Pattern Anal Applic (2017) 20:427–439
DOI 10.1007/s10044-015-0505-z

CrossMark

THEORETICAL ADVANCES

Selecting locally specialised classifiers for one-class classification ensembles

Bartosz Krawczyk[1] · **Bogusław Cyganek**[2]

Received: 1 December 2014 / Accepted: 6 July 2015 / Published online: 26 July 2015

Abstract One-class classification belongs to the one of the novel and very promising topics in contemporary machine learning. In recent years ensemble approaches have gained significant attention due to increasing robustness to unknown outliers and reducing the complexity of the learning process. In our previous works, we proposed a highly efficient one-class classifier ensemble, based on input data clustering and training weighted one-class classifiers on clustered subsets. However, the main drawback of this approach lied in difficult and time consuming selection of a number of competence areas which indirectly affects a number of members in the ensemble. In this paper, we investigate ten different methodologies for an automatic determination of the optimal number of competence areas for the proposed ensemble. They have roots in model selection for clustering, but can be also effectively applied to the classification task. In order to select the most useful technique, we investigate their performance in a number of one-class and multi-class problems. Numerous experimental results, backed-up with statistical testing, allows us to propose an efficient and fully automatic method for tuning the one-class clustering-based ensembles.

Keywords Pattern classification · One-class classification · Fuzzy clustering · Competence areas · Classifier selection · Kernels

✉ Bogusław Cyganek
cyganek@agh.edu.pl

Bartosz Krawczyk
bartosz.krawczyk@pwr.edu.pl

[1] Department of Systems and Computer Networks, Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

[2] AGH University of Science and Technology, Al. Mickiewicza 30, 30-059 Kraków, Poland

1 Introduction

Machine learning becomes frequently used in real-life applications, allowing to analyze massive and complex data. However, most of the canonical learning algorithms assume that considered data belong to one of the pre-defined categories, called classes. The classical approach relies on a set of data with well known classes, which are used for training of a machine learning algorithm. Then, an unknown object can be assigned a class label based on the trained method. Thus, in order to prepare a competent classifier for a given task, one needs to have a training dataset consisting of representatives of each of the possible classes. However, in some cases one of the classes can be obtained with ease, while the remaining ones are hard or impossible to gather [14]. Collecting a representative set of objects may be costly, time consuming, unethical or simply impossible [27]. In such cases, one needs to create a fully operational pattern classification system with the usage of objects originating only from a single class. This learning paradigm is known as the one-class classification (OCC) [22].

In a case of one-class classifier, we need to maintain both good generalization abilities on the known class, and high discriminative power against the unknown classes. Since during the training step we have only objects from a single class at our disposal, then proper parameter and model selections are not trivial [33]. Adding the fact, that many one-class problems have complex distributions [30], one may see that it is difficult to prepare a single accurate and robust one-class classifier. This is why the ensemble

learning paradigm [36] has gained a significant attention in the machine learning community for the last years [6, 23, 28, 34].

Using a pool of classifiers prevents us from selecting the weakest model, and very often a combination of individual classifiers gives better accuracy than any single committee member. However for ensemble to work properly, its individual members should at the same time display high individual quality and be mutually complementary to each other. Combining several but similar models would not contribute anything new to the competence of the combined classifier. Similarly, fusing outputs of diverse but locally incompetent classifiers would return a weak ensemble. When constructing ensembles of classifiers we can consider heterogeneous or homogeneous structures. Heterogeneous committees consists of classifiers trained on the basis of different models (e.g., neural networks, support vector machines, decision trees, etc.) [26], while homogeneous ones use the same classifier model but each fed with a diverse input (e.g., different subsets of objects or features) [17]. Current studies report, that for one-class ensembles combining different object/feature subspaces yields significantly better results than utilizing different models [4], especially when combined with classifier selection step [23].

Following this observation, we have recently proposed a one-class clustering-based ensemble (OCClustE) [24] in a form of a highly efficient committee based on one-class classifiers and soft space partitioning [7]. It resolves around the idea of divide-and-conquer strategy, in which a complex problem is divided into a number of simpler tasks [6]. In OCClustE, we aim at detecting local competence areas and train classifiers on them. Thus, we decompose the original task into a set of smaller problems, reducing the number of instances that must be processed by each classifier. We apply a clustering algorithm, in order to detect groups of objects with spatial relations and to achieve more compact decision boundaries outputted by individual classifiers. For base classifiers, we use the weighted one-class support vector machines (WOCSVM) [3] that assign a weight to each objects, in accordance with its importance defined by some measure. This allows to filter out outliers and noisy objects. To avoid computationally expensive calculation of weights, we use membership values from soft clustering and apply them directly as weights in classifier's training procedure. We have also shown, that OCClustE can be efficiently used for both one-class tasks, as well as for efficient decomposition of the multi-class problems.

However, the main drawback of the aforementioned method lies in the assessment of a number of the competence areas (which directly translates to the number of classifiers in the ensemble). This factor cannot be easily determined beforehand and has a crucial impact on the quality of the ensemble quality. Manual tuning of this parameter is time-consuming and requires some knowledge about the machine learning domain (which cannot always be assumed, especially in case when the end-user is just using it as a data analysis tool).

To remedy this problem, in this paper we propose to investigate ten different methods for automatic detection of a number of competence areas for the OCClustE algorithm. They all have roots in model selection for clustering. We use methods from the three following groups: the one that requires only membership values, the one requiring both—the membership values and the dataset—and the one that is based on a statistical model selection. All of these methods are fully automatic and require no manual tuning. We ran extensive experiments in two scenarios: for typical one-class problems and for decomposing multi-class datasets. This allows us to shed light on the performance of each of the used methods, and to select the best performing ones.

The main contributions of this work are as follows:

- Proposal of new methods for automatic selection of locally specialized one-class classifiers, based on the determination of a number of mutually complementary competence areas.
- A complete step-by-step guidance on construction of an efficient one-class ensemble based on soft object space partitioning, that does not require any parameters to be tuned manually.
- Presentation of extensive experiments that allow evaluating usefulness of the proposed methods for selecting a number of competence areas in both one- and multi-class scenarios.

The remaining part of this paper is organized as follows: The next section discusses the basics of OCC and the possibility of applying these methods to cases without an access to counterexamples, as well as to cases when a multi-class problem decomposition is required. Section 3 describes the used OCClustE. Section 5 describes in detail the experimental study, while the last section concludes the paper and gives an outlook on possible future directions.

2 One-class classification

In this section, a brief overview of the OCC area will be given, with the respect to two possible areas of application: single-class learning and multi-class decomposition.

2.1 Classification in the absence of counterexamples

OCC assumes a scenario in which there can be two or more classes, but during the training step we have at our disposal only objects originating from a single class [18]. This

single class, which examples are often abundant, is known as the target class, or the target concept, and denoted as ω_T. It serves as a positive class for the pattern recognition system. At the same time there may be present one or more additional classes, but for other reasons such as cost, time, ethics, etc. we do not have any access to them during the training step [20]. However, objects of other classes may appear during the exploitation phase of the classifier, and the recognition system must be prepared to deal with them. They are jointly labeled as outliers, and denoted as ω_O. It is important to notice, however, that despite having an uniform label, outliers may be formed from more than one class. Commonly in OCC a uniform distribution of outliers in the decision space is assumed, meaning that they may appear at any point. This assumption reflects the lack of actual knowledge about the nature of outliers, and a need to prepare a highly flexible and robust one-class classifier.

To give a practical example of an one-class scenario, let us consider a nuclear power plant. We would like to prepare a decision support system, based on machine learning for automatic monitoring of condition of the plant [16]. Obtaining positive readings of a safe situation is relatively easy. One needs to define a set of features to observe and collect data for a given time. Here, one can generate a large collection of positive objects. However, gathering negative samples is not so straightforward. One would not want to deliberately damage a nuclear power plant, in order to get some readings. And when finding one with malfunctions main efforts would be laid towards securing the plant, and not gathering a large amount of data. And how one would know, that gathered counterexamples sufficiently describe all of possible malfunctions? In such a case, OCC is the most proper solution. One may bring more examples of the potential use of OCC, such as image/video classification (where it is impossible to determine what will appear on the scene) [7], data stream analysis (where new, unknown classes may appear due to data shifts and drifts) [15, 35], novelty detection [21] or bio-signal classification (where some pathologies may be dependent on the patient) [13].

During the last decade, a number of methods were proposed for tackling OCC problems. They can be divided into three families.

First one relies on the density-estimation methods, assuming that outliers do not fulfill the distribution of the target class [5]. This is the simplest and most straightforward approach for OCC, but surprisingly it tends to work well. However, these methods require a large number of training examples in order to properly capture the properties of the target class. Additionally, they offer quite low robustness to internal outliers and noisy data.

Second group is known as reconstruction methods. They have their roots in clustering algorithms, and aim at providing some description of the structure of the analyzed data [27]. They assume, that every new object from the target class will fulfill the detected structure. Objects, that do not fit into the structure are considered as outliers. Their main drawback is the assumption that training data allow us to construct a full structure of the target concept.

The third group is known as boundary methods [32]. These methods assume that it is often impossible to properly estimate the density or full structure of the target class. Instead, they concentrate on providing an enclosing boundary around the target concept, allowing some outliers at the same time. Volume of such an enclosing boundary cannot be too small (not to be overfitted to training data), but at the same time cannot be too large (not to lose robustness to potential outliers).Therefore, main effort in training a boundary classifier lies in the process of optimizing the boundary volume [19]. Boundary methods work well with small training sets. However, they require a number of parameters to be set, thus making the model selection a non-trivial task.

2.2 One-class classifiers for multi-class problems

OCC can also be used for scenarios, in which we have representatives of all of the classes at our disposal [24, 34]. In recent years, handling multi-class problems with decomposition techniques became a popular approach [11]. Decomposing a multi-class problem into a set of binary tasks is the most often used mechanism [10]. Here two strategies can be applied: one-versus-one (OVO) and one-versus-all (OVA). In OVO, a classifier for each possible pair of classes is constructed. On the other hand, in the case of OVA we construct a classifier for a given class and use the aggregation of the remaining classes as the negative set. OVO decomposes the problem into much more simpler tasks, but returns a high number of classifiers, especially for problems with a large number of classes. OVA gives M classifiers for a M-class problem, but trains them on a highly imbalanced datasets (1 class versus $M-1$ classes).

One may also use one-class classifiers to decompose a multi-class problem [37]. It can be seen as a special case of OVA, as we train a single one-class method for a given class, and use it to discriminate against other classes. Therefore, for M-class problem we get M one-class classifiers, getting a significantly more compact pool of models than OVA. On the other hand, as we do not use counterexamples during the training step, we alleviate the imbalance problem connected with OVA.

Of course using one-class classifiers comes at a cost of discarding a useful knowledge about distribution of the other classes during the training step. For standard scenarios, one-class decomposition will not be as efficient as binarization. However, OCC shows its usefulness in case of complex and difficult scenarios. OCC classifiers are robust

to imbalanced data (as they do not use counterexamples, they cannot be biased towards the majority class), class noise (as some methods have embedded mechanisms for dealing with internal outliers) or complex distributions. This is caused by a different learning paradigm. Binary and multi-class classifiers try to maximize the separability between the classes. One-class methods adapt themselves to the properties of the target concept.

Nevertheless, OCC is not a universal tool for handling multi-class problems. However, when carefully applied, it may outperform binary and multi-class methods for problems with complex distributions of data [24].

3 One-class clustering-based ensemble

OCClustE [24, 25] was proposed as an efficient ensemble system for one-class learning. It originated from trying to answer the problem on how to create a pool of base one-class classifiers, that are at the same time individually accurate and mutually diverse.

This method uses a clustering algorithm to partition the feature space into atomic subsets [7]. In the next step each of these clusters is used to train a one-class classifier. This leads to the formation of a pool of K classifiers assigned to the target class, as follows:

$$\Pi = \{\Psi_1, \Psi_2, \ldots, \Psi_K\}. \tag{1}$$

This allows us to easily create a pool of several one-class learners, dedicated to the target class. It assures the initial diversity (as a result of using different inputs in their training) and complementarity (as classifiers together cover all the decision space), which leads to better performance of the ensemble.

For the clustering step, OCClustE uses kernel fuzzy c-means, which is a modification of the fuzzy c-means algorithm that operates in an artificial feature space created by a kernel function [39]. Different partitioning methods were examined [24], but the kernel soft clustering returned superior results.

To further boost the recognition quality, OCClustE uses WOCSVM [3] as the base learner. It has been shown, that weighted one-class classifiers can outperform the canonical ones, due to an additional measure that controls influence of each object on the shape of the decision boundary. Additionally, weighted methods are more insensitive to internal outliers, that may be present in the target class (as it may contain irrelevant, noisy objects). Any data with a low weight, has a limited impact on the process of shaping the decision boundary.

The crucial element in using WOCSVM is the process of establishing weights, which is heuristic and time-consuming [3]. We introduce a novel approach for establishing the degree of importance of objects, based on the output of clustering algorithm. We use fuzzy clustering algorithm, that returns the membership functions for each object in the given cluster. We use these membership values as weights for WOCSVM [7]. This way, new weights reflect a degree of importance of a given object in a cluster and are pre-calculated, thus also reducing time needed for training WOCSVM [25].

Finally, we need to combine the individual outputs of base classifiers at our disposal. In our proposition, output fusion of the WOCSVM boundary methods is based on computing a distance between an object x and the decision boundary that encompasses the target class ω_{T}. To apply fusion methods we require a support function of an object x for a given class. Hence, for a given kth WOCSVM classifier, we propose to use the following heuristic function:

$$F_k(x, \omega_{\mathrm{T}}) = \frac{1}{c_1}\exp(-\mathrm{d}(x|\omega_{\mathrm{T}})/c_2). \tag{2}$$

It models a Gaussian distribution around the classifier, where $\mathrm{d}(x|\omega_{\mathrm{T}})$ is an Euclidean distance between the considered object and a decision boundary for the class ω_{T}, c_1 denotes a normalization constant and c_2 is the scale parameter. The parameters c_1 and c_2 should be fitted to the target class distribution.

There are several propositions on how to fuse the outputs of individual OCC models after such a mapping [31]. Let us assume that there are K OCC classifiers in the pool. In this paper, we use the mean of the estimated support functions which is expressed by:

$$y_{\mathrm{mp}}(x) = \frac{1}{K}\sum_k F_k(x, \omega_T). \tag{3}$$

This fusion method assumes that the outlier object distribution is independent of x and thus uniform in the area around the target concept. The schema of OCClustE is given in Fig. 1.

Additionally, OCClustE can be applied for multi-class problems, by decomposing a M-class dataset into M separate one-class problems [24]. Then we train M OCClustE algorithms, one for each class. To reconstruct the original multi-class output, error-correcting output codes (ECOC) are used. We have shown, that for difficult problems with a large number of classes our approach can deliver better performance than binary and multi-class classifiers.

In summary, OCClustE algorithm leads to several improvements compared with the standard OCC models, as follows:

- Boundary-based approaches (such as WOCSVM) were shown to display better generalization abilities than clustering-based (reconstruction) OCC [32], but are highly prone to atypical and complex data distributions.

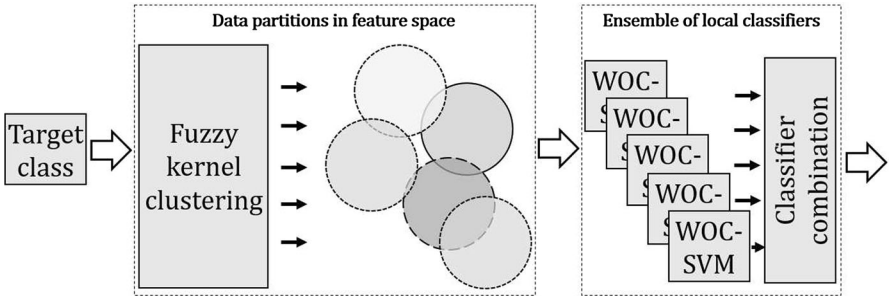

Fig. 1 Schema of the OCClustE framework

Therefore, a hybrid method utilizing both approaches combines the advantages of each while reducing their drawbacks.

- Since classifier is trained only on a reduced chunk of the data, its computational complexity is reduced in comparison to a single model approach. This reduces the probability of overtraining the one-class learner. Additionally, a number of individual classifiers can easily be applied in a distributed environment, leading to a significant decrease in execution time.
- Using chunks of data as the classifier input reduces the influence of negative effect, known as the empty sphere; that is, the area covered by the boundary in which no objects from the training set are located [18].
- A boundary classifier trained on a more compact data partition usually has a lower number of support vectors.
- By combining the fuzzy clustering with weighting scheme, we are able to obtain good estimation of weights assigned to training objects in a reduced time.

However, a limitation of the method has been also observed. OCClustE relies strongly on the number of competence areas (clusters), used in the training step. This number directly translates into a quantity of base classifiers in the ensemble. We noticed, that the variance in accuracy is high even for small changes of this parameter's value [24]. An exemplary influence of the number of clusters on the OCClustE structure is depicted in Fig. 2.

Tuning a proper number of clusters is time consuming and requires some specialist machine learning knowledge, that cannot be always assumed (e.g., in case of real-life applications in decision support systems and their end-users). Therefore, an efficient and fully automatic method for estimating a number of competence areas for OCClustE must be proposed.

4 Automatic selection of competence areas

In the previous section, we have identified the drawback of OCCLustE method. Now, we will discuss how to automatically select the number of competence areas for the ensemble.

As OCClustE works on the basis of clustering the object space, one may treat the problem of determining the number of competence areas as model selection for clustering algorithms [29]. They allow for an automatic selection of the optimal number of groups in data. With this, we can perform a tuning of OCClustE algorithm without any need for manual selection.

We investigate ten methods for automatic selection of a number of clusters, that in our case simultaneously represent a number of competence areas. The selection of such methods was dictated by a recent survey on their performance [38]. They form three groups: the first one using only the membership matrix, the second that uses a membership matrix and the dataset, and the third one which is based on statistical indexes.

For the description of the following methods, we assume that membership values coming from the kernel fuzzy c-means are collected in the membership matrix $\mathbf{U} = [\mu_{ij}]$, where μ_{ij} denotes a membership value of an jth data point into the ith cluster. We assume, that we investigate a number of clusters in the range of $[1, C]$, where C is the maximum number of examined clusters selected by the user. We assume also, that our training set $\mathcal{TRS}$ consists of N objects.

Please note, that all of the mentioned methods work on a set of possible clustering models. They use the same clustering results as input. Their sole purpose is to automatically identify the most suitable number of clusters among the input C models.

4.1 Indexes based on membership values

4.1.1 Partition coefficient

The partition coefficient (PC) can be defined as follows:

$$\mathrm{PC}(\mathbf{U}, C) = \frac{1}{N}\sum_{i=1}^{C}\sum_{j=1}^{N}\mu_{ij}^2 \tag{4}$$

where μ_{ij} is a membership value of an jth data point to the ith cluster, as assigned by the fuzzy c-means method. The PC ranges between $[1 / C, 1]$. The closer the index

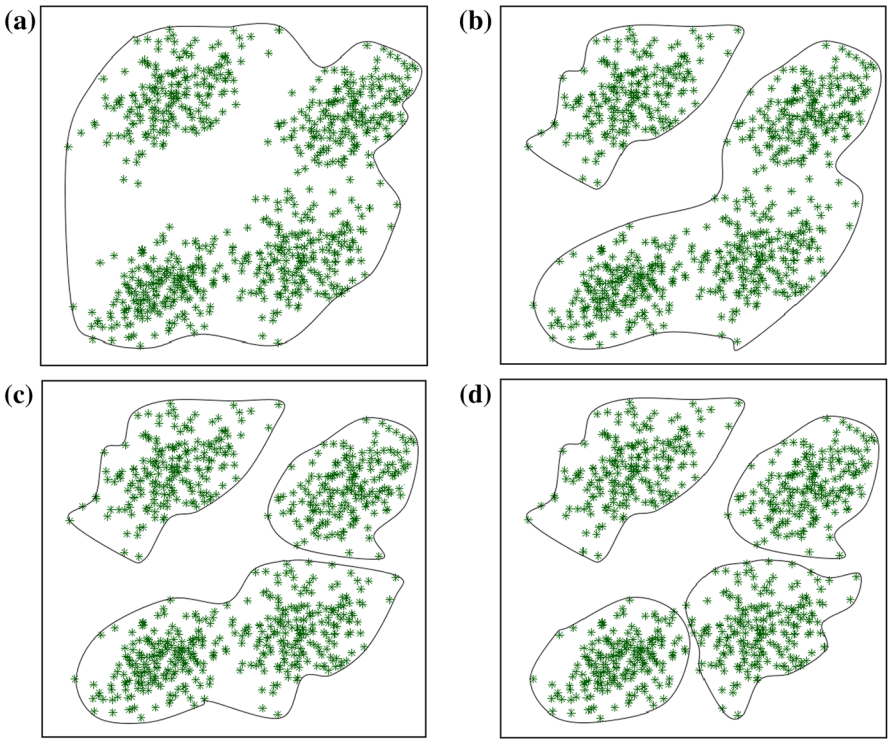

Fig. 2 Exemplary differences between the structures of OCClustE for a different number of competence areas and base classifiers: **a** single one-class classifier, **b** two clusters detected, **c** three clusters detected, **d** four clusters detected

to 1.0, the crisper the clustering. Therefore, a PC value close to 1 / C indicates that there is no clustering tendency in the analyzed data or the clustering method failed to detect one.

4.1.2 Partition entropy

The partition entropy coefficient (PE) can be defined as:

$$\mathrm{PE}(\mathbf{U}, C) = -\frac{1}{N}\sum_{i=1}^{C}\sum_{j=1}^{N}\mu_{ij}\log(\mu_{ij}). \tag{5}$$

This index is calculated only for a number of clusters greater than 1, and its value falls into the range $[0, \log C]$. Values of PE close to 0 indicate the difficulties in clustering of the analyzed data. Once again, the values close to the upper bond 1 / C indicate that there is no clustering tendency in the analyzed data or the clustering method failed to detect one.

4.1.3 Modified partition coefficient

The modified partition coefficient (MPC) method originates from an observation of the weakness of the PC method. PC is monotonously dependent on the number of clusters C. To alleviate this, we should look for a significant knees of increase of the criterion based on the number of clusters versus PC values. One can reduce the monotonicity tendency by using the following formula:

$$\mathrm{MPC}(\mathbf{U}, C) = 1 - \frac{C}{C-1}(1 - C * \mathrm{PC}(\mathbf{U}, C)), \tag{6}$$

where $0 \leq \mathrm{MPC} \leq 1$.

4.2 Indexes based on membership values and dataset

4.2.1 I index

The I index is defined as follows:

$$I(\mathbf{U}, C, \mathcal{TRS}) = \left(\frac{D_{\max}}{C \times E_C(\mathbf{U}, C, \mathcal{TRS})}\right)^p, \tag{7}$$

where

$$E_C(\mathbf{U}, C, \mathcal{TRS}) = \sum_{i=1}^{C}\sum_{j=1}^{N}\mu_{ij}\|x_i - c_i\|, \tag{8}$$

where x_i is an ith datapoint, and c_i denotes the centroid of the v_i cluster. The factor $D_{\max}$ stands for a maximum distance between the cluster prototypes. It will increase with the number of clusters. The second factor $\frac{1}{C}$ is responsible for reducing the value of this index with increase of the number of clusters. The third factor $\frac{1}{E_C}$ measures the total fuzzy dispersion, and penalizes the index with it's increase. Finally, the power of p controls the contrast between the different cluster configurations. In our experiments we set $p = 2$, as suggested in the literature [38].

4.2.2 Cluster validity measure

The cluster validity measure (CVM) is defined as:

$$\mathrm{CVM}(\mathbf{U}, C, \mathcal{TRS}) = C + (f \times G(2,1) + 1)\frac{D_a}{D_e}, \tag{9}$$

where D_a is the measure of compactness of clusters:

$$D_a(C, \mathcal{TRS}) = \frac{1}{N}\sum_{i=1}^{C}\sum_{x \in v_i} \|x - c_i\|^2, \tag{10}$$

and D_e is the measure of the average separation between two clusters over all possible pairs of clusters:

$$D_e(C) = \mathrm{average}(\|c_i - c_j\|)^2, \tag{11}$$

where $i = 1, 2, \ldots, C$, and $j = i + 1, \ldots, C$, f denotes a constant, $G(2, 1)$ is a radial basis function with mean value equal to 2.0 and standard deviation equal to 1.0. CVM measure should be minimized in order to obtain accurate and compact clusters.

4.2.3 Fukuyama–Sugeno index

The Fukuyama–Sugeno index (FS) is defined as follows:

$$\mathrm{FS}(\mathbf{U}, C, \mathcal{TRS}) = \sum_{i=1}^{C}\sum_{j=1}^{N} \mu_{ij}^m \|x_i - c_j\|^2 - \|c_j - \bar{c}\|^2, \tag{12}$$

where $\bar{c} = \sum_{i=1}^{c} c_i / C$ is an average of all centroids. Small values of FS indicate compact and separable clusters. The first term in Eq. 12 measures the compactness of the clusters, while the second measures the distances between a given centroid and the mean of all centroids.

4.2.4 Fuzzy hyper volume

The fuzzy hyper volume (FHV) is based on the concept of hyper volume and its density. It is defined as an average of a measure Q_i:

$$\mathrm{FHV}(\mathbf{U}, C, \mathcal{TRS}) = \sum_{i=1}^{C} Q_i, \tag{13}$$

where

$$Q_i(\mathbf{U}, \mathcal{TRS}) = \left|\sum_i\right|^{1/2} = \left(\frac{\sum_{j=1}^{N}(x_j - c_i)(x_j - c_i)^T}{\sum_{j=1}^{N}\mu_{ij}^m}\right). \tag{14}$$

Small FH values informs about the presence of compact clusters.

4.2.5 Average partition density

The average partition density (APD) can be formulated as follows:

$$\mathrm{APD}(\mathbf{U}, C, \mathcal{TRS}) = \frac{1}{C}\sum_{i=1}^{C}\frac{S_i}{V_i}, \tag{15}$$

where $S_i = \sum_{x \in x_i} \mu_{ij}$, x_i being the set of data points within a center of cluster c_i. S_i is called the sum of the central members of the c_i cluster. V_i should be calculated from Eq. 14.

4.2.6 Xie–Beni index

The Xie–Beni index (XBI), known as the compactness and separation validity function is defined as follows:

$$\mathrm{XBI}(\mathbf{U}, C, \mathcal{TRS}) = \left\{\frac{1}{N}\sum_{i=1}^{C}\sigma_i^2\right\} / \{D_{\min}\}^2, \tag{16}$$

where

$$\sigma_i^2(\mathbf{U}, \mathcal{TRS}) = \sum_{j=1}^{N}\mu_{ij}\|x_j - c_i\|^2, \tag{17}$$

and $D_{\min}$ stands for the minimum distance between the cluster centroids. Each σ_i^2 is a fuzzy weighted mean-square error for the ith cluster, and decreases with the increase of compactness of a cluster.

4.3 Statistical indexes

4.3.1 Akaike information criterion

Akaike information criterion (AIC) is defined as follows:

$$\mathrm{AIC}(C) = D_a + 2\mu(C)\sigma^2, \tag{18}$$

where $\mu(\mathbf{U}, C) = (C - 1)N + C$ denotes a number of degree of freedom of the model, D_a can be computed from Eq. 10, and the noise level σ^2 can be estimated from:

$$\sigma^2(\mathbf{U}, C) = \frac{D_a(C*)}{pN - \mu(C*)}, \tag{19}$$

where $C*$ is the maximum of a number of clusters, p is the co-dimension of the model ($p = 1$). The smaller the AIC value, the better the clustering performance for the data set.

5 Experimental investigations

The aim of the experimental analysis was to examine the usefulness of ten described methods for an automatic detection of a number of the competence areas for

OCClustE algorithm. To reflect the dual possibility of using our classifier, we conduct two types of experiments:

- Evaluation on one-class problems, where we train OCClustE with a single class data without an access to counterexamples.
- Evaluation on multi-class problem, where we decompose a data set into M separate one-class tasks, train OCClustE on each of them, and then reconstruct the original decision with the usage of ECOC combiner.

In the following subsections, we will describe the datasets used, set-up of experiments, and present the results with a discussion.

5.1 Data sets

For the experiments twenty datasets from the UCI repository [9] are used. Ten of them are binary datasets for the OCC experiments, and the remaining ten are multi-class problems for the one-class decomposition experiments.

For classification in the absence of counterexamples, we require one-class problems. As there are no dedicated one-class benchmarks publicly available, we transform binary data into one-class: the objects from the majority class were used as the target concept, while objects from the minority class as outliers.

Details of the datasets are given in Table 1.

Table 1 Details of datasets used in the experiments

No.	Name	Objects	Features	Classes
1.	Breast-cancer	286 (85)	9	2
2.	Breast-Wisconsin	699 (241)	9	2
3.	Colic	368 (191)	22	2
4.	Diabetes	768 (268)	8	2
5.	Heart-statlog	270 (120)	13	2
6.	Hepatitis	155 (32)	19	2
7.	Ionosphere	351 (124)	34	2
8.	Sonar	208 (97)	60	2
9.	Voting records	435 (168)	16	2
10.	CYP2C19 isoform	837 (181)	242	2
11.	Autos	159	25	6
12.	Car	1728	6	4
13.	Cleveland	297	13	5
14.	Dermatology	366	33	6
15.	Ecoli	336	7	8
16.	Flare	1389	10	6
17.	Lymphography	148	18	4
18.	Segment	2310	19	7
19.	Vehicle	846	18	4
20.	Yeast	1484	8	10

For binary problems, values in parentheses indicate the number of objects in the minority class

5.2 Set-up

For the experiment the WOCSVM with the RBF kernel, $\sigma = 0.1$ and cost parameter $\nu = 10$ is used as a base classifier. The pool of classifiers were homogeneous, i.e. consisted of classifiers of the same type. Kernel fuzzy c-means also employed the RBF kernel with $\sigma = 0.5$.

For all the tests, the maximum number of clusters to be examined was set to $C = 20$.

These settings were dictated by our previous experience with the OCClustE [24]. Additionally, we use the same parameters for each dataset, as our aims were to evaluate methods for competence areas detection, not the classifier itself.

For multi-class problems we generated a separate OCClustE for each class, then combined them with ECOC. We used an exhaustive code generation procedure described in [2].

In order to present a detailed comparison among a group of machine learning algorithms, one must use statistical tests to prove, that the reported differences among classifiers are significant [8]. We use both pairwise and multiple comparison tests. Pairwise tests give as an outlook on the specific performance of methods for a given data set, while the multiple comparison allows us to gain a global perspective on the performance of the algorithms over all benchmarks. With this, we get a full statistical information about the quality of the examined classifiers.

- For simultaneous training/testing and pairwise comparison, we use a 5×2 combined CV F-test [1]. It repeats five-time two fold cross-validation so that in each of the folds the size of the training and testing sets is equal. This test is conducted by comparison of all versus all.
- For assessing the ranks of classifiers over all examined benchmarks, we use a Friedman ranking test [8]. It checks, if the assigned ranks are significantly different from assigning to each classifier an average rank.
- We use the Shaffer post-hoc test [12] to find out which of the tested methods are distinctive among an $n \times n$ comparison. The post-hoc procedure is based on a specific value of the significance level α. Additionally, the obtained p values should be examined in order to check how different are the pairs of algorithms.

We fix the significance level $\alpha = 0.05$ for all comparisons.

5.3 Results and discussion

We present the performance of the examined methods according to the average number of competence areas they had identified, and to their final accuracy. For multi-class

problem, we assumed identical number of clusters selected for each class (in order to simplify the presentation of results).

5.3.1 Experiments with one-class classification

The average quantity of detected competence areas by each method is presented in Table 2. Table 3 depicts the accuracies of OCClustE methods trained on the basis of detected subsets of objects.

5.3.2 Experiments with multi-class decomposition

The average quantity of detected competence areas by each method is presented in Table 4. Table 5 depicts the accuracies of OCClustE methods trained on the basis of detected subsets of objects.

5.3.3 Discussion of results

Experimental study conducted on ten different methods for determining the number of competence areas for the purpose of training of the OCClustE classifier, allows us to draw several interesting conclusions.

One must remember, that all of then examined indexes work on the basis of kernel fuzzy *c*-means. So obtained results are related to the clustering model selection procedure, not to the clustering itself.

Firstly, one can observe a great variety in the outputs of all of the examined competence area selection methods. This proves, that they can return diverse results for the same set of points, and thus it is worthwhile to test their behavior over a set of benchmarks. However, for few datasets (e.g., heart-statlog or yeast) an identical number of areas is returned by all of the methods. This can lead to conclusions, that for these cases the clustering task is relatively easy and has a single best-performing solution.

When analyzing the accuracy of different OCClustE classifiers built on the basis of different object space partitions, one may observe how crucial is the impact of a proper selection of the number of base classifiers on the final quality of the model. This happens for both one-class and multi-class problems. Therefore, looking for methods that will be able to most precisely detect the number of partitions is of great importance.

We can observe, that for small datasets some methods tend to output too little competence areas, failing to discover hidden structures. On the other hand, for higher number of objects, these methods tend to over-cluster the data, locating false groups of objects. Both of these situations highly decrease the accuracy of OCClustE, and should be strongly avoided.

One should also take a look at the autos dataset, in which the best performance was delivered when no clustering was done (so a single one-class classifier for each class). Some of the metrics were able to detect, that there is no clustering tendency in this data and prevented the construction of the ensemble. Training multiple classifier system in such a case would only consume computational time, without any benefit for the recognition accuracy. Such an example show, that a proper evaluation metric for tuning OCClustE can also detect situations, in which case a committee of classifiers is not beneficial.

We have examined ten different evaluation methods, originating in three groups. From the tests on both one-class and multi-class problems, we can observe that the best performing ones are PE, *I*, FHV and AIC. These four methods significantly outperform remaining six, so we will concentrate our further discussion on them.

Partition Entropy is the only method, that requires just the membership values for computations. This reduces the required information, but at the same time reduces also the quality of the method—as PE achieves lower overall rank than the three remaining evaluation approaches. However, it is

Table 2 The average number of selected competence areas (clusters) by each approach on the target class ω_T

Dataset	PC	PE	MPC	*I*	CVM	FS	FHV	APD	XBI	AIC
Breast-cancer	5	4	5	3	6	4	3	5	5	3
Breast-Wisconsin	7	6	7	5	3	4	5	3	6	5
Colic	4	4	4	2	4	3	2	3	3	2
Diabetes	5	7	6	7	5	6	7	5	5	7
Heart-statlog	3	3	3	3	3	3	3	3	3	3
Hepatitis	2	3	2	3	4	4	3	2	2	3
Ionosphere	3	4	3	5	7	7	3	4	3	5
Sonar	3	3	3	3	3	3	3	3	3	2
Voting records	5	5	5	5	5	5	5	5	5	5
CYP2C19 isoform	9	10	7	11	7	10	11	9	9	12

All indexes use output of the kernel fuzzy *c*-means

Table 3 Results of the experimental results with the respect to the accuracy (%) and statistical significance

Dataset	PC[1]	PE[2]	MPC[3]	I^4	CVM[5]	FS[6]	FHV[7]	APD[8]	XBI[9]	AIC[10]
Breast-cancer	61.28	63.79	61.28	65.18	57.49	63.79	65.18	61.28	61.28	65.18
	5	1,3,5,8,9	5	1,2,3,5,6,8,9	–	1,3,5,8,9	1,2,3,5,6,8,9	5	5	1,2,3,5,6,8,9
Breast-Wisconsin	88.93	91.45	88.93	92.18	88.25	89.76	92.18	88.25	91.45	92.18
	–	1,3,5,6,8	–	1,3,5,6,8	–	1,3,5,8	1,3,5,6,8	–	1,3,5,6,8	1,3,5,6,8
Colic	78.03	78.03	78.03	80.72	78.03	75.69	80.72	75.69	75.69	80.72
	6,8,9	6,8,9	6,8,9	1,2,3,5,6,8,9	6,8,9	–	1,2,3,5,6,8,9	–	–	1,2,3,5,6,8,9
Diabetes	55.39	62.05	59.16	62.05	55.39	59.16	62.05	55.39	55.39	62.05
	–	1,3,5,6,7,8,9	1,5,8,9	1,3,5,6,7,8,9	–	1,5,8,9	1,3,5,6,7,8,9	–	–	1,3,5,6,7,8,9
Heart-statlog	87.11	87.11	87.11	87.11	87.11	87.11	87.11	87.11	87.11	87.11
	–	–	–	–	–	–	–	–	–	–
Hepatitis	56.78	60.46	56.78	60.46	58.12	58.12	60.46	56.78	56.78	60.46
	–	1,3,5,6,8,9	–	1,3,5,6,8,9	1,3,8,9	1,3,8,9	1,3,5,6,8,9	–	–	1,3,5,6,8,9
Ionosphere	78.64	80.63	78.64	80.92	72.07	72.07	78.64	80.63	78.64	80.92
	5,6	1,3,5,6,7,9	5,6	1,3,5,6,7,9	–	–	5,6	1,3,5,6,7,9	5,6	1,3,5,6,7,9
Sonar	92.12	92.12	92.12	92.12	92.12	92.12	92.12	92.12	92.12	93.56
	–	–	–	–	–	–	–	–	–	*ALL*
Voting records	89.64	89.64	89.64	89.64	89.64	89.64	89.64	89.64	89.64	89.64
	–	–	–	–	–	–	–	–	–	–
CYP2C19 isoform	75.62	80.09	73.18	80.98	73.18	80.09	80.98	75.62	75.62	83.01
	3,5	1,3,5,8,9	–	1,3,5,8,9	–	1,3,5,8,9	1,3,5,8,9	3,5	3,5	*ALL*
Avg. rank	9.20	4.10	8.60	2.70	6.80	5.80	2.80	7.30	5.40	2.30

Differences in obtained accuracy are related to different number of selected clusters according to a given index. Small numbers under accuracies stand for indexes of methods, from which the considered one is statistically superior according to a pairwise test

Table 4 The number of selected competence areas (clusters) averaged over all classes by considered approaches

Dataset	PC	PE	MPC	I	CVM	FS	FHV	APD	XBI	AIC
Autos	2	1	2	1	2	2	1	2	2	1
Car	7	8	7	9	7	8	9	7	7	9
Cleveland	2	2	2	2	2	2	2	2	2	2
Dermatology	2	3	2	3	2	2	3	2	2	3
Ecoli	2	2	2	2	2	2	2	2	2	2
Flare	10	8	10	6	8	9	6	10	8	6
Lymphography	2	2	2	2	2	2	2	2	2	3
Segment	10	5	10	5	8	8	5	10	10	5
Vehicle	7	5	7	5	8	8	5	7	7	4
Yeast	4	4	4	4	4	4	4	4	4	4

All indexes use output of kernel fuzzy c-means

easy and straightforward to compute and may be efficiently used in the cases of limited memory/computational time.

Both I index and FHV require at the same time membership values and dataset for being computed. They outperform PE method, due to the additional information embedded in the dataset. However, differences between them are pretty small, as both methods operate on similar assumptions.

AIC is based on statistical information. What is interesting, this is the only method that succeeded for all of the used datasets. For many cases it returned identical quantity of space partitions as PE, I or FHV, but for some other benchmarks (e.g., sonar) it was able to outperform all of the other methods.

We may conclude, that I, FHV and AIC are the best performing measures for automatic selection of the number

Table 5 Results of the experimental results with the respect to the accuracy (%) and statistical significance

Dataset	PC[1]	PE[2]	MPC[3]	I^4	CVM[5]	FS[6]	FHV[7]	APD[8]	XBI[9]	AIC[10]
Autos	63.98	68.12	63.98	68.12	63.98	63.98	68.12	63.98	63.98	68.12
	–	1,3,5,6,8,9	–	1,3,5,6,8,9	–	–	1,3,5,6,8,9	–	–	1,3,5,6,8,9
Car	85.18	87.82	85.18	90.06	85.18	87.82	90.06	85.18	85.18	90.06
	–	1,3,5,8,9	–	1,2,3,5,6,8,9	–	1,3,5,8,9	1,2,3,5,6,8,9	–	–	1,2,3,5,6,8,9
Cleveland	60.01	60.01	60.01	60.01	60.01	60.01	60.01	60.01	60.01	60.01
	–	–	–	–	–	–	–	–	–	–
Dermatology	91.07	94.52	91.07	94.52	91.07	91.07	94.52	91.07	91.07	94.52
	–	1,3,5,6,8,9	–	1,3,5,6,8,9	–	–	1,3,5,6,8,9	–	–	1,3,5,6,8,9
Ecoli	80.02	80.02	80.02	80.02	80.02	80.02	80.02	80.02	80.02	80.02
	–	–	–	–	–	–	–	–	–	–
Flare	60.86	69.02	60.86	73.28	69.02	60.86	73.28	60.86	69.02	73.28
	–	1,3,6,8	–	1,2,3,5,6,8,9	1,3,6,8	–	1,2,3,5,6,8,9	–	1,3,6,8	1,2,3,5,6,8,9
Lymphography	75.36	75.36	75.36	75.36	75.36	75.36	75.36	75.36	75.36	79.73
	–	–	–	–	–	–	–	–	–	*ALL*
Segment	79.17	92.18	79.17	92.18	84.82	84.82	92.18	79.17	79.17	92.18
	–	1,3,5,6,8,9	–	1,3,5,6,8,9	1,3,5,6,8,9	1,3,8,9	1,3,8,9	–	–	1,3,5,6,8,9
Vehicle	63.87	68.02	63.87	68.02	65.06	65.06	68.02	63.87	63.87	69.94
	–	1,3,5,6,8,9	–	1,3,5,6,8,9	1,3,8,9	1,3,8,9	1,3,5,6,8,9	–	–	*ALL*
Yeast	60.76	60.76	60.76	60.76	60.76	60.76	60.76	60.76	60.76	60.76
	–	–	–	–	–	–	–	–	–	–
Avg. rank	8.80	4.50	8.10	2.80	6.00	5.40	3.00	7.90	6.60	2.20

Differences in obtained accuracy are related to different number of selected clusters according to a given index. Small numbers under accuracies stand for indexes of methods, from which the considered one is statistically superior according to a pairwise test

of competence areas for the OCClustE, with AIC being the most robust one.

In order to prove our claims, we present the results for Shaffer post-hoc test in Table 6. From them one may see, that when considering multiple comparisons, the three selected methods are statistically superior to the remaining indexes. What is interesting, the differences between I and FHV indexes are statistically insignificant, while AIC is significantly better than these two (although the obtained p values are close to the significance threshold).

6 Conclusions

In this paper, the problem of an automatic determination of the optimal number of competence areas for the OCClustE was investigated. It extends our previous work, in which tuning of the competence areas of OCClustE was done manually, which required an extended effort from the end-user side.

To allow an automatic tuning of this parameter, we have investigated ten clustering evaluation indexes, originating

Table 6 Shaffer test for comparison between the indexes and remaining ones over 20 datasets

Hypothesis	p value	Hypothesis	p value	Hypothesis	p value
I vs PC	+(0.0087)	FHV vs PC	+(0.0113)	AIC vs PC	+(0.0064)
I vs PE	+(0.0376)	FHV vs PE	+(0.0407)	AIC vs PE	+(0.0302)
I vs MPC	+(0.0162)	FHV vs MPC	+(0.0202)	AIC vs MPC	+(0.0126)
I vs CVM	+(0.0230)	FHV vs CVM	+(0.0308)	AIC vs CVM	+(0.0184)
I vs FS	+(0.0302)	FHV vs FS	+(0.0394)	AIC vs FS	+(0.0258)
I vs FHV	=(0.1026)	FHV vs APD	+(0.0122)	AIC vs APD	+(0.0076)
I vs APD	+(0.0108)	FHV vs XBI	+(0.0371)	AIC vs XBI	+(0.0259)
I vs XBI	+(0.0339)	FHV vs AIC	−(0.0465)	–	–
I vs AIC	−(0.0488)	–	–	–	–

Symbol '=' stands for classifiers without significant differences, '+' for situation in which the method on the left is superior and '–' vice versa

from the model selection for clustering algorithms. We have applied them as a measure for selection of the number of base classifiers, that should be embedded in OCClustE. These metrics originated from three different groups: working on fuzzy membership values, on fuzzy membership values and data, and based on statistical measures.

We have carried out an extensive computational study with two types of experiments: the native one-class task and the one-class classifiers applied to decomposition of the multi-class problems. This allowed us to explore the dual application possibilities of our OCClustE classifier.

Experimental results, backed-up with a thorough statistical analysis showed that the I index, FHV and AIC indices are the best performing measures for automatic selection of a number of competence areas for the OCClustE, with AIC being the most robust one.

In future, we plan to use these methodologies with different types of space partitioning (e.g., non-negative matrix factorization) and with tensor-based one-class classifiers.

Acknowledgments Bartosz Krawczyk was partially supported by the Polish National Science Centre under the Grant PRELUDIUM No. DEC-2013/09/N/ST6/03504 realized in years 2014–2016. Bogusław Cyganek was partially supported by the Polish National Science Centre NCN in the year 2014, Contract No. DEC-2011/01/B/ST6/01994.

References

1. Alpaydin E (1999) Combined 5×2 cv f test for comparing supervised classification learning algorithms. Neural Comput 11(8):1885–1892
2. Alpaydin E, Mayoraz E (1999) Learning error-correcting output codes from data. IEE Conf Publ 2:743–748
3. Bicego M, Figueiredo MAT (2009) Soft clustering using weighted one-class support vector machines. Pattern Recognit 42(1):27–32
4. Cheplygina V, Tax DMJ (2011) Pruned random subspace method for one-class classifiers. In: Sansone C, Kittler J, Roli F (eds) Multiple classifier systems. Lecture notes in computer science, vol 6713. Springer, Berlin, pp 96–105
5. Cohen G, Sax H, Geissbuhler A (2008) Novelty detection using one-class parzen density estimator. An application to surveillance of nosocomial infections. In: Studies in health technology and informatics, vol 136, pp 21–26
6. Cyganek B (2010) Image segmentation with a hybrid ensemble of one-class support vector machines. In: Proceedings of 5th international conference on Hybrid artificial intelligence systems (HAIS'10), part I, San Sebastián, pp 254–261
7. Cyganek B (2012) One-class support vector ensembles for image segmentation and classification. J Math Imaging Vis 42(2–3):103–117
8. Demsar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
9. Frank A, Asuncion A (2010) UCI machine learning repository. http://archive.ics.uci.edu/ml. Accessed 21 July 2015
10. Galar M, Fernandez A, Barrenechea E, Bustince H, Herrera F (2011) An overview of ensemble methods for binary classifiers in multi-class problems: experimental study on one-vs-one and one-vs-all schemes. Pattern Recognit 44(8):1761–1776
11. Galar M, Fernández A, Barrenechea Tartas E, Bustince Sola H, Herrera F (2013) Dynamic classifier selection for one-vs-one strategy: avoiding non-competent classifiers. Pattern Recognit 46(12):3412–3424
12. García S, Fernández A, Luengo J, Herrera F (2010) Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. Inf Sci 180(10):2044–2064
13. Gardner AB, Krieger AM, Vachtsevanos G, Litt B (2006) One-class novelty detection for seizure analysis from intracranial EEG. J Mach Learn Res 7:1025–1044
14. Giacinto G, Perdisci R, Del Rio M, Roli F (2008) Intrusion detection in computer networks by a modular ensemble of one-class classifiers. Inf Fusion 9:69–82
15. Jackowski K (2014) Fixed-size ensemble classifier system evolutionarily adapted to a recurring context with an unlimited pool of classifiers. Pattern Anal Appl 17(4):709–724
16. Jackowski K, Platos J (2014) Application of AdaSS ensemble approach for prediction of power plant generator tension. In: de la Puerta JG, Ferreira IG, Bringas PG, Klett F, Abraham A, de Carvalho AC, Herrero l, Baruque B, Quintin H, Corchado E (eds) International joint conference SOCO14-CISIS14-ICEUTE14, advances in intelligent systems and computing, vol 299. Springer International Publishing, New York, pp 207–216
17. Jackowski K, Woźniak M (2009) Algorithm of designing compound recognition system on the basis of combining classifiers with simultaneous splitting feature space into competence areas. Pattern Anal Appl 12(4):415–425
18. Juszczak P (2006) Learning to recognise. a study on one-class classification and active learning. Ph.D. thesis, Delft University of Technology
19. Juszczak P, Tax DMJ, Pekalska E, Duin RPW (2009) Minimum spanning tree based one-class classifier. Neurocomputing 72(7–9):1859–1869
20. Kang P, Kim D, Cho S (2014) Evaluating the reliability level of virtual metrology results for flexible process control: a novelty detection-based approach. Pattern Anal Appl 17(4):863–881
21. Karami A, Guerrero-Zapata M (2015) A fuzzy anomaly detection system based on hybrid PSO-Kmeans algorithm in content-centric networks. Neurocomputing 149(Part C(0)):1253–1269
22. Koch MW, Moya MM, Hostetler LD, Fogler RJ (1995) Cueing, feature discovery, and one-class learning for synthetic aperture radar automatic target recognition. Neural Netw 8(7–8):1081–1102
23. Krawczyk B (2015) One-class classifier ensemble pruning and weighting with firefly algorithm. Neurocomputing 150:490–500
24. Krawczyk B, Woźniak M, Cyganek B (2014) Clustering-based ensembles for one-class classification. Inf Sci 264:182–195
25. Krawczyk B, Woźniak M, Cyganek B (2014) Weighted one-class classifier ensemble based on fuzzy feature space partitioning. In: 22nd international conference on pattern recognition (ICPR'14), Stockholm, pp 2838–2843
26. Łysiak R, Kurzyński M, Wołoszynski T (2014) Optimal selection of ensemble classifiers using measures of competence and diversity of base classifiers. Neurocomputing 126:29–35
27. Manevitz L, Yousef M (2007) One-class document classification via neural networks. Neurocomputing 70(7–9):1466–1481
28. Ratsch G, Mika S, Scholkopf B, Muller K (2002) Constructing boosting algorithms from svms: an application to one-class

classification. IEEE Trans Pattern Anal Mach Intel 24(9):1184–1199
29. Sun H, Wang S, Jiang Q (2004) Fcm-based model selection algorithms for determining the number of clusters. Pattern Recognit 37(10):2027–2037
30. Tax D, Duin RPW (2005) Characterizing one-class datasets. In: Proceedings of the sixteenth annual symposium of the pattern recognition association of South Africa, pp 21–26
31. Tax DMJ, Duin RPW (2001) Combining one-class classifiers. In: Proceedings of the second international workshop on multiple classifier systems (MCS'01). Springer, London, pp 299–308
32. Tax DMJ, Duin RPW (2004) Support vector data description. Mach Learn 54(1):45–66
33. Tax DMJ, Muller K (2004) A consistency-based model selection for one-class classification. Proc Int Conf Pattern Recognit 3:363–366
34. Wilk T, Woźniak M (2012) Soft computing methods applied to combination of one-class classifiers. Neurocomputing 75:185–193
35. Woźniak M (2011) A hybrid decision tree training method using data streams. Knowl Inf Syst 29(2):335–347
36. Woźniak M, Grana M, Corchado E (2014) A survey of multiple classifier systems as hybrid systems. Inf Fusion 16(1):3–17
37. Yeh C, Lee Z, Lee S (2009) Boosting one-class support vector machines for multi-class classication. Appl Artif Intel 23(4):297–315
38. Zanaty E (2012) Determining the number of clusters for kernelized fuzzy *c*-means algorithms for automatic medical image segmentation. Egypt Inform J 13(1):39–58
39. Zhang L, Zhou W, Jiao L (2002) Kernel clustering algorithm. Jisuanji Xuebao/Chin J Comput 25(6):587–590