Building Hybrid Fuzzy Classifier Trees by Additive/Subtractive Composition of Sets

Arne-Jens Hempel[1], Holger Hähnel[1], and Gernot Herbst[2]

[1] Technische Universität Chemnitz, Chemnitz, Germany
[2] Siemens AG, Chemnitz, Germany

Abstract. Especially for one-class classification problems, an accurate model of the class is necessary. Since the shape of a class might be arbitrarily complex, it is hard to choose an approach that is generic enough to cope with the variety of shapes, while delivering an interpretable model that remains as simple as possible and thus applicable in practice. In this article, this problem is tackled by combining convex building blocks both additively and subtractively in a tree-like structure. The convex building blocks are represented by multivariate membership functions that aggregate the respective parts of the learning data. During the learning process, proven methods from support vector machines and cluster analysis are employed in order to optimally find the structure of the tree. Several academic examples demonstrate the viability of the approach.

1 Introduction

Besides traditional classification techniques that try to distinguish between two or more classes, the demand for one-class classification methods has recently been growing [1]. In real world applications, this relevance of unary classification can be caused by two facts. First, for example, when modeling the class of "interesting products" for the customer of an online store, the learning data include no counterexamples, by which a "non-interesting" class could be determined. Secondly, the structure of counterexamples could be too extensive to be modeled appropriately, which holds e. g. for the decision of an airbag deployment in a vehicle. One approach here would be to decide on the basis of an accurately described "accident" class in a one-class classification task. This might be more apposite than discriminating between the "accident" class and a "non-accident" class, which may have a highly intricate structure making it difficult to model.

Naturally, a classification approach based on (crisp or fuzzy) sets that model a phenomenon has a "local" character and is thus especially suitable for one-class problems. This holds in contrast to discriminatory methods like linear discriminant analysis or (standard) support vector machines (SVMs), which, by construction, allow "global" decisions beneficial for two- or multi-class tasks.

The amenities of a fuzzy classification approach lie in its ability to cope with practically occuring problems such as noisy data, transitional effects, drifting and evolving classes, or overlaps in the case of two- and multi-class problems [2,3]. The high interpretability of fuzzy classification has often been stressed.

A. Laurent et al. (Eds.): IPMU 2014, Part II, CCIS 443, pp. 516–525, 2014.

It provides the opportunity to incorporate expert knowledge into the classifier [4]. Picking up the above example of an airbag deployment, interpretability might be very important when evaluating data of a car accident by an assessor.

The setup of an accurate and at the same time simple and applicable model can be challenging if the shape of the class that is to be described is non-convex or even arbitrarily complex. Such phenomena can be found in applications such as banknote authentication and machine diagnosis [5,6]. The root idea addressing this problem has been given by [7,8] and others. It revolves around the additive combination of convex fuzzy sets, such as fuzzy partitions. Besides that, a subtractive composition using so-called complementary classes has been proposed [6]. We recommend a combination of these two approaches resulting in a tree-like model, which we refer to as hybrid fuzzy classifier tree (FCT).

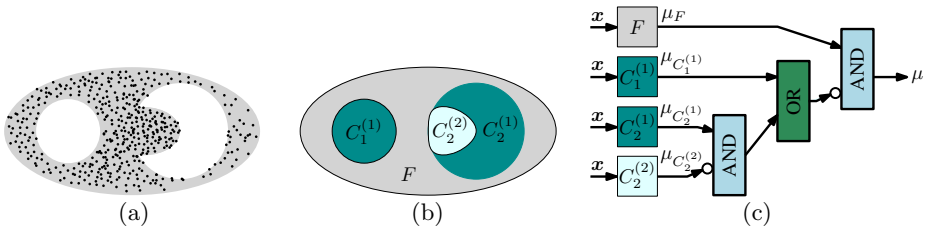

Fig. 1. Additive/subtractive classifier tree (c) built from the least possible number of convex sets (b) for two-dimensional data forming a "two-hole" shape (a)

In Fig. 1, an example leading to such a classifier tree is sketched. Aim of this article is to provide an algorithm which, when given a set of learning data as in Fig. 1a, builds up a tree as in Fig. 1c. As leaf nodes, it employs multidimensional membership functions representing convex basic building blocks similar to the sets given in Fig. 1b. This can be understood as a geometrical viewpoint for setting up the classifier. In contrast, the similar approach of fuzzy pattern trees [9] provides a rather logical description by using one-dimensional fuzzy terms as leaf nodes which form partitions of the corresponding attribute's domains.

2 Towards Hybrid Fuzzy Classifier Trees

Our aim is to build an FCT from a given set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \subset \mathbb{R}^M$ of N learning objects, each with M features. In the unary case, the classifier shall return one truth value $\mu(\boldsymbol{x}) \in [0, 1]$, such that the classification of a test datum $\boldsymbol{x} \in \mathbb{R}^M$ appearing in a region not supported by learning data results in a low degree of membership indicating that $\boldsymbol{x}$ does not belong to the class. Moreover, the procedure can be used for two- or multi-class problems, simply by independently building a tree for each class and comparing the truth values.

Let us assume that we possess a (fuzzy) model and learning method for sets of objects forming a convex shape in their feature space. How could we use them to cope with non-convex data? A straightforward approach would try to break the data set apart into convex subsets, e. g. using partitioning or segmentation, and

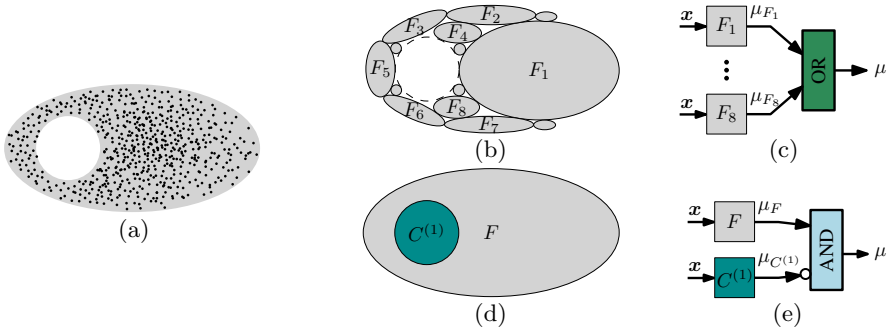

Fig. 2. Building a classifier tree for two-dimensional data forming a non-convex "one-hole" shape (a). Additive approach: (b), (c) and subtractive approach: (d), (e).

subsequently learn convex models F_i for each of the subsets. The overall classifier for this non-convex class would then be built from a disjunctive combination ("F_1 OR F_2 OR ...") of these convex building blocks, cf. Fig. 2a to 2c.

In contrast, the approach from [6] starts with a convex model F covering the whole data set. Afterwards, it subtracts the convex part $C^{(1)}$ in order to fit the model to the non-convex shape ("F AND NOT $C^{(1)}$"), cf. Fig. 2d and 2e. If a non-convex part had to be subtracted, it could be modeled recursively by a subtraction of convex elements ("$C^{(1)}$ AND NOT $C^{(2)}$ ..."). $C^{(l)}$ are called complementary classes with l indicating the level of complementation (cf. Sect. 5).

For arbitrarily complex shapes, one can expect to achieve more compact models (with a smaller number of building blocks) by combining both approaches. A classifier tree which is built up using the subtractive "AND NOT" approach as well as the additive "OR" is shown in Fig. 3. The subscripts in $C_1^{(1)}$ and $C_2^{(1)}$ enumerate the additively combined complementary classes of level 1.

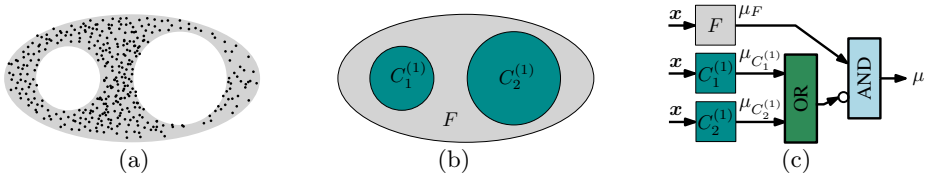

Fig. 3. Hybrid additive/subtractive classifier tree (c) for a "two-hole" shape (a)

In this paper, we start with one model F as a convex "hull" of the data and remove the parts not supported by learning data, referred to as object-unsupported class space C. C itself may be built up additively and/or subtractively from convex elements in a tree-like structure. In this way, one always gets a rough result in one step, i. e. an instance outside of F can never belong to the considered class. Subsequently, this statement is refined using the convex elements of C.

There are various options for both the structure of the tree and the type of membership functions (MFs) modelling the convex sets that are to be combined.

For the latter, we use a specific parametric MF of potential type (Sect. 3). In order to learn these models also for parts of the object-unsupported class space C, we propose to fill up C by an artificially generated set of so-called complementary objects (Sect. 4). Finally, the algorithm for setting up a model for C (Sect. 5) employs a combination of learning of MFs, cluster analysis (to find subsets for additive "OR" combinations), and the recursive "AND NOT" approach from [6].

3 Convex Building Blocks: Fuzzy Pattern Classes

Since we might need several (but as few as possible) convex building blocks to create FCTs, they should be well formalized and possess a comprehensible aggregation process when dealing with high-dimensional feature spaces. It needs to be emphasized that, in principle, any convex fuzzy set could serve as a component for building FCTs. Nevertheless, we propose the usage of a specific multivariate parametric fuzzy set. Besides the above-named properties, it features additional advantages, such as the treatment of asymmetric data distributions. We refer to this set as fuzzy pattern class (FPC), a concept which was introduced by Bocklisch as a generalisation of Aizerman's potential function [10]. It has already been applied for the modeling of traffic flows [11], medical diagnosis [12], condition monitoring [6], and time series analysis [13]. The structure of the FPC approach even suits embedded implementations in industrial applications [14].

Purpose of this section is to give a brief review of the FPC concept together with its main properties. A comprehensive description can be found in [10].

3.1 Definition and Learning of Fuzzy Pattern Classes

An FPC is a multivariate parametric fuzzy set with the membership function

$$\mu(\boldsymbol{x}) = \frac{a}{1 + \frac{1}{M}\sum_{i=1}^{M}\left(\frac{1}{b_{i,\mathrm{l/r}}} - 1\right) \cdot \left|\frac{x_i - u_i}{c_{i,\mathrm{l/r}}}\right|^{d_{i,\mathrm{l/r}}}} \,. \tag{1}$$

It derives from the intersection of M univariate FPC basis functions of the same type, each defined by a set of well interpretable parameters.[1] These include the location parameter $u_i \in \mathbb{R}$, left and right class borders $c_{i,\mathrm{l/r}} \in \mathbb{R}^+$ with their corresponding border memberships $b_{i,\mathrm{l/r}} \in [0,1]$, and the specifiers for the class fuzziness $d_{i,\mathrm{l/r}} \in [2,\infty)$. The parameters' impact on a univariate basis function can be understood by means of Fig. 4a. The quantity a can serve as a weight parameter for prioritising certain blocks of the classifier tree while decreasing the influence of others. Due to their interpretability, all parameters enable the incorporation of expert information in terms of a knowledge-based system.

An FPC is optimally fit to its supporting data by means of a translation relative to the class representative $\boldsymbol{u} = (u_1, \ldots, u_M)^\top \in \mathbb{R}^M$ and a rotation

[1] The intersection leading to (1) is conducted by a compensatory Hamacher operator, preserving the function concept, parameters, and properties of the basis function [10].

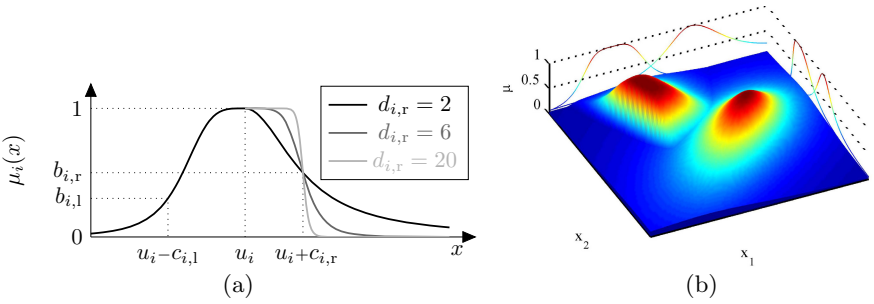

Fig. 4. 1-D (a) and 2-D (b) versions of the MF in (1) including rotation

described by the matrix $T \in \mathbb{R}^{M \times M}$ in the form $\boldsymbol{x'} = T(\boldsymbol{x} - \boldsymbol{u})$. Accordingly, the membership value for a rotated and translated FPC is given by $\mu(\boldsymbol{x}) = \mu'(\boldsymbol{x'}) = \mu'(T(\boldsymbol{x} - \boldsymbol{u}))$, where μ' has the same structure as μ in (1), though with $u_i = 0$, $i = 1, \ldots, M$. Two examples of two-dimensional FPCs are illustrated in Fig. 4b along with their univariate basis functions for each dimension.

The determination of an FPC's parameters on the basis of given learning data is described in detail in [10]. In recent works, the parameterisation has been further developed, notably with regard to its robustness [15].

3.2 Fuzzy Pattern Class Properties

If a class of N learning objects is modeled by only one FPC, which is defined by $8M$ parameters, the approach will provide a data compression ratio of $\frac{N}{8}$. The ratio scales down linearly with a growing number B of building blocks in the tree. In our approach, the aim of a small value of B is achieved not only by the chosen tree structure, but also by the fact that FPCs already can be tuned quite flexibly to the properties of a data set compared to simpler choices of MFs.

4 Exploration of the Object-Unsupported Class Space

The convexity of an FPC makes it a proper description for classes with a convex data-inherent structure. In contrast, a convex set F would obviously not provide a tight description for a non-convex class like in Fig. 1. But how can we decide in practice whether a convex model is sufficiently accurate—only on the basis of a data set? An approach that has been proven to solve this problem efficiently for low- and high-dimensional data sets incorporates one-class SVMs [6]. This graph-based exploration scheme distributes complementary objects uniformly alongside the edges between "border objects", which are found in an optimal manner using SVMs. The algorithm assures that only edges within the object-unsupported class space are deployed. It also limits the number of complementary objects by $N - 1$ and thereby sets an upper limit for further computational costs.

The result of this procedure is depicted in Fig. 5 where it has been applied to an academic example featuring a shape of learning data similar to Fig. 1. Subsequently, complementary classes can be learned from the complementary objects. Since we employ the membership function proposed in Sect. 3 again, they are referred to as complementary fuzzy pattern classes (CFPCs).

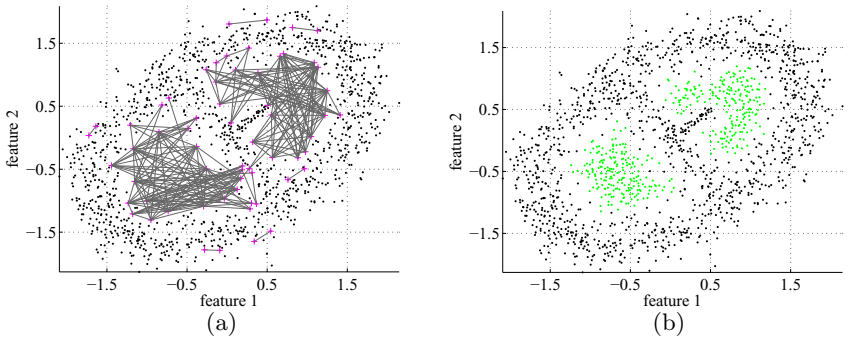

Fig. 5. (a) Learning/border objects and edges (b) learning/complementary objects

One may argue that the use of a one-class SVM sets a limit for the accuracy of the approach. Yet, we employ the SVM solely for generating complementary objects and return to the model set up based on (C)FPCs. Thus, the interpretability of the fuzzy classifier and its low complexity are retained (cf. Sect. 6). However, the accuracy can be controlled e. g. by tuning the SVM's kernel parameters.

5 Composing the Blocks and Building a Hybrid Tree

Given a set of learning data, we want to set up a shape-preserving model of a class via a fuzzy classifier tree where FPCs and CFPCs serve as basic building blocks. As mentioned, we set up an encircling fuzzy pattern class F based upon the given learning objects preliminarily (cf. Sect. 2 and Fig. 1). Obviously, F has to be combined with the "remainder" C of the classifier tree by a fuzzy-logical AND NOT. However, the locating of the CFPCs as well as their suitable interconnection for setting up C in form of a subtree is more intricate. The algorithm consists of five steps, which are processed as depicted in Fig. 6.

Step 1. *Exploration of the object-unsupported class space*
The first step is conducted by generating complementary objects according to Sect. 4. The level of complementation is set to the initial value $l = 1$.

Step 2. *Cluster analysis of complementary objects*
In order to ascertain whether there are separate groups of complementary objects, i. e. distinguishable object-unsupported partitions, we apply a density-based clustering scheme because of its property to find clusters independent of their underlying shape [16]. The clustering is governed by a distance parameter δ, whose value follows from the distribution of complementary objects in step 1.

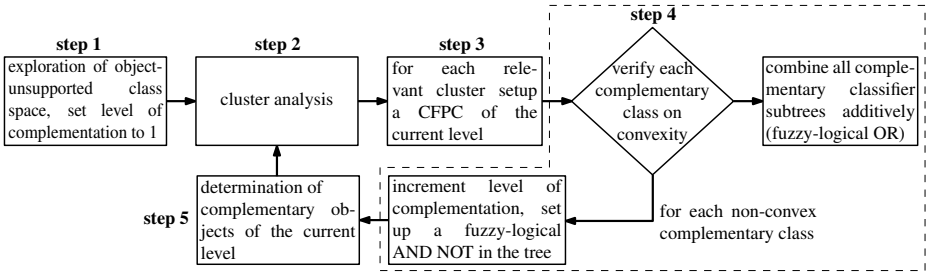

Fig. 6. Algorithm for setting up C as a subtree of complementary fuzzy pattern classes

Step 3. *Aggregation of complementary objects*
Relevant clusters of complementary objects are aggregated to complementary classes, see Sect. 3.1. Clusters are considered to be relevant if their cardinality exceeds the value $r \cdot N$, where $r \in [0, 1]$ is a task-specific parameter. For the example of Fig. 1, this results in the CFPCs $C_1^{(1)}$ and $C_2^{(1)}$.

Step 4. *Verification of convexity for complementary classes*
Due to the fact that the clusters of complementary objects may also be characterized by non-convex shapes, their convex CFPC models may be inadequate as well. For $l = 1$, the suitability of each CFPC is confirmed by a classification of the learning objects $\boldsymbol{x}_i$ applying (1), where $\mu = \mu_{\mathrm{CFPC}}$. If this classification yields low memberships $\mu_{\mathrm{CFPC}}(\boldsymbol{x}_i)$ for at least $(1 - r) \cdot N$ instances, the considered CFPC is assumed to be suitable. The description of the respective cluster is completed and the CFPC is branched off via a fuzzy-logical OR. This applies to $C_1^{(1)}$ in Fig. 1. On the contrary, if the classification of $\boldsymbol{x}_i$ results in high degrees of membership for at least $r \cdot N$ instances, the considered CFPC is expected to be inadequate (as $C_2^{(1)}$ in Fig. 1). The CFPC description itself has to be refined with one or more complementary classes of the next level, see step 5. Hence, the level of complementation is incremented. The respective CFPC is branched off via an OR and an additional AND NOT connective for its further refining. In Fig. 1, this corresponds to the combination "$C_2^{(1)}$ AND NOT $C_2^{(2)}$". However, a fuzzy description of $C_2^{(2)}$ in terms of (1) is not known at this step of the algorithm.

Generally, for $l > 1$, the verification of convexity is performed with the complementary objects of the preceding level (as defined in step 5).

Step 5. *Selection of higher-level complementary objects*
Each CFPC with an indication of non-convexity is refined separately but in the same manner, thus forming new branches in the next layer of the tree. Due to the fact that learning and complementary objects are mutually complementing each other, it follows that we can refrain from a further generation of complementary objects. For l even, the set of complementary objects is given by those learning objects with high degrees of membership to the currently refined CFPC of level $l - 1$ (cf. step 4). For l odd, this set is represented by the complementary objects, generated in step 1, exhibiting high memberships to the associated CFPC.

The selected complementary objects are fed back into the algorithm (step 2). After the cluster analysis, they form complementary classes of level l (step 3). Regarding the example from Fig. 1, the algorithm determines the description of $C_2^{(2)}$ and terminates with the verification of its convexity.

After the tree setup, all blocks (FPCs) are weighted via their parameter a in order to obtain normalized membership values $\mu(\boldsymbol{x}_i)$. For the calculation of the memberships, we apply the max operator as OR combination and the algebraic product with the natural negation for the fuzzy-logical AND NOT.

The process parameter r governs the convergence and the detailedness of the emerging classifier tree. The larger r the coarser the model (i. e. smaller B) and the faster the learning converges. The value $r \cdot N$ represents the least number of objects to form a building block. Thus, one can always choose r sufficiently large in order to achieve highly understandable and interpretable trees.[2] In the special case $r = 1$, the algorithm stops immediately resulting in the trivial tree defined upon F. Usually, r is set based on task-specific knowledge, e. g. in terms of the desired model complexity. The complexity of the algorithm itself scales quadratically with N and linearly with the dimensionality of the feature space.

6 Examples

We will now demonstrate the viability of the approach with the help of several academic examples being set up in the spirit of the introductory examples from Sect. 1 and Sect. 2. To this end, we will start with a simple distribution of learning data in a two-dimensional feature space, forming a convex shape. Subsequently, convex and non-convex parts of the shape are being taken out successively in order to resemble the shapes of Fig. 2a, Fig. 3a, and Fig. 1a.

First aim of this section is to visually confirm that the algorithm from Sect. 5 delivers a classifier tree with a minimal number B of convex sets. That is to say the resulting tree should exhibit the structure of Fig. 2c, 3c, and 1c for the examples from Fig. 2a, 3a, and 1a, respectively. It should not have any branches for the purely convex case. The second aim is to provide first findings for a comparison of the proposed tree with other one-class learners.

The learning data, the generated complementary objects, and the resulting MFs are shown in Fig. 7. The process parameter r has been set to 0.01. Obviously, the shape of each data distribution is captured very well. The resulting tree structures are not shown here since they are equivalent to Fig. 2c, 3c, and 1c.

For the example of Fig. 7d, a comparison with the purely subtractive, an additive (segmentation) tree approach, and a one-class SVM is given in Table 1. Considering only the tree approaches, the hybrid FCT is optimal w. r. t. B. Additionally, the model complexity (number of parameters) is almost one order of magnitude smaller than for a one-class SVM. In this setup, the SVM has been

[2] The understandability of the tree originates from a small number of blocks and their hierarchical arrangement using fuzzy-logical connectives. It is further fostered by the interpretation of a single block as a rule-based (sub)system (due to the used intersection operator) and the semantical parameters of its MF (cf. Sect. 3.1).

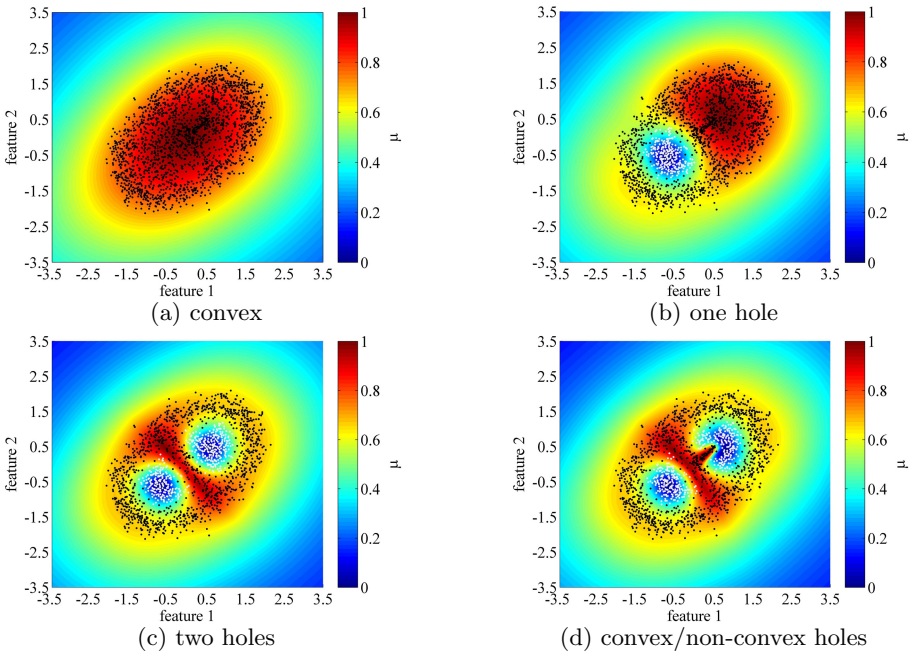

Fig. 7. Learning data (black), complementary objects (white) and resulting membership function for examples with increasing complexity. The classifier tree structure for (b), (c) and (d) can be found in Fig. 2c, 3c and 1c, respectively.

Table 1. Comparison of hybrid fuzzy classifier trees with other one-class approaches

method	hybrid tree	subtractive tree	additive tree	one-class SVM
#convex sets (B) / #SVs	4	5	6	133
#parameters	64	80	96	400
run time	2.4s	3.1s	1.8s	0.6s

tuned in such a way that it produces a similar accuracy, which is measured by the rejection rate of the complementary objects.

7 Conclusions and Outlook

In this article, we presented the idea of a model and method for one-class learning described by a hybrid FCT. The classifier tree consists of additive and/or subtractive combinations of convex fuzzy sets learned from a given set of data with an arbitrarily complex shape. It could be demonstrated that the algorithm is able to learn and parameterize FCTs with the same minimal number and combination of convex sets as an expert would construct manually. Our future work includes a further comparison of the hybrid FCT to other unary classification methods such as Bayesian approaches, artificial neural networks, and boundary

methods. This also involves an application to benchmark and real world data sets, e. g. from the field of condition monitoring.

References

1. Zhuang, L., Dai, H.: Parameter Optimization of Kernel-based One-class Classifier on Imbalance Text Learning. In: Yang, Q., Webb, G. (eds.) PRICAI 2006. LNCS (LNAI), vol. 4099, pp. 434–443. Springer, Heidelberg (2006)
2. Saez, J., Luengo, J., Herrera, F.: On the Suitability of Fuzzy Rule-based Classification Systems with Noisy Data. IEEE Transactions on Fuzzy Systems PP(99) (2012)
3. Szmidt, E., Kukier, M.: Classification of Imbalanced and Overlapping Classes Using Intuitionistic Fuzzy Sets. In: 2006 3rd International IEEE Conference on Intelligent Systems, pp. 722–727. IEEE Press, New York (2006)
4. Li, J.D., Zhang, X.J., Chen, Y.S.: Applying Expert Experience to Interpretable Fuzzy Classification System Using Genetic Algorithms. In: 4th International Conference on Fuzzy Systems and Knowledge Discovery, vol. 2, pp. 129–133 (2007)
5. Hempel, A.-J., Hähnel, H., Mönks, U., Lohweg, V.: SVM-integrated Fuzzy Pattern Classification for Nonconvex Data-inherent Structures Applied to Banknote Authentication. In: Bildverarbeitung in der Automation. inIT, Lemgo (2012)
6. Hempel, A.-J., Hähnel, H., Herbst, G.: Learning Non-convex Fuzzy Classifiers Using Single-class SVMs. In: IEEE International Conference on Fuzzy Systems, pp. 1–8. IEEE Press, New York (2013)
7. Kosko, B.: Fuzzy Systems as Universal Approximators. IEEE Transactions on Computers 43(11), 1329–1333 (1994)
8. Devillez, A.: Four Fuzzy Supervised Classification Methods for Discriminating Classes of Non-convex Shape. Fuzzy Sets and Systems 141(2), 219–240 (2004)
9. Senge, R., Hüllermeier, E.: Top-down Induction of Fuzzy Pattern Trees. IEEE Transactions on Fuzzy Systems 19(2), 241–252 (2011)
10. Hempel, A.-J., Bocklisch, S.F.: Fuzzy Pattern Modelling of Data Inherent Structures Based on Aggregation of Data with Heterogeneous Fuzziness. In: Rey, G.R., Muneta, L.M. (eds.) Modelling Simulation and Optimization, pp. 637–655. InTech (2010)
11. Päßler, M., Bocklisch, S.F.: Fuzzy Time Series Analysis. In: Hampel, R., Wagenknecht, M., Chaker, N. (eds.) Fuzzy Control: Theory and Practice, pp. 331–345. Physica-Verlag HD, Heidelberg (2000)
12. Schmidt, B., Bocklisch, S.F., Päßler, M., Czonsnyka, M., Schwarze, J.J., Klingelhöfer, J.: Fuzzy Pattern Classification of Hemodynamic Data Can Be Used to Determine Noninvasive Intracranial Pressure. Acta Neurochirurgica (suppl. 95), 345–349 (2006)
13. Herbst, G., Bocklisch, S.F.: Recognition of Fuzzy Time Series Patterns Using Evolving Classification Results. Evolving Systems 1(2), 97–110 (2010)
14. Mönks, U., Petker, D., Lohweg, V.: Fuzzy-Pattern-Classifier Training with Small Data Sets. In: Hüllermeier, E., Kruse, R., Hoffmann, F. (eds.) IPMU 2010, Part I. CCIS, vol. 80, pp. 426–435. Springer, Heidelberg (2010)
15. Hähnel, H., Hempel, A.-J., Mönks, U., Lohweg, V.: Integration of Statistical Analyses for Parameterisation of the Fuzzy Pattern Classification. In: 22. Workshop Computational Intelligence, pp. 115–131. KIT, Karlsruhe (2012)
16. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proc. of 2nd International Conference on Knowledge Discovery and Data Mining, pp. 226–231 (1996)