A Three-Stage Approach to Descriptive Data Analysis — Identifying Clusters and Corresponding Interpretable Descriptions from Large Data Sets†

Mario Drobics, Ulrich Bodenhofer, Markus Mittendorfer-Holzer

Software Competence Center Hagenberg
A-4232 Hagenberg, Austria
{mario.drobics|ulrich.bodenhofer|markus.mittendorfer}@scch.at

Abstract

This paper presents a three-stage approach to data mining which puts special emphasis on the visualization and interpretability of the results. In the first stage, the input data is represented by a self-organizing map in order to allow visualization and to reduce the amount of data while removing noise, outliers, and missing values. Then this preprocessed information is used to identify and display fuzzy clusters of similarity. Finally, descriptions close to natural language are computed for these clusters in order to provide the analyst with qualitative information. This is accomplished by generating fuzzy rules using an inductive learning method. The proposed approach is applied to image segmentation and labeling.

Categories and Subject Descriptors (according to ACM CCS): I.5.3 [Pattern Recognition]: Clustering

1. Introduction

This paper addresses the tasks of extracting, displaying, and describing previously unknown clusters of similarity in large data sets. Special emphasis is placed on the robustness and interpretability of the results, with the aim to support decision makers and domain experts who want to make use of this information; therefore, our investigations are not only focused on the accuracy of the clusters, but also on their characterization by means of interpretable, hence user-friendly, knowledge. In order to demonstrate this substantial need, let us consider the following example: a typical application of clustering is market segmentation, i.e. the identification of significant groups of customers from data (e.g. information about customer, transactional data, order history). A salesman who is not an expert in data analysis needs to have a compact and interpretable description of the customer groups (clusters) in order to be able to take this information in his/her daily practice into account.

This contribution is devoted to a novel three-stage approach to data mining, which not only extracts clusters of interest from large data sets, but also generates *qualitative descriptions*.

The first step consists of reducing the data to a reasonable amount using a self-organizing map. Such a map still contains all significant information while eliminating possible data faults like noise, outliers, or missing values. While, for the purpose of data reduction, other methods may be used as well, SOMs have the advantages that they preserve the topology of the data space and allow to visualize the results [5], which is a particularly important aspect according to our requirement that results should be easy to evaluate and interpret also by non-experts.

The second step is concerned with identifying significant regions of interest within the SOM nodes using a modified fuzzy c-means algorithm [2, 9]. In our variant, two main drawbacks of the standard fuzzy c-means algorithm are resolved in a very simple and elegant way. Firstly, the clustering is not influenced by outliers anymore, as they have been eliminated by the SOM. Secondly, the problem of finding appropriate initial values for the cluster centers is solved by computing a crisp Ward clustering [20] first.

In the third and last stage, we create linguistic descriptions of the centers of these clusters which help the analyst to interpret the results. As the number of data samples under consideration has been reduced tremendously by using the SOM nodes, we are able to apply inductive learning methods [13] to find fuzzy descriptions of the clusters. Using the previously found clusters, we can make use of supervised learning methods in an unsupervised environment by considering the cluster membership as goal parameter. Descriptions are composed using fuzzy predicates of the form "*x is/is not/is at least/is at most A*", where *x* is the parameter under consideration, and *A* is a linguistic expression modeled by a fuzzy set. The method used here is FS-FOIL [7], a fuzzy variant of Quinlan's First Order Inductive Learner (FOIL) [16, 17].

† This paper is a shortened and updated version of [8].

In all the three stages, it is possible to take available domain knowledge into account. This can be done by assigning an importance weight to each attribute in the preprocessing and clustering steps or by defining domain-specific predicates in the final description step.

The proposed method is applied to several real-world data sets. Through the combination of the two-dimensional projection of the data by means of a SOM and the fuzzy descriptions generated, the results are not only significant and accurate, but also very intuitive and easy to interpret.

2. Preprocessing

With the aim to reduce computational effort in the consecutive steps without losing significant information, we create a mapping of the data space by means of a *self-organizing map* (SOM) [10].

Let us assume that we are given a data set I consisting of K samples from a data space $\mathcal{R}$. Throughout the remaining paper, we restrict to the n-dimensional real case $\mathcal{R} = \mathbb{R}^n$, i.e.

$$I = \{\vec{x}^1, \ldots, \vec{x}^K\} \subset \mathbb{R}^n.$$

We refer to the r-th dimension ($r = 1, \ldots, n$) as *r-th input variable* or *r-th attribute*.

A SOM is defined as a mapping from the data space $\mathcal{R}$ to a discrete output space $\mathcal{C}$. The output space consists of a finite number $N_\mathcal{C}$ of nodes ("neurons"). By enumerating the nodes, we are able to identify each node in $\mathcal{C}$ with a unique index $i \in \{1, \ldots, N_\mathcal{C}\}$. On the set of nodes $\mathcal{C}$, we assume a certain topological structure (i.e. a neighborhood relation) and a distance function $d_\mathcal{C}(.,.)$. In this paper, we will suffice with planar hexagonal grids; in order to define the distance function $d_\mathcal{C}(.,.)$, we assign two-dimensional coordinates to each node according to the planar hexagonal grid and apply the standard Euclidean norm. To each node $i \in \{1, \ldots, N_\mathcal{C}\}$, we associate a *weight vector* in the data space $\vec{w}^i = (w_1^i, w_2^i, \ldots, w_n^i)^T \in \mathcal{R}$. The mapping

$$\begin{aligned} \phi : \mathcal{R} &\longrightarrow \mathcal{C} \\ \vec{x} &\longmapsto \phi(\vec{x}) = \text{argmin}_i \{\|\vec{x} - \vec{w}^i\|_V\} \end{aligned}$$

defines the *"neural" map* of the data space $\mathcal{R}$ in the topological structure $\mathcal{C}$, where

$$\|\vec{x}\|_V^2 = \frac{\sum_{r=1}^n \frac{m_r}{V_r} x_r^2}{\sum_{r=1}^n m_r}.$$

In the norm $\|.\|_V$, m_r denotes an individual weight parameter which allows us to take different importance of the attributes into account. For the r-th attribute, the factor V_r stands for the variance of the values $(x_r^1, \ldots, x_r^K)$. In other words, the function ϕ maps every input sample $\vec{x}$ to that node of the map whose weight vector is closest to $\vec{x}$ with respect to the norm $\|.\|_V$. Obviously, this mapping creates a partition of the data space. Each part is a convex polytope and contains one unique weight vector $\vec{w}^i$ which is called its *nucleus*. This partition of the input space into convex polytopes is usually called *Voronoi tessellation*.

Appropriate weight vectors for a given topological structure $\mathcal{C}$ and a data set I have to be found in a *training process*. The aim of this process is to fit the data by the weight vectors as close as possible. More specifically, the goal is to minimize the *distortion error*

$$\sum_{j=1}^{K} \sum_{i=1}^{N_\mathcal{C}} \frac{h(i, \phi(\vec{x}^j))}{K} \cdot \|\vec{x}^j - \vec{w}^i\|_V,$$

where $h(i,k)$ defines the lateral interaction in $\mathcal{C}$ relative to the distance $d_\mathcal{C}(i,k)$. We use a Gaussian bell

$$h(i,k) = \exp\Big(-\frac{d_\mathcal{C}(i,k)^2}{2 \cdot r(t)^2}\Big)$$

with radius $r(t) \geq 1$ decreasing from $\frac{\sqrt{N_\mathcal{C}}}{4}$ to 1 over training time t.

Algorithm 1 (SOM Training)

Input: data samples $I = \{\vec{x}^1, \ldots, \vec{x}^K\}$
Output: weight vectors $W = \{\vec{w}^1, \ldots, \vec{w}^{N_\mathcal{C}}\}$

$t = 0$
Initialize node weights $\vec{w}^1, \ldots, \vec{w}^{N_\mathcal{C}}$ (e.g. randomly)
do {
  Select a random sample $\vec{x}^j$ with $1 \leq j \leq K$
  for $i = 1, \ldots, N_\mathcal{C}$
    Update weight vector of node i:
    $\vec{w}^i = \vec{w}^i + \lambda(t) \cdot h(i, \phi(\vec{x}^j)) \cdot (\vec{x}^j - \vec{w}^i)$
  $t = t + 1$
} while $\sum_{i=1}^{N_\mathcal{C}} \Delta\vec{w}^i > \varepsilon$;

In the update formula for the weight vectors, $\lambda(t)$ denotes the so-called *learning factor*, which decreases over time

$$\lambda(t) = \frac{1}{2} \cdot \exp\Big(-\frac{t^2}{2 \cdot (\sigma \cdot K)^2}\Big).$$

In our applications, we sufficed with a fixed value $\sigma = 20$, which ensures that a few ten-thousand learning steps are performed. For data sets with much redundant information, this coefficient may be decreased, while, for very complex data sets, a larger value can be beneficial [11].

SOM training falls into the category of *unsupervised competitive learning paradigms*, since the data samples are assigned to nodes in a self-organizing process without explicit goal information. The crucial difference between SOMs and conventional competitive learning is that not only the weight vector of the "winning node" $\phi(\vec{x}^j)$ is updated. Instead, by taking the lateral interaction $h(i, \phi(\vec{x}^j))$ into account, also the weight vectors of neighboring nodes are modified. Therefore, it is possible to achieve that the weight vectors of nodes which are close to each other in the topological structure $\mathcal{C}$ are also close in the data space $\mathcal{R}$. In other words, the topological structure of the data set I is preserved by the SOM (see [18] for a more detailed argumentation).

Since the topological structure $\mathcal{C}$ is typically chosen such that it can be projected to a two-dimensional plane (like the planar hexagonal grid we use), it is possible to plot each attribute with respect to the map in a very straightforward manner. This graphical information can be used to analyze dependencies in the data in a visual manner [5].

In our three-stage approach, the purpose of the SOM preprocessing is twofold. On the one hand, since the weight vector of each node approximates the mean of several data records, outliers and small variations in the input set are eliminated and noise is drastically reduced. On the other hand, the amount of data is heavily reduced, while the weight

vectors still contain all significant information. The two following steps, therefore, are able to work with a clean and compact set of data, which helps to improve their efficiency and robustness.

3. Clustering

The second building block of our approach is the identification of possible regions of interest in the map. Regions of interest are characterized through nodes, the weight vectors of which are close to each other in the data space. To find these regions of interest, we use a modified *fuzzy c-means* clustering algorithm [2]. The aim is not only to find clusters which are homogeneous within the data space, but also within the map. This is accomplished by taking the topological information of the map into account, too. The number of regions of interest L is fixed a priori.

In order to find appropriate initial values for the fuzzy c-means algorithm, we first perform a crisp *Ward clustering* [20]. Ward clustering is an agglomerative clustering method, which is often not applicable due to its high computational effort. This problem is avoided, as we use the compressed node information. The homogeneity of the clusters within the map is achieved by merging only nodes/clusters which are neighbors in the map (according to the topological structure, e.g. a hexagonal grid). After this initialization process, fuzzy clustering is done by applying the usual iterative two-step update procedure.

Algorithm 2 (Fuzzy c-Means)

Input: number of clusters L,
weight vectors $W = \{\vec{w}^1, \ldots, \vec{w}^{N_C}\}$
Output: cluster centers $V = \{\vec{v}^1, \ldots, \vec{v}^L\}$

Initialize cluster centers $\vec{v}^1, \ldots, \vec{v}^L$ using Ward clustering
do {
  for $i = 1, \ldots, N_C$
    for $l = 1, \ldots, L$
      Compute degree of membership $C(l,i)$ of $\vec{w}^i$ to cluster l
  for $l = 1, \ldots, L$
    Compute new cluster center $\vec{v}^l$
} while $\sum_{l=1}^{L} \Delta\vec{v}^l > \varepsilon$;

The degree of membership to which node i belongs to cluster l, denoted $C(l,i)$, is computed as

$$a(l,i) = \exp\big(-\alpha \cdot d^2(\vec{w}^i, \vec{v}^l)\big) \tag{1}$$

$$C(l,i) = \frac{a(l,i)}{\sum_{k=1}^{L} a(k,i)} \tag{2}$$

The factor $\alpha \geq 1$ determines the fuzziness of the clusters (1 corresponds to rather blurred clusters, while, e.g., a value of 100 gives very sharp boundaries between clusters; we use $\alpha = 40$). By normalizing the sum of class memberships to 1 for each node (cf. (2)), we create a fuzzy partition of $\mathcal{C}$ (in the sense of [19]). Taking the exponential function to compute the membership degree overcomes the problem of non-convexity [9].

In Eq. (1), the distance measure $d(.,.)$ is defined such that the weights and the topological structure of the map are taken into account, i.e.

$$d_R^2(\vec{x},\vec{y}) = \frac{1}{\sum_{r=1}^{n} m_r} \cdot \sum_{r=1}^{n} m_r \cdot \Big(\frac{|x_r - y_r|}{\max_{k,l} |w_r^k - w_r^l|}\Big)^2$$

$$d^2(\vec{x},\vec{y}) = d_R^2(\vec{x},\vec{y})^{h(\phi(\vec{x}),\phi(\vec{y}))^\beta}$$

For the radius of the lateral interaction $h(.,.)$ we use $r = \sqrt{N_C}$. Note that, due to the division by $\max_{k,l} |w_r^k - w_r^l|$, $d_R(\vec{x},\vec{y})$ is always a value between 0 and 1 if all values x_r, y_r are in the interval $[\min_k w_r^k, \max_k w_r^k]$. Since we apply $d_R(.,.)$ only to weight vectors or weighted sums of them, this requirement holds throughout the clustering process. The exponent

$$h(\phi(\vec{x}),\phi(\vec{y}))^\beta$$

corresponds to the influence of the neighborhood relation $h(.,.)$; the smaller $h(\phi(\vec{x}),\phi(\vec{y}))$ is (i.e. the larger the distance of node $\phi(\vec{x})$ and node $\phi(\vec{y})$ in the map is), the stronger the raw distance $d_R(\vec{x},\vec{y})$ is stretched, while the exponent β controls how strong the influence of the topological structure is (we use $\beta = 2$).

In the second step of the main loop, the cluster centers $\vec{v}^l$ are updated by computing the weighted means of all nodes [9]:

$$\vec{v}^l = \frac{1}{\sum_{i=1}^{N_C} C(l,i)^2} \cdot \sum_{i=1}^{N_C} C(l,i)^2 \cdot \vec{w}^i$$

The complete procedure is repeated until the change of the cluster centers is lower than a predefined limit ε. The result is a set of cluster centers which uniquely characterize regions of interest in the data set.

4. Cluster Descriptions

The final stage is concerned with finding interpretable, simple, and unambiguous descriptions of the regions of interest determined in the clustering step.

In order to create descriptions in a logical manner, it is necessary to define a discrete set of predicates for each attribute. In case of continuously valued numerical attributes, discretization cannot be avoided. If this discretization is done by means of partitions into crisp sets (intervals), small variations (e.g. noise) can cause large changes in the classification quality and mislead the search. This entails the demand for admitting vagueness in the assignment of samples to predicates. Beside rough set theory [15], fuzzy sets [21] have emerged as a standard models for solving this problem of artificial preciseness arising from sharp interval boundaries.

We create a set of fuzzy predicates for each attribute. Depending on the underlying context of the attribute under consideration, we assign natural language expressions like *very very low*, *medium*, *large*, etc. to each attribute. These expressions are modeled by fuzzy sets which can either be defined by hand or automatically. When defining the fuzzy sets, in particular when they are generated automatically, it is important to preserve the semantic content of the natural language expressions (e.g. that *low* corresponds to smaller values than *high*) in order to achieve interpretable descriptions [3].

To find appropriate fuzzy segmentations of the input domains according to the distribution of the sample data automatically, we use a simple clustering-based algorithm. The cluster centers are first initialized by dividing the input range into a certain number of equally sized intervals (the number

is denoted by s_r for the r-th attribute). Then these centers are iteratively updated using a modified *k-means* algorithm [12] with simple neighborhood interaction to preserve the ordering of the clusters. Usually, a few iterations (less than ten) are sufficient for each input variable. The fuzzy sets are then created as s_r trapezoidal membership functions around the centers. For more details, see [6].

Assume that the j-th fuzzy set for the r-th attribute ($r = 1,\ldots,n$ and $j = 1,\ldots,s_r$) is denoted with M_{rj}. Then M_{rj}, uniquely characterized by its membership function $\mu_{M_{rj}} : \mathbf{R} \to [0,1]$, induces a fuzzy predicate (for $x_r \in \mathbf{R}$)

$$t(x_r \text{ is } M_{rj}) = \mu_{M_{rj}}(x_r),$$

where t is the function assigning a truth value to the assertion "x_r is M_{rj}". In a straightforward way, a fuzzy set M_{rj} also induces the following fuzzy predicates [4]:

$$\begin{aligned} t(x_r \text{ is not } M_{rj}) &= 1 - \mu_{M_{rj}}(x_r) \\ t(x_r \text{ is at least } M_{rj}) &= \sup\{\mu_{M_{rj}}(y) \mid y \le x_r\} \\ t(x_r \text{ is at most } M_{rj}) &= \sup\{\mu_{M_{rj}}(y) \mid y \ge x_r\} \end{aligned}$$

We refer to these four kinds of predicates as *atoms* or *atomic predicates* in the following. Without loss of generality, we consider all atoms as predicates on $\mathcal{R}$:

$$\begin{aligned} t(\vec{x} \text{ is } M_{rj}) &= t(x_r \text{ is } M_{rj}) \\ t(\vec{x} \text{ is not } M_{rj}) &= t(x_r \text{ is not } M_{rj}) \\ t(\vec{x} \text{ is at least } M_{rj}) &= t(x_r \text{ is at least } M_{rj}) \\ t(\vec{x} \text{ is at most } M_{rj}) &= t(x_r \text{ is at most } M_{rj}) \end{aligned}$$

Let us assume that we want to compute a description of the l-th cluster obtained in the previous step. We denote the predicate corresponding to the membership to cluster l as (for convenience, using prefix notation)

$$t(C^l(\vec{x})) = C(l, \phi(\vec{x})).$$

Now the goal is to find an appropriate approximation of C^l by means of a fuzzy predicate consisting of conjunctions/disjunctions of atomic predicates. Logical operations on the truth values are defined using triangular norms and conorms, i.e. commutative, associative, and non-decreasing binary operations on the unit interval with neutral elements 1 and 0, respectively. In our applications, we restricted to the Łukasiewicz operations:

$$\begin{aligned} T_{\mathbf{L}}(x,y) &= \max(x+y-1, 0) \\ S_{\mathbf{L}}(x,y) &= \min(x+y, 1) \end{aligned}$$

Then conjunctions and disjunctions of arbitrary fuzzy predicates A and B can be defined as follows:

$$\begin{aligned} t(A(\vec{x}) \wedge B(\vec{x})) &= T_{\mathbf{L}}\big(t(A(\vec{x})), t(B(\vec{x}))\big) \\ t(A(\vec{x}) \vee B(\vec{x})) &= S_{\mathbf{L}}\big(t(A(\vec{x})), t(B(\vec{x}))\big) \end{aligned}$$

The *degree of common fulfillment* of a predicate A and the goal predicate C^l (for a given sample $\vec{x}$) is defined as $t(A(\vec{x}) \wedge C^l(\vec{x}))$. For a given finite sample set X, the fuzzy set of samples fulfilling a predicate A, which we denote with $A(X)$, is defined as (for all $\vec{x} \in X$)

$$\mu_{A(X)}(\vec{x}) = t(A(\vec{x})).$$

The *cardinality* of a fuzzy set N on X is defined as the sum of $\mu_N(\vec{x})$, i.e.

$$|N| = \sum_{\vec{x} \in X} \mu_N(\vec{x}).$$

Finally, the cardinality of samples commonly fulfilling a predicate A and the goal predicate C^l can be defined by

$$\begin{aligned} |A(X) \cap C^l(X)| &= \sum_{\vec{x} \in X} t\big(A(\vec{x}) \wedge C^l(\vec{x})\big) \\ &= \sum_{\vec{x} \in X} T_{\mathbf{L}}\big(t(A(\vec{x})), t(C^l(\vec{x}))\big). \end{aligned}$$

In order to find an approximating predicate for C^l, we use *FS-FOIL* [7] — a generalization of the *FOIL* algorithm [16, 17] to fuzzy predicates. FS-FOIL creates a sequential coverage of the goal predicate C^l by means of a set S^l consisting of fuzzy predicates which are composed by conjunction of atoms. The final descriptor is given as the disjunction of the predicates in S^l, i.e. FS-FOIL uses a disjunctive normal form to represent the description. In contrast to the original *FOIL* algorithm, which performs a steepest ascent search in the solution space, we use a beam search hill-climbing approach, to find a description (i.e. not only a single candidate, but the best k candidates are kept).

Algorithm 3 (FS-FOIL)

Input: goal predicate C^l
weight vectors $W = \{\vec{w}^1, \ldots, \vec{w}^{N_C}\}$
Output: predicate set S^l

open nodes $O = C^l(W)$
final predicate set $S^l = \emptyset$
intermediate predicate set *preds* $= \{\top\}$
do {
  preds' = best k predicates of *preds* according to gain measure G
  preds = expand all predicates in *preds'*
  if a predicate $A \in$ *preds* is accurate & significant
  {
    add predicate A to set S^l
    remove nodes covered by A from the set of open nodes O
    preds $= \{\top\}$
  }
} while stopping condition is not fulfilled;

FS-FOIL works with an intermediate set of predicates *preds* which are sequentially expanded in each iteration. The fuzzy set O corresponds to the samples from W fulfilling C^l which have not yet been described by a predicate in S^l.

In the first step of the loop, we select the best k predicates in *preds* (we use $k = 10$) according to the following entropy-based information gain measure G (see [14] for a detailed explanation):

$$G(A) = |A(W) \cap C^l(W)| \cdot \left(\log_2 \frac{|A(W) \cap C^l(W)|}{|A(W)|} - \log_2 \frac{|C^l(W)|}{|W|}\right)$$

Note that this is a slight adaptation of the original FOIL gain measure to the beam search (best k) used in FS-FOIL.

In the next step, the predicates are expanded by all atomic fuzzy predicates. The expansion of a predicate A with an atomic fuzzy predicate B is done by means of conjunction, i.e. $A \wedge B$. In case that *preds* only contains the initial trivial predicate $\top$ (the predicate which is always fulfilled), the expansion by an atomic predicate B is defined as $\top \wedge B = B$.

The third step is concerned with the search for an appropriate predicate within the set of expanded predicates. As quality criteria, we require significance and accuracy. The significance of a predicate A is defined as the common *support* of a predicate A and the goal predicate C^l. The support is defined as the ratio of the cardinality of samples commonly fulfilling A and C^l to the total number of samples:

$$\text{supp}(A,C^l) = \frac{|A(W) \cap C^l(W)|}{|W|} = \frac{1}{N_C} \cdot \sum_{i=1}^{N_C} t\big(A(\vec{w}^i) \wedge C^l(\vec{w}^i)\big).$$

The *accuracy* of a predicate A is defined as the *confidence* of predicate A with respect to C^l

$$\text{conf}(A,C^l) = \frac{\text{supp}(A,C^l)}{\text{supp(A)}},$$

where $\text{supp}(A)$ is defined as

$$\text{supp}(A) = \frac{|A(W)|}{|W|} = \frac{1}{N_C} \cdot \sum_{i=1}^{N_C} t\big(A(\vec{w}^i)\big).$$

Hence, the following holds:

$$\text{conf}(A,C^l) = \frac{\sum_{i=1}^{N_C} t\big(A(\vec{w}^i) \wedge C^l(\vec{w}^i)\big)}{\sum_{i=1}^{N_C} t\big(A(\vec{w}^i)\big)}$$

In other words, the confidence of A with respect to C^l is the ratio of the number of samples commonly fulfilling A and C^l to the number of samples fulfilling A.

Provided that we find a predicate $A \in \mathit{preds}$ fulfilling reasonable requirements in terms of support and confidence (by imposing two lower bounds $\text{supp}_{\min}$ and $\text{conf}_{\min}$, respectively [1]), we can add A to the final predicate set S^l and remove those nodes from the set of open nodes O which are covered/described by A. In detail, we replace O by $O \backslash A(W) = O \cap A(W)^c$ the membership function of which is defined as (for all $\vec{x} \in W$)

$$\begin{aligned} &T_{\text{L}}\big(\mu_O(\vec{x}), 1 - \mu_{A(W)}(\vec{x})\big) \\ &\quad = \max\big(\mu_O(\vec{x}) + 1 - \mu_{A(W)}(\vec{x}) - 1, 0\big) \\ &\quad = \max\big(\mu_O(\vec{x}) - \mu_{A(W)}(\vec{x}), 0\big). \end{aligned}$$

The loop terminates if either the percentage of undescribed nodes $\frac{|O|}{N_C}$ falls under a certain threshold (we use 10%) or no new significant and accurate predicates can be found by expansion anymore.

Given a result set of predicates S^l, the final predicate A^l approximating the goal predicate C^l is defined as the disjunction of all predicates from S^l. In detail, the truth value to which a sample $\vec{x} \in \mathcal{R}$ fulfills A^l is defined as

$$t(A^l(\vec{x})) = \bigvee_{A \in S^l} t(A(\vec{x})) = \underset{A \in S^l}{S_{\text{L}}}\, t(A(\vec{x}))$$

Since A^l only involves atomic predicates and logical operations, A^l can be regarded as an expression close to natural language which *describes* the goal predicate C^l and, thereby, the l-th cluster in our original data set.

Figure 1: *Original image (left) and its segmentation (right)*

5. Image Segmentation and Labeling: An Example

The approach proposed in this paper is widely applicable to any kind of data. We decided to demonstrate the method by means of a rather unusual example—image segmentation. Other examples (that are more familiar to the data mining domain) can be found in [8].

A typical application of clustering in computer vision is image segmentation, therefore, it seemed interesting to apply our three-stage approach to this problem. Moreover, the possibility to describe segments with natural language expressions gives rise to completely new opportunities in image understanding.

First experiments using pixel coordinates and RGB values showed good results in terms of accuracy, however, the descriptions were rarely intuitive. In order to overcome this problem, we added HSL features (hue, saturation and lightness), which match the way humans perceive color much closer, in the final description step. Since humans have a certain understanding of colors and intensities, it is more appropriate to assume predefined fuzzy predicates corresponding to natural language terms describing these properties. By this way, moreover, it can be avoided that the meaning of the descriptions depend on the image under consideration.

The example in Fig. 1 shows an image with 170×256 pixels, i.e. we have $K = 43520$ samples with $n = 8$ features. First, it was mapped onto a SOM with 10×10 nodes. Then four clusters and the corresponding descriptions were computed. On the right-hand side, Fig. 1 shows the computed segmentation. Figure 2 shows the final rule sets and the cross validation table which illustrates how good the clusters are described by the rule sets.

The rule sets in Fig. 2 can be interpreted as follows: The first rule set corresponds to the blue sky. The second rule set describes the black pants. The snow is classified by the third rule set. Finally, the jackets are identified by the fourth rule set.

6. Conclusion

In this paper, we have shown how the synergistic combination of different methods (self-organizing maps, clustering, and inductive learning) can be applied to perform descriptive

	Description
Cluster 1:	(Blue Is High) OR (Red IsAtMost Low AND Blue IsAtLeast VeryHigh)
Cluster 2:	Lightness IsAtMost Dark
Cluster 3:	Lightness IsAtLeast Light
Cluster 4:	(Hue Is Orange) OR (Hue Is Red) OR (Hue Is Yellow) OR (Hue Is Green AND Lightness Is Normal)

Figure 2: *Rule sets for the image segmentation*

data analysis. The modular design allows to change individual components on demand or even to omit one step (e.g. when labeled data is available, it is possible to use this information as the goal parameter in the inductive learning step, instead of performing unsupervised clustering first).

Due to the data reduction provided by the SOM preprocessing step, this approach is applicable to large data sets. It is clear, therefore, that this step is the bottleneck in terms of computation time, since it is the only component that has to process the complete input data set.

As an experimental study, we presented an image segmentation example. For traditional image segmentation, the cluster descriptions do not necessarily add value. These descriptions, however, can be considered as specific image features that can be used as features for image understanding and image indexing. Beyond the example in this paper, there is a big potential in business intelligence (as the salesman example in Section 1 underscores). Moreover, structuring and indexing of large text corpora appears to be another promising application field. Future work will go into these directions.

Acknowledgements

This work has been supported by the Austrian Government, the State of Upper Austria, and the Johannes Kepler University Linz in the framework of the K*plus* Competence Center Program.

References

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 207–216, Washington, DC, 1993.

[2] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Plenum Press, New York, 1981.

[3] U. Bodenhofer and P. Bauer. A formal model of interpretability of linguistic variables. In J. Casillas, O. Cordón, F. Herrera, and L. Magdalena, editors, *Interpretability Issues in Fuzzy Modeling*, volume 128 of *Studies in Fuzziness and Soft Computing*, pages 524–545. Springer, Berlin, 2003.

[4] U. Bodenhofer, M. De Cock, and E. E. Kerre. Openings and closures of fuzzy preorderings: Theoretical basics and applications to fuzzy rule-based systems. *Int. J. General Systems*, 32(4):343–360, 2003.

[5] G. J. Deboeck and T. Kohonen, editors. *Visual Explorations in Finance with Self-Organizing Maps*. Springer, London, 1998.

[6] M. Drobics. Choosing the best predicates for data-driven fuzzy modeling. In *Proc. 13th IEEE Int. Conf. on Fuzzy Systems*, pages 245–249, Budapest, July 2004.

[7] M. Drobics, U. Bodenhofer, and E. P. Klement. FS-FOIL: An inductive learning method for extracting interpretable fuzzy descriptions. *Internat. J. Approx. Reason.*, 32(2–3):131–152, 2003.

[8] M. Drobics, U. Bodenhofer, and W. Winiwarter. Mining clusters and corresponding interpretable descriptions — a three-stage approach. *Expert Systems*, 19(4):224–234, 2002.

[9] F. Hoeppner, F. Klawonn, R. Kruse, and T. A. Runkler. *Fuzzy Cluster Analysis — Methods for Image Recognition, Classification, and Data Analysis*. John Wiley & Sons, Chichester, 1999.

[10] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biol. Cyb.*, 43:59–69, 1982.

[11] T. Kohonen. *Self-Organizing Maps*. Springer, Berlin, second edition, 1997.

[12] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Trans. Commun.*, 28(1):84–95, 1980.

[13] R. S. Michalski, I. Bratko, and M. Kubat. *Machine Learning and Data Mining*. John Wiley & Sons, Chichester, 1998.

[14] T. M. Mitchell, editor. *Machine Learning*. McGraw Hill, 1997.

[15] Z. Pawlak and A. Skowron. A rough set approach for decision rules generation. In *Proc. IJCAI'93 Workshop W12: The Management of Uncertainty in Artificial Intelligence*, Chambéry, 1993.

[16] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3):239–266, 1990.

[17] J. R. Quinlan and R. M. Cameron-Jones. Induction of logic programs: FOIL and related systems. *New Generation Computing*, 13:287–312, 1995.

[18] H. Ritter and K. Schulten. Convergence properties of Kohonen's topology preserving maps: fluctuations, stability, and dimension selection. *Biol. Cyb.*, 60(1):59–71, 1988.

[19] E. H. Ruspini. A new approach to clustering. *Inf. Control*, 15:22–32, 1969.

[20] J. H. Ward Jr. Hierarchical grouping to optimize an objective function. *J. Amer. Statist. Assoc.*, 58:236–244, 1963.

[21] L. A. Zadeh. Fuzzy sets. *Inf. Control*, 8:338–353, 1965.