nonlinear multivariate analysis*. DSWO Press, Leiden, The Netherlands, 1986.

[14] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, editors. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood Limited, 1994.

[15] S. Salzberg. A nearest hyperrectangle learning method. In *Machine Learning*, 6, pages 251–276, 1991.

[16] J. W. Sammon, Jr. A non-linear mapping for data structure analysis. *IEEE Transactions on Computers*, C18 (5):401–409, May 1969.

[17] P. K. Simpson. Fuzzy min-max neural networks – part 1: Classification. *IEEE Transactions on Neural Networks*, 3(5):776–786, Sept. 1992.

[18] L.-X. Wang and J. M. Mendel. Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(6):1313–1427, 1992.

[19] D. Wettschereck. A hybrid nearest-neighbour and nearest-hyperrectangle learning algorithm. In *Proceedings of the European Conference on Machine Learning*, pages 323–335, 1994.

[20] M. Zhang, L. O. Hall, and D. Goldgof. Knowledge-based classification of czcs images and monitoring of red tides off the west florida shelf. In *The* 13^{th} *International Conference on Pattern Recognition*, volume B, pages 452–456, 1996.

[21] M. Zhang, L. O. Hall, F. E. Muller-Karger, and D. B. Goldgof. Knowledge guided classification of coastal zone color images off the west florida shelf. *International Journal of Pattern Recognition and Artificial Intelligence*, V. 14, No. 8, pp. 987–1007, 2000.

[22] M. Zhang, L. O. Hall, D. B. Goldgof, and F. E. Muller-Karger. Fuzzy analysis of satellite images to find phytoplankton blooms. In *IEEE International Conference on Systems Man and Cybernetics*, 1997.