Available online at www.sciencedirect.com
SCIENCE DIRECT®
Fuzzy Sets and Systems 149 (2005) 209–233
FUZZY sets and systems
www.elsevier.com/locate/fss

Semi-supervised learning in knowledge discovery

Aljoscha Klose*, Rudolf Kruse

Department of Knowledge Processing and Language Engineering, Otto-von-Guericke-University of Magdeburg, D-39106 Magdeburg, Germany

Abstract

Recently, semi-supervised learning has received quite a lot of attention. The idea of semi-supervised learning is to learn not only from the labeled training data, but to exploit also the structural information in additionally available unlabeled data. In this paper we review existing semi-supervised approaches, and propose an evolutionary algorithm suited to learn interpretable fuzzy if-then classification rules from partially labeled data. Feasibility of our approach is shown on artificial datasets, as well as on a real-world image analysis application.

Keywords: Semi-supervised learning; Fuzzy classification rules; Data mining; Image analysis

1. Introduction

Modern information technology, which produces more powerful computers every year, makes it possible today to collect and process huge amounts of data at very low cost. However, exploiting the information contained in the data in an intelligent way turns out to be fairly difficult. In reply to this challenge a new area of research has emerged, named "knowledge discovery in databases" or "data mining" [12]. Some well-known analysis methods and tools that are used in data mining are, for instance, statistics (regression analysis, discriminant analysis, etc.), time series analysis, decision trees, cluster analysis, neural networks, inductive logic programming, and association rules.

However, characteristics and thus demands of current applications are changing and call for new algorithms. In this introduction we outline which important trends we see in the sources of the data to be analyzed, and thus which trends we expect in the characteristics of the data. We argue why fuzzy methods

* Corresponding author.
E-mail address: klose@iws.cs.uni-magdeburg.de (A. Klose).

0165-0114/$ - see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.fss.2004.07.016

are well suited to deal with these changing demands, and why these challenges consequently lately lead to a considerably growing interest of the research community into semi-supervised learning methods.

1.1. Changing data characteristics

Most classical data mining methods, like decision trees and neural networks, expect an input of single uniform tables of tuples of attribute values. In many modern applications, however, the data to be analyzed come from heterogeneous information sources: Many of the archives contain images, texts, video, or even sound data. We can certainly not expect to find data mining algorithms that are generally applicable to all mentioned kinds of information sources. The approaches will always strongly depend on some kind of application-specific pre-processing to extract characterizing features from the specific type of media. Additionally, to enable data mining in such feature spaces we suppose that it is crucial to exploit any available *a priori* knowledge, and thus to have algorithms that support to incorporate such information.

As the data seldomly come from well-designed experiments or measurements, they are often unevenly distributed in the input space and class frequencies are often unbalanced. Furthermore, the data are often of low quality. Algorithms must thus be able to deal with uncertainty and imprecision. Missing values are also a common problem that data mining algorithms should be able to handle. A special case of missing values are missing class labels: The focusing of data mining methods on supervised learning is a severe drawback in many real world applications. In contrast to the abundance of data available in the archives, labeling these data is often a problem. In many cases, the labels for the training samples have to be assigned manually or determined by expensive analyses. Typical examples of such domains include speech processing, object recognition, text classification, or medical or biological applications. Labeling a complete dataset can become an at least tedious if not infeasible task when there are many objects—and in some applications "many" can easily mean tens of thousands. With increasing sizes of the databases to be analyzed, learning from data that is only partially labeled becomes more and more interesting.

1.2. The role of fuzzy techniques

The outlined characteristics of the data sources—their quantity, complexity, dimensionality and imperfection—, the essential of extracting understandable patterns from these, and the need to incorporate available background knowledge in that process, make fuzzy techniques an interesting tool for data mining. They can transform between computer representations and (naturally linguistic) human concepts. The inherent imprecision of words is not necessarily a weakness, but, on the contrary, can be crucial to model complex systems. From our own experience we observed that many practical applications have this certain robustness where full precision is not necessary. In such cases, exaggerated precision can be a waste of resources, and solutions obtained using fuzzy approaches might be easier to understand and to apply. Good examples of models that gain their strengths by explicitly taking into account vagueness, imprecision or uncertainty are systems based on fuzzy rules.

One important task in data mining is classification. Fuzzy *if-then* rules have become popular for this task, as the use of linguistic variables is close to human descriptions of structure in data. For data analysis we are looking for procedures that can extract fuzzy rules from an example dataset. Ideally, these rules allow to accurately classify new objects, and still describe the structure of the data distribution in an understandable fashion. If we apply such techniques, we must be aware of the trade-off between precision and interpretability. However, the results in data mining are not only judged for their accuracy,

but also for their interpretability, as the ultimate goal is to extract human understandable patterns. Not all possible choices for fuzzy models are equally suited to accurately solve the task at hand and still generate interpretable models.

1.3. Semi-supervised learning

Most existing approaches to fuzzy rule extraction are supervised, i.e. they expect that all data are labeled. However, as initially mentioned, in many current domains it is not possible to access the labels for all objects. In such cases, one usually confines the examples to a certain—hopefully representative—fraction of the data and leaves the unlabeled data aside. However, though unlabeled, the additional data might still bear valuable information on the true distribution of the objects in the input space.

There have been several proposals for methods that use the remaining, otherwise discarded data to support the learning of a classifier. With growing database sizes it is not surprising that there is an increasing interest in approaches that are able to learn from partially labeled data. However, most approaches induce models that are less human-understandable than fuzzy if-then rules. We thus investigate how to combine fuzzy classification rules and semi-supervised learning.

1.4. Outline

In Section 2, we review the key concepts of fuzzy rule based classifiers, including evaluation and induction of fuzzy rules. In Section 3, we review previous work in the field of semi-supervised learning. We focus on methods for semi-supervised learning of fuzzy models. However, little has been done on the extraction of *interpretable* fuzzy rules from partially labeled data. We thus propose our own semi-supervised approach for fuzzy rule induction based on evolutionary algorithms in Section 4. The working of all semi-supervised methods is illustrated and compared on artificial datasets. In Section 5, an additional real-world application is presented to show the applicability of our approach.

2. Fuzzy rule-based classification

A fuzzy classification system [19] consists of a rule base, which contains a set of fuzzy classification rules (like the one shown below), and an inference engine, which evaluates the rule base for the datum to be classified. The basic idea of fuzzy classification systems is to describe the areas of the input space, to which different class labels are assigned, by vague cluster prototypes. These prototypes are defined by a number of fuzzy sets which characterize them in the different dimensions of the domain under consideration. That is, a specific cluster β is defined by a fuzzy classification rule r of the form:

if A_1 is μ_1 **and** A_2 is μ_2 **and** ... **and** A_n is μ_n
then pattern $(A_1, A_2, \ldots, A_n)$ belongs to class c,

where the μ_k are fuzzy sets describing the cluster β in the corresponding dimensions. In addition, some approaches introduce so-called rule weights w_r, which are intended to indicate the "importance" or "reliability" of a rule.

2.1. Supervised extraction of fuzzy rules

By far the most common learning paradigm in machine learning is supervised learning. The supervised learning task consists of a dataset of tuples $\boldsymbol{x}$ together with the corresponding class labels c. The task is to generalize the regularities found in the labeled example data.

There is a variety of methods that have been proposed to induce fuzzy rules from data. A popular way are neuro-fuzzy systems, which use learning algorithms derived from neural network theory to generate fuzzy rules [25]. Another group of approaches are based on decision trees. Decision trees can very efficiently be induced from data by greedy divide-and-conquer heuristics, and rules can afterwards be extracted from the trees [37]. Several extensions to generate fuzzy rules can be found in the literature [48,20].

Genetic or evolutionary algorithms are also often discussed for optimizing or creating fuzzy systems. The advantage of evolution strategies is the ability to modify and optimize model structure and parameters, whereas most optimization strategies can only adapt model parameters. Thus a unified optimization of rule base and membership functions can be performed. This comes at the cost of an (possibly drastically) enlarged search space. Discussions of evolutionary algorithms for fuzzy rule systems can for example be found in [9].

The *fuzzy min–max neural network* (*FMM*) proposed by Simpson [41,42] is another classical fuzzy rule induction approach. The approach is *hyperbox-oriented*, i.e. each rule is associated with a hyperbox in input space, within which the rule fully applies. The hyperboxes are represented by pairs of minimal and maximal value in each dimension. [1] The firing strength of the rule outside the hyperbox is controlled by a fuzzy membership function and monotonously decreases with distance from the hyperbox. The original algorithm loops once over the tuples. If a tuple is already contained in a compatible hyperbox, i.e. if the corresponding rule's and the tuple's label match, the algorithm proceeds to the next tuple. If no such hyperbox exists, the algorithm tries to extend the closest compatible hyperbox. If this violates the maximal allowed size of a hyperbox, the algorithm creates a new hyperbox containing only the tuple. Additionally, the resulting rules are checked for overlap with the other rules. If overlap occurs, the hyperboxes are contracted based on the "minimal adjustment principle", i.e. that dimension and min–max point is chosen that allows to resolve the overlap with a minimal change.

2.2. Unsupervised extraction of fuzzy rules

The second important learning paradigm is unsupervised learning, i.e. learning from example data without known class labels. The task to reconstruct the class information from the inherent structure in the data, is also known as cluster analysis. Cluster analysis tries to find groups in the data such that objects in the same group are similar to each other. Fuzzy extensions of cluster analysis represent the clusters by multidimensional fuzzy sets and thus can better deal with partially overlapping clusters [19]. The results of fuzzy clustering can be transformed into a fuzzy rule base, and can therefore be used for unsupervised extraction of fuzzy rules from data [21]. Every cluster represents a fuzzy if-then rule. The fuzzy sets in the single dimensions are derived by projecting the clusters to the specific dimensions. A fuzzy rule base can be obtained by projecting all clusters. Usually the projection is approximated by triangular or trapezoidal fuzzy sets. Due to this approximation and the projection of the clusters the generated fuzzy rules only

[1] The name "Min–Max" classifier refers to these corner points of the hyperboxes. It does not specify the inference mechanism, which is actually "average-max" in the original proposal [41].

roughly represent the original clusters. This error can be reduced if the cluster search is restricted to axes parallel clusters. It is also possible to improve the classification rules by fine tuning them with a neuro-fuzzy approach.

However, the main problem of unsupervised learning is that it strongly depends on the given distance measures and normalizations of the data. It has commonly problems in finding the right number of clusters. Usually, we cannot guarantee that the clusters in the data space correspond to meaningful classes of the objects. A further problem of the fuzzy rule bases obtained from cluster analysis is that they are sometimes hard to interpret linguistically, since the resulting fuzzy sets are not restricted to match any semantic interpretation.

3. Semi-supervised learning

The methods for the extraction of fuzzy classification rules from data that we mentioned in the last sections are either supervised or unsupervised. Both learning paradigms have their drawbacks. The main drawback of supervised learning is clearly its need for supervision, i.e. the need to present labels together with the objects. The result of unsupervised learning, however, strongly depends on a number of prior assumptions (explicitly or implicitly). Thus, it depends on an appropriate choice of, e.g. attribute scaling, distance measure, distribution function and expected number of classes or clusters, whether the clusters found in the data space correspond to any "meaningful" classes of objects. Hence, unsupervised learning does in many cases not yield satisfactory results, and supervised learning is much more common in practice.

If, however, the remaining, otherwise discarded, unlabeled data contained some additional useful information, it would be an appealing idea to use it to support the learning of the classifier. The idea of exploiting the information in both labeled and unlabeled data is not new, and early approaches of semi-supervised learning date back to the 1980s [35]. However, as argued above, with tremendously growing sizes of datasets in real-world applications labeling all of the data becomes more and more infeasible and the exploitation of additional unlabeled examples gets increasingly interesting. Thus over the past years a growing number of publications, workshops and conference tracks on semi-supervised learning can be observed.

In most publications, the approach of Pedrycz [35] is cited as the first work in the area of semi-supervised clustering. Twelve years later he revisited the problem and published some extended results and more detailed discussion of his 1985s ideas. In [36] he stated that

> "surprisingly, limited attention has been paid to the mechanisms of partial supervision."

Ever since this publication partial supervision has obviously come into the focus of current research in computational intelligence. There is a growing quantity of publications in that field, and Successful applications of semi-supervised approaches have been reported, for example, in the field of image processing [4,47] and especially in text classification [32,29]. A number of different ideas have been proposed how to combine the information of labeled and unlabeled data. These ideas can be categorized into four main groups:

(1) *Labeled examples as seed points*: A supervised classifier is used to build a model from the class information of labeled points. The model is then used to apply labels to the unlabeled points, and is iteratively re-learned. Approaches differ in the type of model used (e.g. point prototypes, naive

Bayes classifiers or neural networks), and the "speed" of applying the new labels (from one pattern per iteration to labeling all unlabeled patterns in one step) [4,15,29,46].

(2) *Labeled examples as cluster labels*: An unsupervised algorithm is used to find structure in the dataset, e.g. by cluster analysis. The clusters are then labeled using the given labeled points. This can be done in various ways [3,34]. The labeled points can also be used to guide the clustering (e.g. the number of clusters) [1,15]. Dara et al. proposed to find low-dimensional structures by training a self-organizing map from all available data in an unsupervised manner [10]. The map nodes are then labeled from the labeled dataset.

(3) *Unlabeled examples for density estimation*: The abundance of unlabeled examples can be used for a more reliable estimation of the probability density function in the input space. This is similar to the second group, as cluster analysis also performs a kind of density estimation. However, approaches like [26,43] explicitly model and use the probability density function. A different approach has been proposed in [2,13] for the semi-supervised learning of support vector machines. Instead of using regions of high density to find clusters, regions of data scarcity are used to find the optimal class borders. Verikas et al. propose a similar method for the learning of feed-forward neural networks [46].

(4) *Specialized objective functions*: There is a variety of approaches that have specialized objective functions that can take into account labeled *and* unlabeled examples, for example by adding a penalty term for labeled examples that are assigned to 'wrong' clusters [35,36,45] or formulating a combined probability [31,47]. In [11] a mixture of cluster dispersion and cluster impurity is optimized. Our approach presented in Section 4.2 falls also into this group.

This categorization can only give a rough overview. The borders between the categories are not crisp, and many approaches could be assigned to more than one category, depending on the point of view.

A number of approaches have been proposed for models like, for example, neural networks or support vector machines, that are generally hardly human understandable. Little has been done on the semi-supervised extraction of (interpretable) fuzzy rules. The methods described in the following sections are able to induce fuzzy models in a partially supervised manner. In Section 4, we will discuss their capabilities and their suitability to induce interpretable fuzzy rule bases.

3.1. Semi-supervised extensions of fuzzy clustering

It is probably easier to support an unsupervised algorithm with additional labels than vice versa. Thus, it is not surprising that there are a number of semi-supervised extensions of fuzzy clustering.

3.1.1. Semi-supervised FCM: ssFCM

Bensaid et al. [4] proposed an extension of the fuzzy c-means algorithm [5]. An unlabeled dataset D^u and a labeled dataset D^l are concatenated, with a common membership matrix U with $n_l + n_u$ columns and $n_c = |C|$ rows (i.e. one row for each class c_i). The labels of D^l are presumed to be correct. This is taken into account by setting the corresponding columns in U to the 1-in-n encoded class labels, i.e. let j_ω denote the column index corresponding to object ω, then

$$u_{ij_\omega} = \begin{cases} 1 & \text{if } c^\omega = c_i, \\ 0 & \text{else.} \end{cases} \tag{1}$$

These columns are fixed, i.e. they are ignored when updating U from the current cluster prototypes V. The memberships for the remaining columns are calculated exactly as in the original FCM. With these memberships the updating of the prototypes V remains basically unchanged. The only difference is the introduction of a factor α^ω to weight the influence of the labeled examples:

$$v_i = \frac{\sum_{\omega \in D^l \wedge c^\omega = c_i} \alpha^\omega \boldsymbol{x}^\omega + \sum_{\omega \in D^u} u_{ij_\omega}^m \boldsymbol{x}^\omega}{\sum_{\omega \in D^l \wedge c^\omega = c_i} \alpha^\omega \quad + \sum_{\omega \in D^u} u_{ij_\omega}^m}. \tag{2}$$

If nothing is known about the reliability of individual examples or classes, the same α^ω is chosen for all $\omega \in D^l$.

The idea of this approach is simply to trust the labeled examples, and fix them in the (otherwise almost unmodified) FCM clustering algorithm. Trusting the labeled examples is of course reasonable. The problem is that, as the authors also remark, this approach expects the labeled examples to be good estimations of the (final) cluster prototypes. If the chosen (or given) labeled examples are untypical for their cluster, they will unwantedly attract the prototype. On the other hand, if they already were good estimations of the cluster centers, it would not be necessary to take the additional unlabeled data into account, and we could simply use any supervised approach on them.

The authors note that the straight-forward modification of the update equations interferes with the minimization of the objective function the original FCM is based on. Alternatively, the approach of the following section puts the focus on the extension of the objective function, and derives modified update rules.

3.1.2. Partially supervised Gustafson and Kessel

The approach by Pedrycz [35,36] extends the objective function of Gustafson and Kessel [17]. As in the previous section, the (input vectors of the) datasets D^u and D^l are assumed to be joined in a matrix X. Additionally, a vector $\vec{b} = [b_j]$ with binary entries

$$b_{j_\omega} = \begin{cases} 1 & \text{if } \omega \in D^l \\ 0 & \text{else} \end{cases} \tag{3}$$

marks the labeled examples. A matrix F contains the known labels in 1-in-n encoding (cf. Eq. (1)). The original Gustafson/Kessel objective function J_m^{GK} is extended by adding a penalty term

$$J_m(U, \boldsymbol{V}, \boldsymbol{A}) = J_m^{\mathrm{GK}}(U, \boldsymbol{V}, \boldsymbol{A}) + \alpha \sum_{i=1}^{k} \sum_{j=1}^{n} (u_{ij} - f_{ij} b_j)^m \|\boldsymbol{x}_j - v_i\|_{A_i}^2, \tag{4}$$

where α is intended to balance the influence of the labeled data. Intuitively, the added term penalizes labeled points that have low membership degrees to the cluster suggested by their label.

In spite of the modifications of J_m^{GK}, the update rules for the cluster centers $\boldsymbol{V}$ and the covariance matrices $\boldsymbol{A}$ remain identical to the original algorithm. The modified update rule for U can be shown to be (for $m = 2$)

$$u_{ij} = \frac{1}{1+\alpha} \cdot \frac{1 + \alpha(1 - b_j \sum_{i'=1}^{k} f_{i'j})}{\sum_{i'=1}^{k} \frac{\|v_i - x_j\|_{A_i}^2}{\|v_{i'} - x_j\|_{A_{i'}}^2}} + \frac{\alpha}{1+\alpha} \cdot f_{ij} b_j. \tag{5}$$

For unlabeled examples, this also leads to the original update rule

$$u_{ij} = \frac{1}{\sum_{i'=1}^{k} \frac{\|v_i - x_j\|^2_{A_i}}{\|v_{i'} - x_j\|^2_{A_{i'}}}}. \tag{6}$$

As we assumed 1-in-n encoding (and thus $\sum_{i=1}^{k} f_{ij} = 1, \forall j$), the memberships of the labeled examples are updated with

$$u_{ij} = \frac{1}{1+\alpha} \cdot \frac{1}{\sum_{i'=1}^{k} \frac{\|v_i - x_j\|^2_{A_i}}{\|v_{i'} - x_j\|^2_{A_{i'}}}} + \frac{\alpha}{1+\alpha} \cdot f_{ij}. \tag{7}$$

For $\alpha = 0$ (and thus $\alpha^u = 1$, $\alpha^l = 0$), the labels of D^l are completely ignored. Interestingly, for $\alpha \to \infty$, the Gustafson/Kessel clustering with partial supervision becomes equal to ssFCM (except that ssFCM uses fixed covariance matrices $A_i = \mathbb{1}, \forall i$). For intermediate values of α, the memberships of the labeled data are more or less pushed towards their corresponding clusters. In the following, we refer to this algorithm as ssGK.

3.1.3. Semi-supervised point-prototype clustering

Both presented semi-supervised fuzzy clustering approaches expect that each class can be represented by one cluster. Complex classes, that cannot be appropriately described by one cluster, will lead to significantly decreased performance.

Bensaid et al. [3,28] propose a semi-supervised point-prototype clustering algorithm that is based on the fuzzy c-means algorithm. Their algorithm, called ssPPC, first overpartitions the unlabeled input patterns in a fully unsupervised manner using fuzzy c-means (although the authors remark that any point-prototype cluster algorithm can be used). Then the resulting clusters are labeled based on the labeled examples. In a final step, the unlabeled tuples are labeled based on their memberships to the clusters. The algorithm performs the following steps:

- Cluster X^u with fuzzy c-means. The number of clusters is heuristically chosen as n_d, i.e. there is one cluster for each labeled example. Let $\boldsymbol{V} = \{v_1, \ldots, v_{n_d}\}$ denote the resulting cluster prototypes, and $U_{(n_d, n_u)}$ the matrix of cluster memberships.
- Assign class labels to the cluster prototypes v_i. The labels of the prototypes $L_{(n_d, c)}$ can be possibilistic, i.e. $l_{ij} \in [0, 1]$. Three alternative strategies have been proposed to find the labels [28]:
  - A. For each prototype v_i find the nearest labeled example $x^d_{j_{nn(i)}}$ from X^d and adopt its (crisp) label $l_j = u^d_{j_{nn(i)}}$.
  - B. Assign each labeled example to its nearest prototypes. Let X^d_j be the labeled examples of class j, and $X^d_{ij} \subset X^d_j$ the subset assigned to prototype v_i. Use their fraction to define possibilistic class labels:

$$l_{ij} = \frac{X^d_{ij}}{X^d_j}. \tag{8}$$

C. The distance between a prototype v_i and the labeled examples of a class j, X_j^d, is measured as $d_{ij} = \min_{x^d \in X_j^d}\{\|v_i - x^d\|\}$. The labels for the prototype are defined as the ratio between the class distances:

$$l_{ij} = \frac{1}{c-1}\left(1 - \frac{d_{ij}}{\sum_{k=1}^{c} d_{ik}}\right). \tag{9}$$

- Compute the labels $\hat{u}_{lj}, l = 1, \ldots, n_u$ for the unlabeled tuples $x_l^u \in X^u$ by aggregating the prototype labels and the tuples' cluster memberships:

$$\hat{u}_{lj} = \min\left\{1, \sum_{k=1}^{n_d} l_{kj} u_{kl}\right\}. \tag{10}$$

If necessary, crisp labels are generated from $\hat{u}_{lj}$ by winner-takes-all defuzzification.

Bensaid et al. use this approach for transduction, i.e. they label only the unlabeled fraction of the example data. Application to new data was not intended by the authors. However, it is possible by calculating memberships U for new data from the given prototype positions.

As this approach allows multiple clusters for each class, it performs better on datasets that need this flexibility. Therefore, the authors call their approach the "successor" of their proposal in [4] (see Section 3.1.1). However, the underlying mechanisms are obviously rather different. Additionally, due to the initial overpartitioning, ssPPC cannot appropriately take the density information of the unlabeled data into account, and results are close to supervised learning. The experiments on artificial datasets in Section 3.3 underline this problem.

3.2. Generalized fuzzy min–max classifier: GFMM

In Section 2, we reviewed the *fuzzy min–max classifier* as the prototype of hyperbox oriented fuzzy rule learners. In [14] that scheme is extended to semi-supervised learning. [2] The performed modifications to the original operations (*initialization* of new rules, *expansion*, *overlap test*, and *contraction*) are astonishingly straightforward:

- First, the definition of *compatibility* between hyperboxes and tuples is modified. If the label of a tuple is unknown, or a hyperbox contains only unlabeled tuples and thus its consequent label is unknown, then there is no information that the tuple is *not* compatible to the hyperbox. Therefore, these cases are defined as compatible.
- The first labeled example that is added to a hyperbox defines the corresponding rule's consequent.
- Unlabeled hyperboxes are tested for overlap with any other hyperbox, as they might assume any label.

The expansion and contraction operations remain unchanged.

GFMM was successfully applied a "toy dataset" and the Iris data. A similar semi-supervised extension of the original fuzzy min–max classifier scheme is presented in [27]. The authors basically employ the same semi-supervised operations, but use hyperspheres instead of hyperboxes as prototype shapes.

[2] The extension to semi-supervised learning is only one aspect among several proposed modifications and enhancements.

3.3. Capabilities and limitations

In the previous sections, we presented a variety of methods that learn from labeled and unlabeled data. They have in common that they either induce fuzzy rules, or fuzzy cluster prototypes, or are general enough to be extended to fuzzy classifier learning. Before we further discuss their suitability for the semi-supervised learning of interpretable fuzzy rules on realistic problems, we demonstrate and compare their classification abilities on an illustrative example.

So far, there are no publicly available and commonly agreed on benchmark datasets for semi-supervised classifiers. The usual way of testing the methods uses the datasets of the UCI machine learning repository [6], and treats an arbitrarily chosen random subset as unlabeled. However, evenly distributed missing labels are neither the most realistic case in practice, nor the case, where semi-supervised learning can be expected to yield its best results. Another problem is that many UCI datasets are rather small. Therefore, it is not uncommon that in publications the labels of 50 (or more) percent of the data are used for learning. From the practical point of view, the main motivation for semi-supervised learning algorithms is of course their capability to work in cases where $|D^u| \ll |D^l|$.[3] However, for real-world problems with small numbers of possibly less representative labeled examples the underlying ground truth for the unlabeled data is often not available, and therefore fair comparison of algorithms by e.g. the number of misclassifications is not possible. Hence, we constructed two artificial datasets, that contain some of the problems that we expect to occur in realistic applications. The datasets also raise difficulties for supervised (on the labeled subset) or unsupervised methods. These examples help to illustrate how the presented semi-supervised methods cope with these problems and which deficiencies they have.

Both datasets have two input dimensions and two classes, marked by "△" and "○". Each dataset contains 1000 points, however, the labels of only a fraction of these points is assumed to be known (these are indicated by bigger symbols).

In the first dataset (cf. Fig. 1) the classes build two well separated clusters that should easily be identified by most unsupervised clustering algorithms. The problem of the given class labels is that they do not characterize the classes very well. A supervised classifier, which tries to separate the examples in D^l, thus is easily misled. In practice such situations might be rather common if the data is labeled by hand. Manually given labels are often defined for those objects that differ most significantly from the other classes, and not for those that are typical for their own class.

The joint partially labeled dataset $D^l \cup D^u$ was used to induce the classifiers. The induced models have been applied to an independent test dataset of 1000 tuples from the same distribution. The results are shown in Fig. 3. Misclassified points are shown darker with the symbol that corresponds to their true class label. Additionally, the decision boundaries are shown. The background shading of the diagrams relates to activation of a rule or to distance from prototype center.

Both partially supervised algorithms directly based on fuzzy clustering—namely ssFCM and ssGK—perform rather well on this dataset. This is not surprising, as the unlabeled data contain much information on the optimal location of the decision boundary, and thus even pure unsupervised learning would yield satisfying results. Generally speaking, ssPPC is closer to supervised learning. Its decision boundary is located half-way between the labeled examples and thus less optimal for the given unlabeled examples.

[3] If someone has already labeled 50% of the data, he or she can probably as well label all of it. Or, from the learner's side: if 50% of the data are not sufficient to learn the model in an fully supervised manner, then the remaining 50% will probably not be of much help either (especially as they are unlabeled).

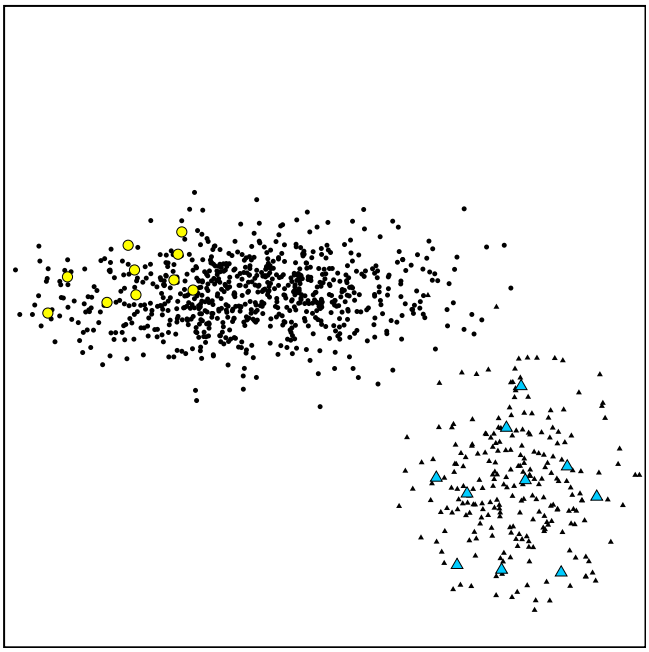

Fig. 1. The first artificial dataset.

GFMM fails almost completely to assign labels to the unlabeled examples and thus produces an extremely high error rate.

The second dataset is more complex (cf. Fig. 2). The first class is split into two clusters. One of them is well separated from the remaining examples. However, the other cluster lies very close to, and even slightly overlaps with the second class. Their boundary can hardly be found in the density distribution of the points. Thus, unsupervised learning will very probably fail on this dataset. Additionally, the chosen labeled examples are not very representative for their respective clusters. A supervised classifier, that can only learn from the labeled data, is again easily mislead. We suppose that such distributions bear the highest potential for semi-supervised methods (Fig. 3).

The partial supervision of ssFCM and ssGK could potentially solve these challenges. However, both approaches do not allow to induce more than one cluster per class. As can be seen in Figs. 4a and b, this significantly deteriorates the performances of these semi-supervised algorithms on this dataset. Notice that in both approaches, the upper left cluster is attracted to a certain degree by the labeled examples at the lower right, as they belong to the same class. The result of ssPPC, shown in Fig. 4c is rather good. However, it shows a tendency of ssPPC to overfitting due to the highly flexible decision boundary. Obviously, this boundary is pretty close to that resulting from nearest neighbor classification from the labeled data alone. This is not surprising, as ssPPC does not really seek for structure, but effectively uses the distance from the labeled examples. Fig. 4d depicts the results of GFMM. It again generates hyperboxes from unlabeled examples that remain unlabeled and thus unavoidably produce errors. As in the first example, the error for these rejected tuples could be reduced simply by *guessing* their class label (or that of the containing hyperbox). However, this does obviously not help to reveal the structure of the dataset.

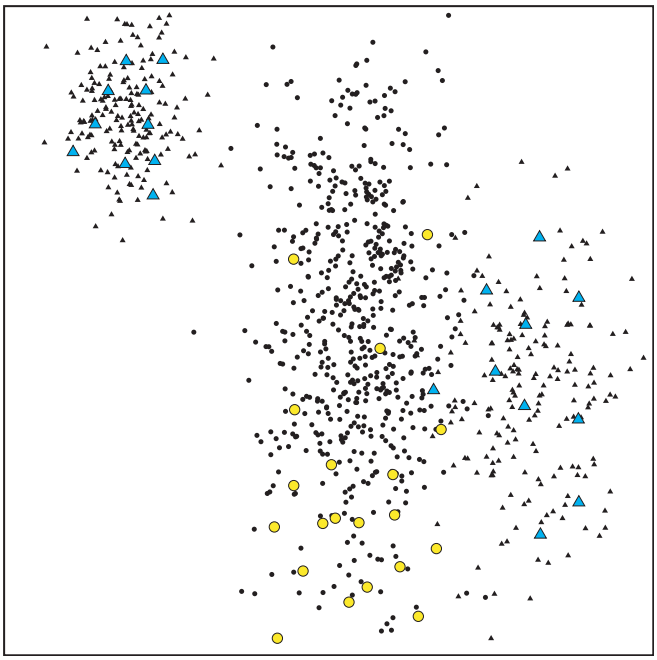

Fig. 2. The second artificial dataset.

As already mentioned, for knowledge discovery, we are interested into models that give the user insight into his data. Hence, we discuss the capabilities of the presented semi-supervised models in the following aspects:

- *Expressiveness:* How flexible is the induced model? What kind of distribution and decision boundaries can be modeled? How capable are single rules/clusters? How flexible is the model induced by the interaction of rules/clusters?
- *Interpretability:* How difficult is it to transform the model into a fuzzy rule base? How readable will this rule base be? How many rules does it have? Will it give the user insight to the data?
- *Semi-supervised learning:* How capable was the semi-supervised learning algorithm? Does it depend on the representativeness of the examples? How strongly does it respect the information of the labeled data D^l?

Generally, fuzzy classifiers are very flexible models that can approximate arbitrary class boundaries with arbitrary precision—at least if the number of fuzzy rules is unrestricted. Both, ssPPC and GFMM use generally rather high numbers of clusters or hyperboxes, respectively. This allows them to represent a wide range of decision boundaries. On the other hand, the strict cluster-class correspondence of ssFCM and ssGK wastes much of the flexibility of fuzzy classifiers. While ssGK can partially compensate for that weakness by its more flexible cluster shape, ssFCM can only represent Voronoi cells, as it considers only Euclidean distances. This means that for any problem to be perfectly modeled with ssFCM, each pair of classes has to be linear separable, which is obviously a strong restriction.

One common motivation for using fuzzy methods is the demand for explicable data analysis results, that can be understood by human experts and, for example, checked for plausibility. However, the models generated by fuzzy methods somewhat differ in their interpretability.

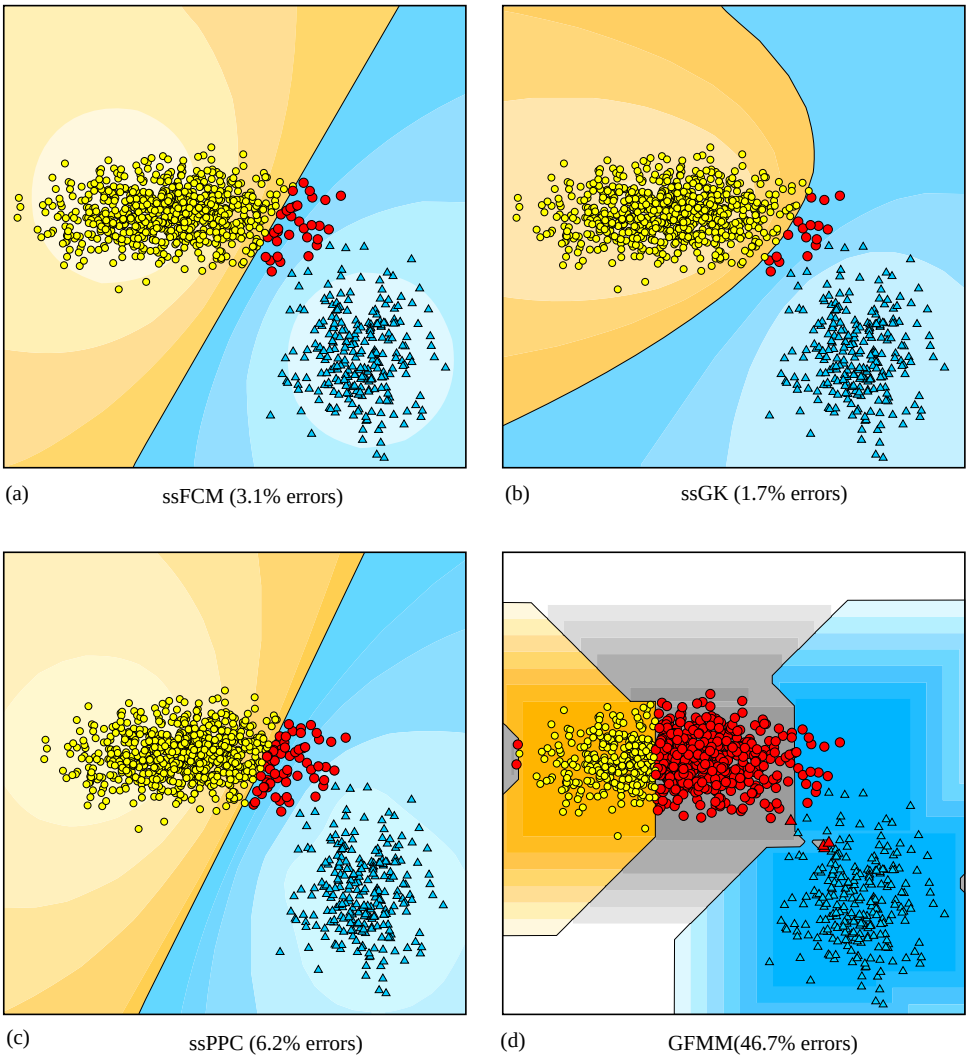

Fig. 3. Experimental results on the first artificial dataset: (a) ssFCM (3.1% errors); (b) ssGK (1.7% errors); (c) ssPPC (6.2% errors); (d) GFMM (46.7% errors).

As we would like to have the classification knowledge in form of fuzzy rules, for all models but GFMM the first problem is to transform clusters into fuzzy membership functions and rules. In GFMM each hyperbox corresponds to a fuzzy rule with trapezoidal membership function and $\top_{\min}$-conjunction. For the cluster approaches, the projection method mentioned in Section 2.2 can be used. If we restrict ssGK to axis-parallel clusters, the projection errors should be reasonably small.

All compared approaches yield local (i.e. rule-wise) definitions of membership functions. In general it is much easier to find linguistic descriptions for globally defined membership functions that are shared by all rules. However, for ssFCM and ssGK, only one membership function per class is created for each input dimension. If the number of classes is small, generating linguistic descriptions—like “small”, “medium”,

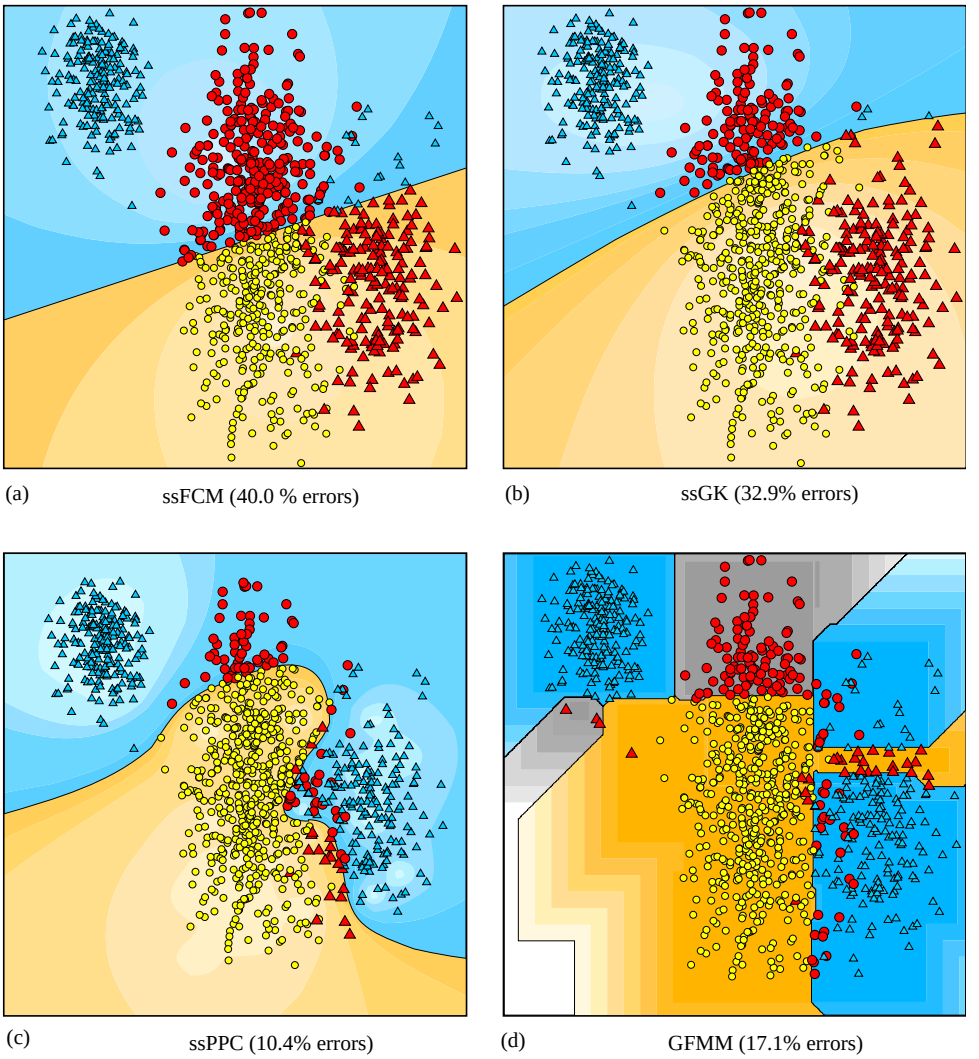

Fig. 4. Experimental results on the second artificial dataset: (a) ssFCM (40.0% errors); (b) ssGK (32.9% errors); (c) ssPPC (10.4% errors); (d) GFMM (17.1% errors).

"large"—will be possible in most cases. The generally high numbers of rules extracted by ssPPC and GFMM make a linguistic description almost infeasible.

One aspect that is often more relevant than expressiveness and interpretability is the ability to learn high quality models from partially labeled data. Semi-supervised learning can be expected to be most effective in situations where many unlabeled, but only few labeled, possibly untypical examples are available for inducing a classifier. One question was thus whether an algorithm is able to cope with such situations. None of the models showed an overall convincing performance on the considered example datasets. Especially the second dataset is certainly rather difficult, and the algorithms could perform much

better (e.g. if the cluster-class relationship is not violated for ssFCM/ssGK). However, we assume that the artificial datasets contain realistic problems of partially labeled datasets.

We conclude that none of these algorithms is fully appropriate for the semi-supervised induction of readable and interpretable rule based fuzzy classifiers. In the next section, we propose our semi-supervised learning algorithm for fuzzy classification rules.

4. An evolutionary algorithm for semi-supervised fuzzy classification

There are several reasons why we see the need to propose another algorithm. First, the algorithms are not aimed at inducing fuzzy classifiers that are satisfying when interpreted as rule bases. The strict cluster-class correspondence of ssFCM and ssGK wastes much of the flexibility of fuzzy classifiers. The extreme overpartitioning of ssPPC, on the other hand, degrades interpretability and readability of the rule base. We would like to have an algorithm that exploits the expression ability of fuzzy rule bases (including, for example, flexible cluster shapes, or "don't cares" in the antecedents), and still preserves interpretability and readability. Secondly, neither of the approaches is very good in learning when the labeled examples are less representative for the prototypes. However, these are the cases where semi-supervised learning could be most powerful (if the labeled examples were representative, supervised methods would also perform well).

In this section, we present an alternative approach to the induction of fuzzy rules from partially labeled data. We use an evolutionary algorithm (EA) that incorporates labeled and unlabeled examples into its objective function. Several reasons speak for evolutionary rule learning in comparison to other search techniques like, for example, alternating optimization or gradient descent. EAs allow to learn structure and incorporate constraints to maintain interpretability. They are less sensitive to premature convergence in local minima, which can be a problem in semi-supervised learning due to the "strong" guidance from the labeled and the "weak" guidance from the unlabeled examples. And, last but not least, it also works with non-differentiable objective functions, as that presented in this section. In the following, we briefly discuss the key implementation aspects of our evolutionary rule learner.

4.1. Evolutionary fuzzy rule learning

Our implementation roughly follows the "Pittsburgh-style" [44] approach described in [8]. As opposed to "Michigan-style" [18] approaches, each candidate solution represents a complete rule base (instead of cooperating individual rules).

As selection operator we chose the tournament selection [16], i.e. we randomly choose two chromosomes from the pool and take the fitter of the two. This operator is more robust compared to fitness proportional approaches, as it does not depend on the scaling of the fitness function, and computationally cheaper than rank based selection, which implies sorting the chromosomes by fitness.

Good recombination operators should produce valid offsprings that preserve good partial solutions from their parents. In our case, reasonable building blocks are obviously the rules, or combinations of them. Usual crossover operators work on the one-dimensional gene string and hence cannot take the multi-dimensional position of the rules into account. We use a variant of the so-called *one-point ordered crossover* [8,22]. Intuitively, it splits the input space into two parts. It then exchanges the parents' rules in these subspaces. Thus, positional information is preserved.

The mutation operator changes structure as well as parameters of the fuzzy sets. Thus rules might be deleted or added, dimensions might be set to "don't care", and the fuzzy set parameters are overlayed with small Gaussian distributed noise. To still enable linguistic interpretation of the fuzzy sets after learning, a minimum requirement is that they are ordered. In that way we can assign the usual labels (e.g. *small*, *medium*, *large*) or mixtures of them (e.g. *small to medium*). We implemented a repair mechanism that checks the fuzzy sets of every dimension after crossover and mutation. If the constraints are violated, the fuzzy sets are appropriately shifted to restore them.

4.2. A fitness function based on the MDL principle

To measure the quality of a rule base for a given partially labeled dataset, we have to take several aspects into account: The error on the labeled examples, the fitting of the model to the unlabeled data, and, possibly, the complexity of the model. We based our fitness function on the minimum description length principle (MDL), which gives us the appealing opportunity to derive all three aspects in the same theoretic context [38].

The idea of MDL is that the data has to be transmitted from an (imaginary) sender to an (imaginary) receiver. Structure in the data can be used for more efficient codes that result in shorter messages. However, both sender and transmitter need to know the encoding scheme of the data. Thus the message is compound by first transmitting the coding scheme, and then the data, encoded using this scheme. Complex models need a longer coding scheme, as more free parameters have to be transmitted. However, the resulting data part of the message will usually be shorter. The model with the shortest overall message length is assumed to fit the data best and is chosen. MDL is equivalent to maximum likelihood for the estimation of models with a fixed number of free parameters, but additionally offers a possibility to compare objectively between models of different complexity.

To apply the MDL principle to our problem, we have to define an appropriate encoding scheme. The message consists of the following parts:

- *the rule base* (*the code*), e.g. the number of dimensions, the number of rules, the fuzzy sets used in a rule. This information is then used to encode
- *the data tuples* (*the data itself*), e.g. the index of the rule that is used to encode this tuple, the exact values of the tuple using a rule specific code, and the class labels.

According to Shannon's coding theorem, the lengths in bits of parts of a message are calculated as $-\log_2 p_i$, where p_i is the probability of an instantiation i within all possible alternatives over some appropriately chosen probability distribution. To illustrate this, we present the equations for the message length for data tuples below.

If the t-norm $\top_{\text{prod}}$ is used and some restrictions are placed on the fuzzy sets and rule weights, (neuro-) fuzzy classification systems can be shown to perform the same calculations as Naive Bayes classifiers [33]. We can use this interpretation of the fuzzy membership values as probability densities to transmit the tuple coordinates. Let x be a tuple, r the rule which is used to encode it, i.e. the rule with the highest activation, and w_r the appropriate normalizing rule weight. Then the part of the message that transmits the coordinates of the d dimensions of a tuple has a length of

$$l(\mathbf{x}) = -\log_2 w_r - \sum_{i=1}^{d} \log_2 \mu_{r,i}(x_i).$$

This part measures cluster dispersion. It is used for encoding both, labeled and unlabeled tuples. The closer a tuple lies to the center of the membership functions of its rule (i.e. the better the rule is adapted to the data), the higher the probabilities and thus the shorter the code length is.

Additionally, we have to transmit the class labels of the tuples. We associate each rule with the majority class of the tuples covered by it. Thus we can use the (already transmitted) information which rule is used to encode a tuple. However, there might be tuples of several classes covered by one rule. Therefore, we have also to transmit the exceptions from the rule. We have to encode the class distribution in a rule and the explicit class labels for the tuples. Let X_i denote the tuples covered by a rule i, and within those let X_{ij} denote the tuples of class j. Let c be the number of classes in the dataset. In that part of the message the X_i^u unlabeled points play a role. We consider them as belonging to the majority class $j_{\max}$ in that rule and thus transmit this class (the most probable class yields the shortest message length). For the encoding of the class information for all tuples covered by one rule i we thus get a length of

$$l(r) = \log_2 \frac{(|X_i| + c - 1)!}{|X_i|!(c-1)!} + \log_2 \frac{|X_i|!}{(\prod_{j=1, j\neq j_{\max}}^{c} |X_{ij}|!)|X_{ij_{\max}} \cup X_i^u|!}.$$

Misclassified labeled tuples make that part of the message longer, as the probability distribution of a rule becomes more heterogeneous, and thus this part of the measure quantifies cluster impurity.

One problem of this measure is that the relation of the lengths of the code part and the data part of the message depends on the number of tuples and the number of dimensions. It can therefore be important to weight the individual parts of the message. This can be interpreted as adjusting the precision of the transmitted tuple coordinates.

Figs. 5 and 6 show the result of applying our proposed algorithm to the artificial datasets. Although these datasets were challenging for the other semi-supervised algorithms, and in spite of the restrictions that we imposed on the fuzzy sets to maintain linguistic interpretability, the algorithm performs almost perfect.

In the next section, we demonstrate applicability of our approach to a real world problem.

5. Filtering object primitives in image analysis

This application considered in the following has been studied in a cooperation with the *Research Institute for Optronics and Pattern Recognition* (FGAN/FOM) in Ettlingen/Germany. The focus of this project was the analysis of aerial images to extract man-made structures, like, for instance, airfields. Results of the cooperation are described in [23,24]. In these publications we had a different view on the problem, and applied pure supervised learning. However, as we will explain, it might be more appropriate to consider the task as semi-supervised.

5.1. Application domain

The automatic identification of man-made objects in remotely sensed images is still a challenging task. In the framework of structural analysis of complex scenes a blackboard-based production system (BPI) has been developed at FGAN/FOM [30]. In this system transformations of the simple objects extracted from SAR (synthetic aperture radar) images into more complex objects are controlled by production rules. A production net proceeds stepwise according to a model and produces intermediate results with

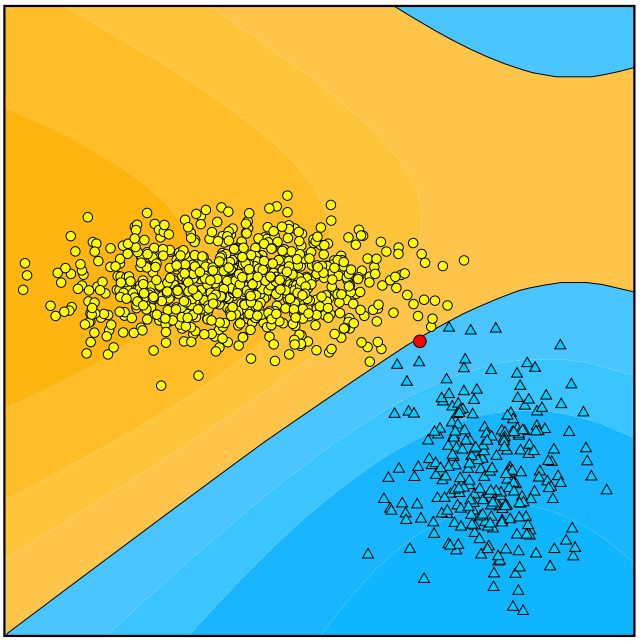

Fig. 5. Experimental result on the first artificial dataset (0.1% errors).

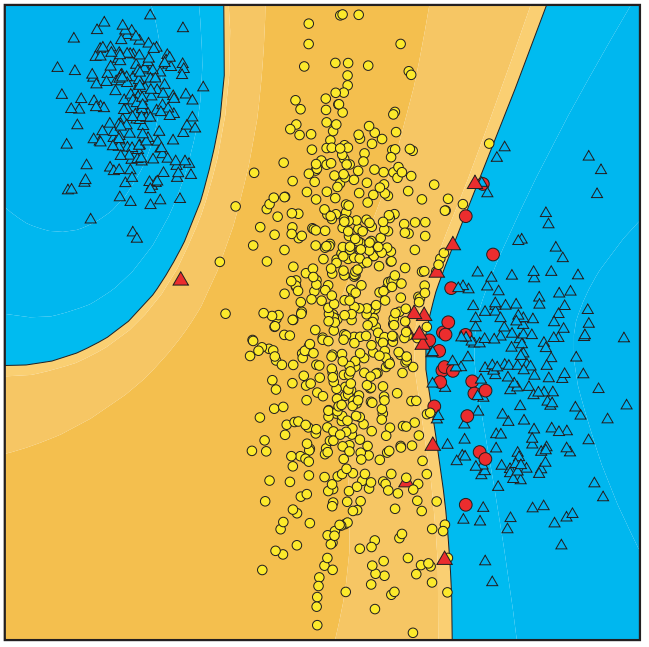

Fig. 6. Experimental result on the second artificial dataset (3.4% errors).

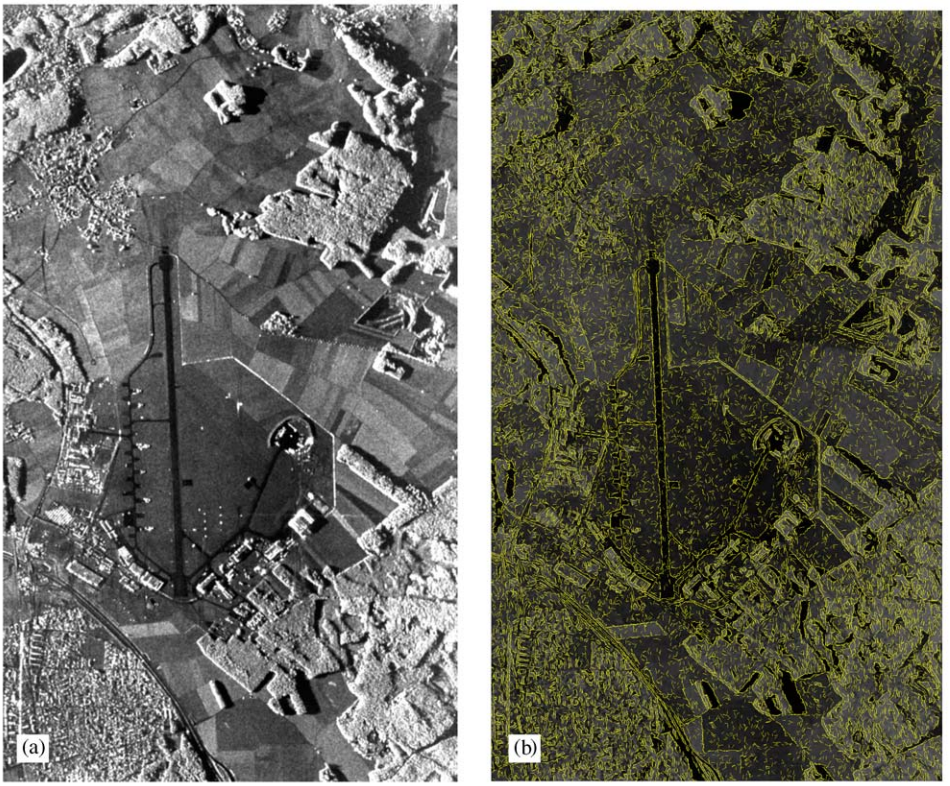

Fig. 7. (a) Example SAR image (b) 37,659 line segments extracted from the SAR image by Burns' edge detector.

an increasing degree of abstraction [39,40]. For instance, the abstraction degrees for the construction of a runway are

edges $\Rightarrow$ **lines** $\Rightarrow$ **long-lines** $\Rightarrow$ **parallel-lines** $\Rightarrow$ **runways**.

The image analysis is based on line segments as object primitives, which is typical for the extraction of man-made objects. Fig. 7a shows a typical SAR image of an airfield. The result of gradient-based edge detection [4] applied to the SAR image is shown in Fig. 7b. As the resolution of the images is rather high, the edge detector extracts more than 37,000 edges on this image, which have to be considered during the structural analysis. Although only a fraction of the lines are used to construct, e.g., the runway, the analyzing system has to take all of the lines into account. Unfortunately, time consumption is typically at least $O(n^2)$. Although the system is usually successful in extracting the desired structures from single images, the runtimes are too slow to process image data for larger areas. The idea was that the production process could significantly be sped up if only the most promising primitive objects are identified and the analysis is started with them.

[4] The used edge extraction algorithm was proposed in [7]. Especially in noisy images, this operator has a tendency of extracting high numbers of short line segments.

The idea was to extract features from the image that describe the primitive objects and allow to train a classifier that decides which lines can be discarded. Experiments showed that in the case of line primitives the regions next to the lines bear useful information. Therefore, rectangular windows adjacent to the lines are constructed. The gradient across the edge is used to define the line direction and to uniquely distinguish between left and right window. For each line segment a set of statistical (e.g. mean, standard deviation) and textural features (e.g. energy, entropy) is calculated from the gray values in the region. From this set of features, we chose the eight most important features.

The initial idea in our project was to apply the analysis process for a number of images on the complete set of object primitives. The resulting extracted runways are used to divide the lines into those that were used for the construction of complex structures (the *positive* class) and those that were not (the *negative* class). We tried to learn to differentiate between the classes with a number of classifier approaches. However, it turned out that the problem can hardly be solved by most classifiers without modifications. The classes were extremely unbalanced. For the shown image, only 20 lines were used to build the runway, and are thus used as positive examples. Moreover, the classes were strongly overlapping, and thus almost any classifier approach will simply predict the majority class for any input, because this leads to an extremely low error rate of $\frac{20}{37,659} \approx 0.05\%$. Although it is seemingly perfect, this result is obviously completely useless, as it filters out all object primitives, and thus hinders any object recognition.

We concluded that a classifier has to take into account the special semantics of the task. Misclassifications of positive and negative class have to be treated differently. As a matter of fact, every missed positive can turn out to be very expensive. Too many false negatives [5] can completely prevent the correct recognition of objects, whereas false positives [6] lead 'only' to considerably longer execution times. We considered this asymmetry in a misclassification cost matrix. This matrix was integrated into the NEFCLASS rule learning and pruning algorithms, which allowed successful application of NEFCLASS on this task. Although the classification was not perfect, in most cases all (or at least enough) relevant object primitives were classified as positive, while the total number of line segments to be processed was significantly reduced [23].

5.2. Semi-supervised line filtering

Although the obtained error rates seemed to be rather high for a classification problem, the image processing experts were quite content. [7] It turned out that they were actually glad about some of the false positives: the edges of taxiways, parking lots, or roads—having characteristics very similar to those of runway edges—are also needed in later processing stages. We concluded that the classification problem as formulated by the experts was actually ill-posed: although these additional object primitives should in fact be positive examples, they appear as negative examples in the training dataset. This makes the learning task harder for the classifier, as it—in spite of the included asymmetric misclassification costs—still tries to separate positives from negatives.

---

[5] False negatives, i.e. positives classified as negatives, refer to runway primitives that are discarded.

[6] False positives, i.e. negatives classified as positives, result in superfluous primitives that have to be considered in the production process.

[7] On some images about 50% of the examples were classified as positive, and thus the (unweighted) error rate was about the same. However, this still means halving the number of the object primitives.

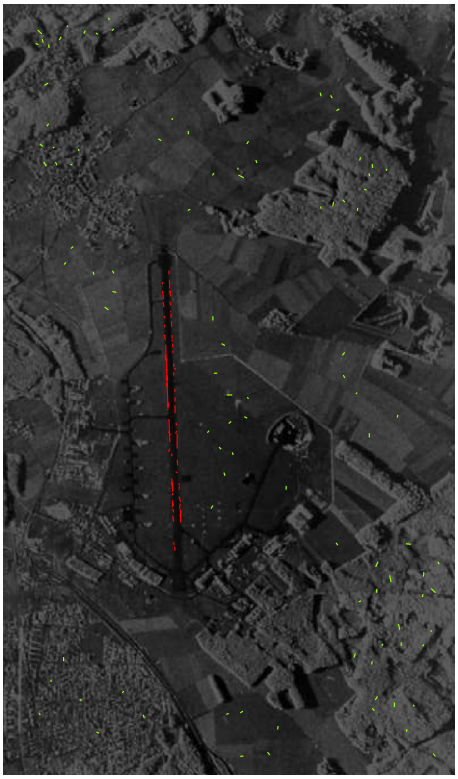

Fig. 8. Manually labeled examples: 57 positive examples (runway line segments, in red), 98 negative examples (in green).

We thus think that it might be more appropriate to understand the problem as one of semi-supervised learning. The runway segments *plus* similar primitives like taxiways make up the positive examples, though only a part of them is labeled. Additionally, we manually label some negative examples, i.e. line segments that should be discarded during the analysis process. A supervised learning algorithm is then applied to the corpus of unlabeled data plus the few labeled examples. The algorithm should learn to separate the labeled examples, and additionally locate the decision boundary according to the structure of the unlabeled examples.

Fig. 8 shows the line segments that we manually labeled as positive and negative examples, respectively. Altogether, we labeled only 155 examples (out of 37, 659). Obviously, the dataset has the mentioned challenges for semi-supervised learning: extremely few labeled examples, and manually given labels that are probably neither prototypical for their respective classes, nor do they cover the complete spectrum of occurring edge segment characteristics. We applied our semi-supervised evolutionary fuzzy rule learner to this problem.

As the number of unlabeled examples exceeds the number of labeled examples by far, we have to balance the influences of the parts of the quality measure (cf. Section 4.2). For the presented result, we set the weight w_{labels} of the impurity measure to ten times the weight w_{base} of the rule base length, and to 2000 times the weight w_{tuples} of the dispersion measure, which brings the individual lengths to roughly the same ranges. On this dataset, ssMDL was not too sensitive to changes of the weights. Fig. 9a

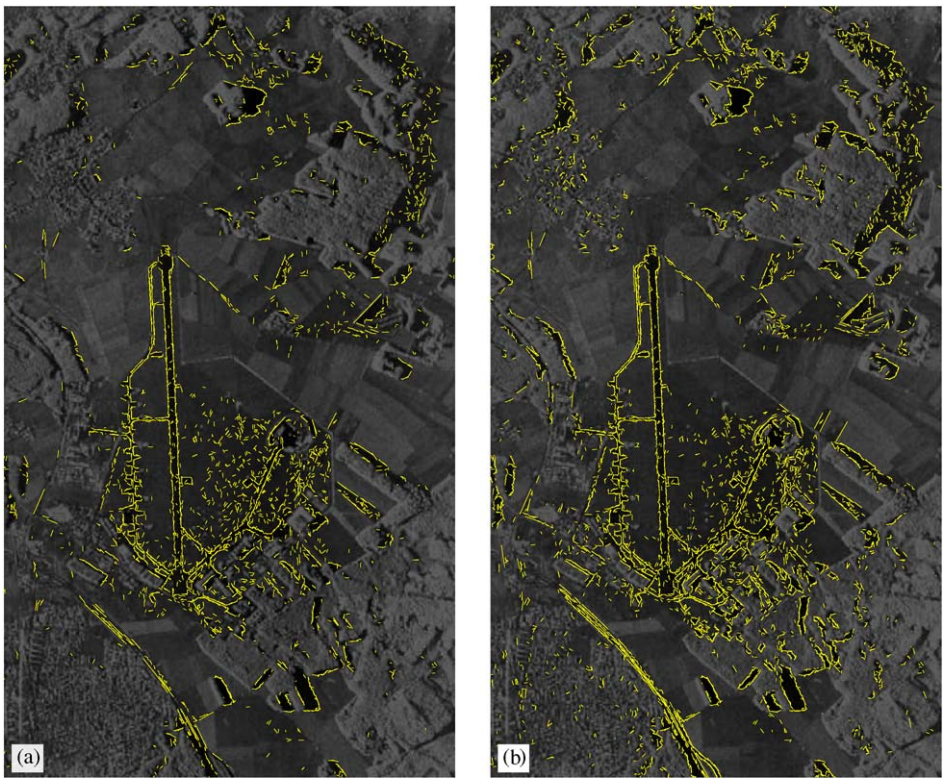

Fig. 9. (a) 3878 lines classified as positive by semi-supervised ssMDL; (b) 6994 by pure supervised learning.

depicts the 3878 line segments that ssMDL classified as positive. On the labeled examples, it produces 5 errors (3.2%). It can be seen that the line segments necessary to construct runway, taxiways, and traffic system have been successfully identified. The number of potentially relevant object primitives is reduced to 10.3%, which leads to an significantly decreased processing time.

As a comparison, we alternatively treated the problem by pure supervised learning. To reduce the influence of the learning algorithm, we again applied ssMDL to the problem. However, we presented only the 155 labeled examples for learning, and still used a higher weight for the impurity measure. In this way, ssMDL works as a pure supervised rule learner. It produces the same number of misclassification, i.e. 5% or 3.2%. However, it cannot take the unlabeled data into account for locating the decision boundary. On this image, it extracts 6994 lines as possible runway or traffic system primitives, only a reduction to 18.6%. The resulting positive lines are shown in Fig. 9b. Obviously, the segments extracted by this classifier are also sufficient to construct the desired objects. Although the results of the two classifiers look rather similar, it can clearly be seen that there are many more spurious positive edge segments in Fig. 9b.

It should be noted that we presented the results of reclassification, i.e. we applied the classifier to the data it was trained on. Error rates estimated from reclassification are usually optimistically biased, and thus less credible. However, we suppose that our application is less prone to this optimistic bias. The main problem of reclassification is that of overfitting: a classifier could ‘memorize’ the examples’ labels

instead of revealing the underlying regularities. However, most of the data in our application do not have a class label. In case of the supervised application of ssMDL, the unlabeled data were not used at all. In case of semi-supervised learning, only their positional information is used. Thus, the results are still meaningful (apart from the certainly optimistically estimated 3.2% errors on the labeled data). Another typical problem of semi-supervised learning can also be seen: since the labels of the unlabeled data are indeed unknown, we can hardly estimate the overall error rate. However, in image analysis applications like ours, we can often at least visually inspect the results.

The alternative interpretation of the line filtering problem as a semi-supervised learning task yields a feasible solution, comparable to the results of our previous approach using asymmetric error costs [23,24]. The semi-supervised learning from labeled and unlabeled data resulted in a much more selective classifier than the pure supervised approach. However, we should note that we can only visually assess the classification results, as we do not have the true class labels.

6. Conclusions

The state of the art in knowledge discovery techniques is dominated by supervised approaches. However, in many current real-world problems the assignment of labels for all objects is a severe problem and thus calls for semi-supervised methods.

We presented an evolutionary algorithm to induce fuzzy rules that exploits labeled and unlabeled training data. We compared it to existing fuzzy semi-supervised algorithms. Our MDL-based approach outperformed the other semi-supervised algorithms on the artificial example datasets, where a certain flexibility was required to model the distribution and where the given labeled examples were less representative. Additionally, we have shown the applicability of our semi-supervised rule learner on a real-world problem.

It remains an interesting open question to investigate, if it is possible to decide in advance, whether semi-supervised learning will be promising for a given dataset, i.e. whether the data have suitable characteristics.

References

[1] A. Amar, N.T. Labzour, A.M. Bensaid, Semi-supervised hierarchical clustering algorithms, in: Proc. 6th Scand. Conf. on Artificial Intelligence (SCAI'97), Helsinki, Finland, 1997, pp. 232–239.
[2] K.P. Bennett, A. Demiriz, Semi-supervised support vector machines, in: D.A. Cohn, M.S. Kearns, S.A. Solla (Eds.), Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA, 1998, pp. 368–374.
[3] A.M. Bensaid, J.C. Bezdek, Semi-supervised point prototype clustering, Pattern Recognition Artif. Intell. 12 (2) (1998) 625–643.
[4] A.M. Bensaid, L.O. Hall, J.C. Bezdek, L.P. Clarke, Partially supervised clustering for image segmentation, Pattern Recognition 29 (2) (1996) 859–871.
[5] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.
[6] C.L. Blake, C.J. Merz, UCI repository of machine learning databases, 1998.
[7] J. Burns, A. Hanson, E. Riseman, Extracting straight lines, IEEE Trans. Pattern Anal. Mach. Intell. 8 (1986) 425–455.
[8] B. Carse, T.C. Fogarty, A. Munro, Evolving fuzzy rule based controllers using genetic algorithms, Fuzzy Sets and Systems 80 (1996) 273–293.
[9] O. Cordón, F. Herrera, P. Villar, Generating the knowledge base of a fuzzy rule-based system by the genetic learning of data base, IEEE Trans. Fuzzy Systems 2001.
[10] R. Dara, S.C. Kremer, D.A. Stacey, Clustering unlabeled data with SOMs improves classification of labeled real-world data, in: Proc. IEEE Internat. Conf. on Fuzzy Systems (FUZZ-IEEE'02), Honolulu, HI, 2002, published on CD-ROM.

[11] A. Demiriz, K.P. Bennett, M.J. Embrechts, A genetic algorithm approach for semi-supervised clustering, J. Smart Eng. System Design 4 (2002) 35–44.
[12] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, AAAI Press/MIT Press, Cambridge, MA, 1996.
[13] G. Fung, O.L. Mangasarian, Semi-supervised support vector machines for unlabeled data classification, Tech. Rep., Data Mining Institute, 1999.
[14] B. Gabrys, A. Bargiela, General fuzzy min–max neural network for clustering and classification, IEEE Trans. Neural Networks 11 (2) (2000) 769–783.
[15] B. Gabrys, L. Petrakieva, Combining labelled and unlabelled data in the design of pattern classification systems, in: Proc. EUNITE'2002, 2002, pp. 441–449.
[16] D.E. Goldberg, K. Deb, A comparative analysis of selection schemes used in genetic algorithms, in: G. Rawlins (Ed.), Foundations of Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1991.
[17] E.E. Gustafson, W. Kessel, Fuzzy clustering with a fuzzy covariance matrix, in: Proc. IEEE Conf. on Decision and Control, San Diego, IEEE Press, Piscataway, NJ, 1979, pp. 761–766.
[18] J.H. Holland, J.S. Reitman, Cognitive systems based on adaptive algorithms, in: D.A. Waterman, F. Hayes-Roth (Eds.), Pattern-Directed Inference Systems, Academic Press, 1978, pp. 313–329.
[19] F. Höppner, F. Klawonn, R. Kruse, T. Runkler, Fuzzy Cluster Analysis, Wiley, Chichester, 1999.
[20] J. Jäkel, L. Gröll, R. Mikut, Automatic generation and evaluation of interpretable rule bases for fuzzy systems, in: Proc. CIMCA'99, IOS Press, Amsterdam, 1999, pp. 192–197.
[21] F. Klawonn, R. Kruse, Constructing a fuzzy controller from data, Fuzzy Sets and Systems 85 (1997) 177–193.
[22] A. Klose, R. Kruse, Enabling neuro-fuzzy classification to learn from partially labelled data, in: Proc. IEEE Internat. Conf. on Fuzzy Systems, (FUZZ-IEEE'02), Honolulu, HI, IEEE Press, Piscataway, NJ, 2002, published on CD-ROM.
[23] A. Klose, R. Kruse, K. Schulz, U. Thönnessen, Controlling asymmetric errors in neuro-fuzzy classification, in: Proc. ACM SAC'00, ACM Press, 2000, pp. 505–509.
[24] A. Klose, D. Nauck, K. Schulz, U. Thönnessen, Learning a neuro-fuzzy classifier from unbalanced data in a machine vision domain, in: Proc. 6th Internat. Workshop on Fuzzy-Neuro Systems FNS'99, Leipziger Universitätsverlag, Leipzig, 1999, pp. 23–32.
[25] A. Klose, A. Nürnberger, D. Nauck, R. Kruse, Data mining with neuro-fuzzy models, in: A. Kandel, M. Last, H. Bunke (Eds.), Data Mining and Computational Intelligence, Physica, 2001, pp. 1–36.
[26] R. Kothari, V. Jain, Learning from labeled and unlabeled data, in: Proc. IEEE Internat. Conf. on Fuzzy Systems (FUZZ-IEEE'02), Honolulu, HI, 2002, published on CD-ROM.
[27] U.V. Kulkarni, D.D. Doye, T.R. Sontakke, General fuzzy hypersphere neural network, in: Proc. IEEE Internat. Conf. on Fuzzy Systems (FUZZ-IEEE'02), Honolulu, HI, 2002, published on CD-ROM.
[28] T. Labzour, A. Bensaid, J. Bezdek, Improved semi-supervised point-prototype clustering algorithms, in: Proc. Internat. Conf. on Fuzzy Systems, 1998, pp. 1383–1387.
[29] C. Lanquillon, Enhancing text classification to improve information filtering, Ph.D. Thesis, Otto-von-Guericke-Universität Magdeburg, Germany, 2001.
[30] K. Lütjen, BPI: Ein Blackboard-basiertes Produktionssystem für die automatische Bildauswertung, in: G. Hartmann (Ed.), Mustererkennung 1986, 8. DAGM-Symposium, Springer, Berlin, 1986, pp. 164–168.
[31] D.J. Miller, H.S. Uyar, A mixture of experts classifier with learning based on both labelled and unlabelled data, in: Advances in Neural Information Processing Systems (NIPS-9), MIT Press, Cambridge, MA, 1997, pp. 571–577.
[32] K. Nigam, A. McCallum, S. Thrun, T. Mitchell, Text classification from labeled and unlabeled documents using EM, Mach. Learning 39 (2/3) (2000) 103–134.
[33] A. Nürnberger, C. Borgelt, A. Klose, Improving naive bayes classifiers using neuro-fuzzy learning, in: Proc. 6th Internat. Conf. on Neural Information Processing (ICONIP'99), Perth, Australia, 1999, pp. 154–159.
[34] J.-M. Park, H.-D. Yae, Analysis of active feature selection in optic nerve data using labeled fuzzy c-means clustering, in: Proc. IEEE Internat. Conf. on Fuzzy Systems (FUZZ-IEEE'02), Honolulu, HI, 2002, published on CD-ROM.
[35] W. Pedrycz, Algorithms of fuzzy clustering with partial supervision, Pattern Recognition Lett. 3 (1985) 13–20.
[36] W. Pedrycz, J. Waletzky, Fuzzy clustering with partial supervision, IEEE Trans. System Man Cybernet. Part B: Cybernet. 27 (2) (1997) 787–795.
[37] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, CA, 1993.

[38] J. Rissanen, A universal prior for integers and estimation by minimum description length, Ann. Statist. 11 (1983) 416–431.
[39] R. Schärf, H. Schwan, U. Thönnessen, Reconnaissance in SAR images, in: Proc. European Conf. on Synthetic Aperture Radar,1998, pp. 343–346.
[40] H. Schwan, R. Schärf, U. Thönnessen, Reconnaissance of extended targets in SAR image data, in: Proc. European Symp. on Remote Sensing, Barcelona, 1998, pp. 164–171.
[41] P.K. Simpson, Fuzzy min–max neural networks—part 1: classification, IEEE Trans. Neural Networks 3 (2) (1992) 776–786.
[42] P.K. Simpson, Fuzzy min–max neural networks—part 2: clustering, IEEE Trans. Fuzzy Systems 1 (2) (1993) 32–45.
[43] A. Skabar, Augmenting supervised neural classifier training using a corpus of unlabeled data, in: Proc. 25th German Conf. on Artificial Intelligence (KI-2002), Aachen, Germany, 2002, pp. 174–185.
[44] S.F. Smith, A learning system based on genetic adaptive algorithms, Ph.D. Thesis, Department of Computer Science, University of Pittsburgh, 1980.
[45] H. Timm, Fuzzy-Clusteranalyse: Methoden zur Exploration von Daten mit fehlenden Werten sowie klassifizierten Daten, Ph.D. Thesis, Otto-von-Guericke-Universität Magdeburg, Germany, 2002.
[46] A. Verikas, A. Gelzinis, K. Malmquist, Using unlabeled data for learning classification problems, in: L.C. Jain, J. Kacprzyk (Eds.), New Learning Paradigms in Soft Computing, Physica, Heidelberg, 2002, pp. 368–403.
[47] Y. Wu, Q. Tian, T.S. Huang, Discriminant-EM algorithm with application to image retrieval, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2000, pp. 222–227.
[48] Y. Yuan, M.J. Shaw, Induction of fuzzy decision trees, Fuzzy Sets and Systems 69 (2) (1995) 125–139.