Neurocomputing 196 (2016) 14–22

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Untrained weighted classifier combination with embedded ensemble pruning

Bartosz Krawczyk *, Michał Woźniak

Department of Systems and Computer Networks, Wroclaw University of Technology, Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland

ARTICLE INFO

Article history:
Received 22 April 2015
Received in revised form
3 February 2016
Accepted 16 February 2016
Available online 4 March 2016

Keywords:
Machine learning
Classifier ensemble
Combination rule
Ensemble pruning
Weighted aggregation

ABSTRACT

One of the crucial problems of the classifier ensemble is the so-called combination rule which is responsible for establishing a single decision from the pool of predictors. The final decision is made on the basis of the outputs of individual classifiers. At the same time, some of the individuals do not contribute much to the collective decision and may be discarded. This paper discusses how to design an effective combination rule, based on support functions returned by individual classifiers. We express our interest in aggregation methods which do not require training, because in many real-life problems we do not have an abundance of training objects or we are working under time constraints. Additionally, we show how to use proposed operators for simultaneous classifier combination and ensemble pruning. Our proposed schemes have embedded classifier selection step, which is based on weight thresholding. The experimental analysis carried out on the set of benchmark datasets and backed up with a statistical analysis, proved the usefulness of the proposed method, especially when the number of class labels is high.

1. Introduction

Machine learning becomes an irreplaceable tool for managing the data flood in the big data era. The amount, complexity and velocity of generated data greatly exceeds perceptive abilities of any human being. Therefore, designing novel and efficient methods for automated learning from data [9,38] is still the focus of intense research.

For a considered recognition task, we may often have more than a single classifier available. What is interesting, the number of misclassified objects by all individual classifiers is typically small. From this we can conclude that even if individual classifiers do not have high quality, their union could form a reasonably good compound classifier. The considered approach is called a multiple classifier system (MCS), combined classifier or classifier ensemble and is considered as one of the most vital fields in the contemporary machine learning [43].

Forming an ensemble requires an input pool of classifiers and a method for combining their individual outputs into a single committee decision. Optionally, one may assume that not all of classifiers from the pool are significantly important, and perform a pruning step in order to discard some learners. In recent years, many different schemes for this task were presented, because each of them is subject to some restrictions. Many approaches are computationally expensive and cannot be used for real-time classification or stream mining. Therefore, despite almost two decades of progress there are still novel ensemble approaches being frequently proposed in the literature.

This work focuses on crucial ensemble forming steps: pruning and weighted classifier combination.

For most real-life classification tasks we can create/collect a large number of classifiers. However, for ensemble to work properly it should be formed by mutually complementary models of high individual quality. Adding new classifiers that do not exploit a new area of competence do not improve the ensemble, only increases the computational cost and reduces its robustness. The problem lies on how to select a useful subgroup from a large pool of classifiers at hand. Due to the high computational complexity of full-search over all of the possible classifier subgroups, several heuristic groups of methods were proposed [39]. However, one should note that these methods are highly dependent on properly established evaluation criteria and require significant processing time. These properties can become prohibitive in certain application areas.

When having selected a number of competent classifiers, one need to design a combination rule in order to establish a collective decision of the ensemble [35]. Such a mechanism should be able to exploit the individual strengths of classifiers in the pool, while at the same time minimizing their drawbacks [34]. For many years

* Corresponding author.
E-mail addresses: bartosz.krawczyk@pwr.edu.pl (B. Krawczyk), michal.wozniak@pwr.edu.pl (M. Woźniak).

http://dx.doi.org/10.1016/j.neucom.2016.02.040
0925-2312/

voting algorithms were the ones most popular. However, for more complex data one requires flexible approaches that can adjust their combination methods to the properties of analyzed datasets. Trained combiners have gained a significant attention of the machine learning community [45]. However, such aggregation methods require additional training time and access to separate subset of examples—and once again this requirements can become prohibitive in certain application areas.

In this work, we introduce a novel method for weighted combination with embedded ensemble pruning step based on our previous works on efficient, fast and simple combination rules [26]. We propose novel weighted aggregation operators which do not require learning and have embedded pruning procedure that do not require any criterion to work. We work on modification of two popular operators: average of supports and maximum of supports. Their main drawback lies in lack of robustness to weak and irrelevant classifiers, and in minimizing the influence of other ensemble members. By using a Gaussian function to estimate the weights for the entire ensemble, we achieve a smooth method for reducing, but not eliminating the influence of weaker classifiers. At the same time by adjusting a threshold on the value of weights, we are able to prune the ensemble by discarding incompetent learners.

The main contributions of this paper are given below:

1. We introduce two novel untrained combination rules for forming efficient classifier ensembles. They use continuous outputs of base classifiers (support functions) and are based on popular *maximum* and *average* operators. Our methods provide a more robust combination, as they use a Gaussian function to assign weights to each base classifier. Therefore, we are able to control the level of influence of each classifier on the combination procedure.
2. We propose an unsupervised methodology for calculating weights for base classifiers in the ensemble. This way we are able to apply a weighted combination scheme, where weights assigned to each classifier are based on the individual model and class number without a need for an external validation set. This allows us to boost the classifier's influence over the classes where it is most competent, while reducing its role for classes that cannot be properly recognized by it. It is a highly suitable solution for scenarios where we do not have abundant data and cannot afford to use some of them to form a separate combiner training set. Additionally, this allows us to efficiently mine datasets with a large number of classes.
3. We propose to embed an unsupervised ensemble pruning step within the combination operators. It is based on thresholding the weights assigned to models and discarding classifiers with lowest weights assigned (i.e., non-competent ones). It does not require any external procedure or extensive computational effort to perform and is able to significantly reduce the size of the committee. This is highly suitable for scenarios with limited computational resources.

With the use of a wide selection of benchmark datasets with large number of classes we show the quality of proposed combiners. We present the results of pruning step that indicates the possibility of creating smaller, but efficient ensembles with our methods. Comparison with state-of-the-art untrained and trained combiners is backed-up with a rigorous statistical analysis that further proves the usefulness of the proposed ensemble fusion algorithms.

The remaining parts of this manuscript are organized as follows. Next section described the background and advantages of ensemble systems in machine learning. Section 3 presents a detailed description of our weighting method together with the embedded pruning mechanism. Section 4 depicts the set-up, datasets and experimental analysis. Section 5 summarizes the main findings, while the final section concludes this manuscript.

2. Classifier ensembles in machine learning

Classifier ensembles concentrate on the problem of efficient exploitation of different classifiers available for a considered recognition problem, believing that utilizing more than one single model can be beneficial for the formed system. This concept was first presented by Chow [6], who proved that the decision of independent classifiers with appropriately defined weights is optimal.

Let us now present some advantages of an MCS:

- The design of an MCS can be seen as following similar steps as the design of a canonical pattern recognition system [17]. In the standard approach, we concentrate on selecting the most informative features and choosing the best classification algorithm from the set of available ones. When forming a classifier ensemble, we aim to create a set of mutually complementary and individually accurate classifiers and assign an appropriate combination method, which can most efficiently combine their individual decisions [32].
- One may find numerous literature reports stating that MCSs are able to improve the overall performance when compared with the best individual classifier from the pool. This happens because they are able to exploit unique strengths of each of the individual classifiers. In some cases (e.g., when a majority voting is applied on a group of independent classifiers) the characteristics have been proven in an analytical way [40]. Additionally, an MCS protects against the selection of the worst classifier, when we have only a small training sample at our disposal [33].
- One should notice that some machine learning algorithms (e.g., C4.5 based on a top down induction decision tree concept) are de facto heuristic search algorithms. For them it is not guaranteed that the best possible model for a given dataset is found. One may alleviate this by the combined approach, which would start simultaneously searching from different points of the search space.
- Combined classifiers could be easily used in high-speed computing environments such as parallel and multithreaded computer architectures [8]. Another attractive area of application is distributed computing systems (P2P, GRID) [24], especially in the case of sensor networks [37] or multi-source datasets.

When designing an MCS one should take into consideration a number of important issues that can be grouped into the following problems:

1. How to select a pool of diverse and complementary individual classifiers for the ensemble?
2. How to design a combination rule that can exploit the strengths of the selected classifiers and combine their outputs optimally?
3. How to propose a suitable topology for a given system i.e., interconnections among classifiers in the ensemble.

We do not address the last issue because most of the combined classifiers are based on a parallel topology, which has a good methodological background [29] and is used in this work.

When selecting members to the committee one should assure that they work on different principles or utilize different components/data subsets. Apart from increasing the computational complexity, combining similar classifiers should not contribute much to the MCS under construction. An ideal ensemble consists

of classifiers that display at the same time high individual accuracy and high diversity (i.e., mutually complementary). To obtain positive results from the fusion process, we need to select only the valuable classifiers from the pool [5].

A strategy for generating the ensemble should guarantee an improvement in its diversity, while maintaining the accuracy [47]. Although, the concept 'diversity' is known in the machine learning community, but it is rather intuitive. The problem how to measure it still remains unsolved, while a plethora of propositions have been formulated, but their usefulness is limited and could be applied under the strong assumptions [28]. There are several proposals on how to enforce the diversity within the pool of classifiers:

- One may use different data partitions or create a number of sub-datasets through data splitting, a cross-validated committee, bagging, or boosting [29], in the hope that classifiers trained on different inputs would be complementary [2,17]. One may split training objects [30] or feature space [19]. Selected subspaces are then used to train a pool of classifiers to assure diversity of the pool [7].
- We could train each individual classifier to recognize a reduced subset of classes (e.g., a binary classifier with the usage of the one class against one strategy [36]) in order to create simplified and class-specialized learners [15]. Then we need to use a fusion method that can reconstruct the original multi-class problem from a number of reduced decisions [14]. Error-Correcting Output Codes [11] is a well-known technique for such tasks.
- We could train individual classifiers based on different models or different versions of models.

Another important issue is the choice of a method that will allow us to combine individual outputs of classifiers in the ensemble [4]. We can divide the above-mentioned algorithm into two groups:

1. Methods that make decisions on the basis of individual classifiers' labels.
2. Methods that propose constructing new support functions based on supports of individuals [31].

The former group consists mainly of voting algorithms [3,46]. Initially, majority voting was the most popular approaches, but in later works more advanced methods were proposed. Their main advantage is taking into consideration the importance of decisions coming from particular committee members [41,27].

Many interesting properties of MCSs have been derived analytically, but these are typically valid only under strong restrictions, such as particular cases of the majority vote [18]. Unfortunately, such assumptions and restrictions are in most cases not very useful for solving practical problems. Therefore, we look for more flexible methods that can adjust themselves to the considered problem. This can be achieved by training the weights over a given set of objects [42,44].

The second group of combination methods relies on discriminant analysis. Standard combination methods from this group do not require learning, and use simple operators, as minimum, maximum, product, or mean. Yet one must notice that they are typically subject to very restrictive conditions [12], which limit their practical use.

3. Untrained ensemble pruning and weighted combination

In this work, we propose new untrained aggregation operators which could exploit the competencies of the individual classifiers. Let us now present in detail the proposed combination operators with embedded ensemble pruning step.

3.1. Weighted combination of classifiers

As in this work we concentrate on weighted combination of continuous outputs, therefore let us assume that each individual classifier makes a decision on the basis of the values of support functions.

Let $\Pi = \left\{\Psi^{(1)}, \Psi^{(2)}, \ldots, \Psi^{(n)}\right\}$ be the pool of n individual classifiers and $F_{i,k}(x)$ stands for a support function that is assigned to class i ($i \in \mathcal{M} = \{1, \ldots, M\}$) for a given observation x and which is used by the classifier $\Psi^{(k)}$ from the pool Π.

The combined classifier $\Psi(x)$ uses the following decision rule

$$\Psi(x) = i \quad \text{if } F_i(x) = \max_{k \in M} F_k(x), \tag{1}$$

where $F_k(x)$ is the weighted combination of the support functions of the individual classifiers from Π for the class k .

In this work, we assume that weights are dependent on classifier and class number. Weight $w_{i,k}$ is assigned to the k-th classifier and the i-th class. For a given classifier, weights assigned for different classes could be different. In our previous works, we have shown that this approach leads to a significant improvement over traditional methods [22]. With this, we can formulate our combination scheme as follows:

$$F_i(x) = \sum_{k=1}^{n} w_{i,k} F_{i,k}(x) \quad \text{and} \quad \forall i \in \mathcal{M} \ \sum_{k=1}^{n} w_{i,k} = 1. \tag{2}$$

In this work we focus on such a method, because as shown in [45], this type of combiner achieves fairly good quality and does not require a priori knowledge of the weights. This is contrary to the case where weights are also dependent on the feature values. If weights depend on x, they are de facto functions (as well known *Mixture of Experts* [23]) and their estimation is more complicated, usually requiring a priori knowledge about them.

3.2. Proposed operators for classifier combination

The simple operators as maximum or average usually behave reasonably well but their work could be spoiled by very imprecise estimators of the support functions used by only a few classifiers from a pool. To alleviate this strong disadvantage, we propose novel modifications of the above-mentioned operators that are able to more efficiently exploit the available pool of classifiers. Introduced combination schemes take into consideration all available support functions returned by the individual classifiers from the pool, but the functions which have the similar values to maximum or average have the strongest impact in the final value of the common support function calculated by using Eq. (2). With this, we are able to deal with two drawbacks of these popular operators:

1. Maximum operator can efficiently work in case of a single dominant model. However, it is unable to deal with a situation in which we have one strong model, but other members of the pool can also contribute some valuable knowledge to the combination process. This usually happens, when we train classifiers on the basis of different data subsets where each of them achieved some local specialization.
2. Average operator can efficiently filter out single outlier classifiers and work with a pool of similarly competent models. However, when more models output highly varying decisions we get a less stable prediction, as each classifier has identical influence over the final decision. Therefore, we should somehow control the degree of their importance.

The two proposed combiners offer an improved robustness to the mentioned pessimistic scenarios, as they calculate weights for all of classifiers (using *max* or *avg* operators as the basis for the weight estimation process). The main advantage of these operators is the fact that the weight calculation process does not require additional data or computational time. This is especially attractive for high-speed data stream mining in non-stationary environments where we need to quickly update our learning model [21].

Additionally, we should notice that there may be some irrelevant classifiers in the pool and that for a large pool of classifiers most of the weights will become very small (in order to satisfy the condition from Eq. (2)). Additionally, similar classifiers can destabilize the method, as they will have assigned similar weights without contributing anything new to the formed ensemble. Therefore, one should be able to discard them before the combination step. This problem is known as ensemble pruning and is usually realized through and external search procedure that selects the optimal subset of learners [48]. However, most of the pruning methods are time consuming, require a large validation set and properly selected criteria, such as accuracy, AUC or diversity. Such criteria do not often lead to a satisfactory results (as using accuracy may lead to large and similar ensembles, while diversity will not take into account the individual quality of models) and selecting a proper metric for a given problem is not a trivial task. Another important fact is that in most of the literature methods there is no direct relation between the ensemble pruning and establishing classifier weights [25]. Usually pruning is used with simple aggregation schemes, while more compound and weighted combinations are examined for a pre-defined pool of classifiers. This can be seen as a significant drawback, as the subset of models have a great impact on selected weights and vice versa.

To deal with these shortcomings, we propose to embed an ensemble pruning algorithm within our weight estimation and combination approach. However, such a method must have low computational complexity and do not impose any additional constraints on proposed combination schemes. We propose to implement the pruning threshold ϕ, in order to discard all classifiers with assigned weights $w_{i,k} \leq \phi$. Then we normalize the weights for a reduced number of learners, thus increasing their level of influence over the ensemble decision.

The proposed operators are called NP-AVG and NP-MAX and can be calculated according to Algorithm 1. The only difference is the calculation of the $\overline{F}_i(x)$.

For NP-AVG it is calculated according to

$$\overline{F}_i(x) = \frac{\sum_{k=1}^{N} F_{i,k}(x)}{N}, \tag{3}$$

and for NP-MAX using the following formulae:

$$\overline{F}_i(x) = \max_{k \in \mathcal{M}} F_{i,k}(x). \tag{4}$$

Algorithm 1. General framework for ensemble pruning and weight calculation.

Require: Π—pool of n elementary classifiers
$F_{i,k}(x)$—support function value for each class i returned by each individual classifier k from Π
ϕ - pruning threshold
Ensure $w_{i,k}(x)$—weights assigned to each support function $F_{i,k}(x)$ which could be used in Eq. (2)
1: **for** i:=1 **to** M **do**
2: w:=0
3: Calculate $\overline{F}_i(x)$ according to Eq. (3) for NP-AVG or according to Eq. (4) for NP-MAX
4: **for** k:=1 **to** n **do**
5: $w_{i,k}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-(F_{i,k}(x) - \overline{F}_i(x))}{2\sigma^2}\right)$
6: w:=w+$w_{i,k}(x)$
7: **end for**
8: **for** k:=1 **to** n **do**
9: **if** $w_{i,k} \leq \phi$
10: discard the k-th classifier
11: **end for**
12: **return** pruned pool of p classifiers
13: **for** k:=1 **to** p **do**
14: w:=$\frac{w_{i,k}(x)}{w}$
15: **end for**
16: **end for**

The only parameters of the proposed operators is σ which is equivalent of standard deviation in normal distribution, and a pruning threshold ϕ.

Let us now take a closer look at the proposed algorithm. As an input it requires only the support functions from each base classifier in the pool. Here one can use almost any kind of classifier, as one may use any continuous output of a classifier (e.g., final decision in neural networks or distance in support vector machines). Therefore, the proposed method is highly flexible and does not impose any restrictions on the type of classifier used.

We consider each class independently (see line 1 of the algorithm), as the class weights are dependent on the classifier and class number. For each class we calculate the NP-average (according to Eq. (3)) or NP-maximum (according to Eq. (4)) of supplied supports in order to create a basis for our method (see line 2 of the algorithm). Then for each base classifier we calculate its new weight according to a Gaussian distribution with its mean value equal to NP-AVG or NP-MAX values (see line 5 of the algorithm). Having calculated the weights we may now conduct the pruning step. We compare the weight assigned to examined classifier with a predefined pruning threshold (see line 9 of the algorithm). A learner is discarded if its associated weight is below the threshold (see line 10 of the algorithm).

Additionally, we must ensure that the weights stay within the given bounds (see Eq. (2) and lines 6 and 14 of the algorithm).

As an output we receive a pruned ensemble with calculated and nominalized weights.

4. Experimental investigations

The presented experimental study was designed in order to answer the following open questions:

- Are the proposed weighted aggregation operators N-AVG and N-MAX are able to outperform their canonical equivalents?
- How the proposed operators perform compared with popular untrained classifier combiners in terms of classification accuracy improvement?
- How the proposed untrained methods perform compared with popular trained classifier combiners in terms of classification accuracy improvement and time complexities?
- Is the embedded pruning approach able to deliver more compact ensembles that display similar or better classification accuracies compared with unpruned pool?

Let us now present the detailed set-up of experiments, obtained results and following discussion.

Table 1
Details of datasets used in the experiments.

No.	Name	Objects	Features	Classes
1.	Auslan	2565	128	95
2.	Autos	159	25	6
3.	Car	1728	6	4
4.	Cleveland	297	13	5
5.	Dermatology	366	33	6
6.	*E. coli*	336	7	8
7.	Flare	1389	10	6
8.	Isolet	7797	617	26
9.	Led7digit	500	7	10
10.	Letter-2	20 000	16	26
11.	Lymphography	148	18	4
12.	Nursery	1296	8	5
13.	Penbased	1099	16	10
14.	Satimage	643	36	7
15.	Segment	2310	19	7
16.	Shuttle	2175	9	7
17.	Vehicle	846	18	4
18.	Vowel	990	13	11
19.	Yeast	1484	8	10
20.	Zoo	101	16	7

4.1. Datasets

In total we chose 20 well known datasets from the UCI Repository [13] with a large number of classes. For datasets with missing values, instances without full set of features available were removed.

Details of the chosen datasets are given in Table 1.

4.2. Set-up

As a base classifier, we have decided to use neural network (NN)—realized as a multi-layer perceptron, trained with back-propagation algorithm, with a number of neurons depending on the considered dataset: in the input layer equal to the number of features, in the output layer equal to the number of classes and in the hidden layer equal to half of the sum of neurons in previously mentioned layers. They used sigmoid activation function and were trained for 500 iterations each.

The base models for the ensemble were constructed with the usage of Bagging. The pool of classifiers used for experiments was homogeneous and consisted of 50 neural networks.

As reference methods we decided to use the following popular classifier combination algorithms:

- For combiners that do not require training: majority voting (MV), maximum of support (MAX), average of supports (AVG), product of supports (PRO), and weights assigned according to individual classifier accuracy (ICA):

$$w_{i,k} = \log \frac{p_k}{1-p_k}, \tag{5}$$

  where p_k is the accuracy of k-th classifier [29].
- For combiners that require training: Behavior-Knowledge Space (BKS) [20], Decision Templates using Euclidean distance (DT) [27] and neural combiner based on single-layer perceptron (NC) [45].

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 [16]. We use both pairwise and multiple comparison tests. Pairwise tests give as an outlook on the specific performance of methods for a given dataset, while 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.

- We use a 5×2 CV combined F-test [1] for simultaneous training/testing and pairwise statistical analysis. It repeats five-times two fold cross-validation. To evaluate the proposed pruning scheme, we need to have a validation set $\mathcal{VS}$. Therefore, for each iteration of 5×2 CV, we separate 5% of the training data for validation purposes, having 45% of objects in training set $\mathcal{TRS}$, 5% in validation set $\mathcal{VS}$ and 50% in testing set $\mathcal{TS}$. This procedure additionally ensures that these sets are disjoint: $\mathcal{TRS} \cap \mathcal{VS} \cap \mathcal{TS} = \emptyset$. The combined F-test is conducted by comparison of all versus all. As a test score the probability of rejecting the null hypothesis is adopted, i.e. that classifiers have the same error rates. As an alternative hypothesis, it is conjectured that tested classifiers have different error rates. A small difference in the error rate implies that the different algorithms construct two similar classifiers with similar error rates; thus, the hypothesis should not be rejected. For a large difference, the classifiers have different error rates and the hypothesis should be rejected.
- Demsar's critical difference plots based on Nemenyi test are used for visual comparison among tested methods. One must remember that although this is very illustrative Nemenyi test is very conservative and has a low power. Therefore, additional testing is required.
- For assessing the ranks of classifiers over all examined benchmarks, we use a Friedman ranking test [10]. It checks, if the assigned ranks are significantly different from assigning to each classifier an average rank.
- We use the Shaffer post hoc test [16] 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 given two algorithms are.
- We fix the significance level $\alpha = 0.05$ for all comparisons.

The experiments were performed on a machine equipped with an Intel Core i7-4700MQ Haswell at 2.40 GHz processor and 16.00 GB of RAM and conducted in R[1] environment.

4.3. Experiment 1—untrained combiners

Firstly, we need to establish the level of influence of value of pruning threshold ϕ on the quality of the ensemble. A grid search was performed for $\phi \in [0; 0.5]$ with step = 0.05. The best parameter values according to the final accuracy and the average size of the ensemble after pruning are given in Table 2. If $\phi = 0$, then no pruning was applied. We use the established values of this parameter for further comparisons.

Results of the experiments, presented according to the accuracy of the examined methods, are given in Table 3. Fig. 1 depicts the plot of Demsar's critical difference test. Results of Shaffer post hoc test over accuracy are given in Table 4.

Let us present the conclusions derived from the experiments.

The proposed operators behaved reasonably well and outperformed, with statistical significance, all of the traditional methods for 14 out of 20 data sets. Shaffer test showed that N-MAX is slightly better than N-AVG, and what is interesting is it can outperform most of the traditional untrained approaches except maximum operator.

[1] http://www.r-project.org

Table 2
Selecting the value of pruning threshold ϕ, and is influence on the size of the ensemble.

Dataset	Best ϕ value		Avg. size of the ensemble	
	NP-AVG	NP-MAX	NP-AVG	NP-MAX
Auslan	0.20	0.25	38 ± 2.98	37 ± 1.72
Autos	0.00	0.00	50 ± 0.00	50 ± 0.00
Car	0.30	0.25	31 ± 2.45	29 ± 3.03
Cleveland	0.00	0.00	50 ± 0.00	50 ± 0.00
Dermatology	0.15	0.10	19 ± 3.23	16 ± 2.09
E. coli	0.10	0.15	17 ± 4.23	18 ± 2.78
Flare	0.20	0.15	33 ± 1.28	32 ± 2.03
Isolet	0.15	0.15	25 ± 3.82	26 ± 3.04
Led7digit	0.25	0.30	39 ± 2.75	37 ± 2.62
Letter-2	0.10	0.10	23 ± 1.43	23 ± 1.02
Lymphography	0.00	0.00	50 ± 0.00	50 ± 0.00
Nursery	0.20	0.30	43 ± 3.42	42 ± 2.76
Penbased	0.10	0.10	13 ± 2.04	11 ± 1.58
Satimage	0.25	0.25	16 ± 2.87	16 ± 2.87
Segment	0.20	0.15	30 ± 4.02	28 ± 3.11
Shuttle	0.20	0.25	41 ± 0.74	41 ± 0.36
Vehicle	0.05	0.05	17 ± 2.26	17 ± 1.84
Vowel	0.05	0.05	35 ± 1.04	33 ± 0.89
Yeast	0.15	0.05	15 ± 3.72	14 ± 2.39
Zoo	0.05	0.05	34 ± 0.53	34 ± 0.53

Analyzing characteristics of the used data benchmark sets we can suppose that proposed operators work well especially for the classification task where the number of possible classes is high (e.g. Auslan, Letter-2 or Isolet). This can be explained by the properties of our combination methods (weights depend on the classifier and the class number), as they are able to efficiently adapt the weights to class-based local competencies.

Each of the proposed operators outperform majority voting and product for almost all data sets. We can conclude that in the case of an absence of additional learning examples (which can be used to train the combination rule) the untrained aggregation is a better choice than voting methods. This observation is also known and confirmed by other researches as [45].

Additionally, one should notice that the proposed pruning method allowed for an efficient reduction of the number of classifiers in the pool. For most cases, we obtained 20–50% reduction in the ensemble size, still achieving excellent classification accuracy.

4.4. Experiment 2—trained combiners

In this experiment we compare our proposed approaches with popular trained classifier combination methods. We use here exactly the same data partitioning as in experiment 1, therefore pruning setting presented in Table 2 applies here as well. Results of the experiments, presented according to the accuracy of the examined methods, are given in Table 5, while their averaged processing times are depicted in Table 6. Fig. 2 depicts the plot of Demsar's critical difference test. Results of Shaffer post hoc test over accuracy are given in Table 7.

From analyzing the results of comparison between the proposed untrained operators and trained combiners we can derive the following observations.

With respect to accuracy the proposed approaches are not highly inferior to the trained combiners. In 7 out of 20 cases they are even able to deliver better performance than any of the more compound aggregation schemes. This can be explained by the properties of these datasets. They are relatively small with low number of examples per class. Therefore, 5% of the training set delegated to combiner training is not enough to form efficient combination rules. The proposed operators do not suffer from this limitation and are able to efficiently fuse individual outputs of base classifiers.

For remaining datasets we can see that the proposed methods are always able to outperform the BKS combiner. This can be explained by the fact that BKS requires a high number of examples

Table 3
Comparison with untrained classifier combination methods, with respect to their accuracy (%). Small numbers under accuracies stand for indexes of methods, from which the considered one is statistically superior. Last line presents the ranks obtained from the Friedman test.

Dataset	MV[1]	MAX[2]	AVG[3]	PRO[4]	ICA[5]	NP-AVG[6]	NP-MAX[7]
Auslan	80.43	83.85	81.38	79.22	85.18	86.36 1,2,3,4	**87.48** ALL
Autos	62.34	65.84	64.23	63.05	64.98	**67.54** *ALL*	66.32 1,3,4
Car	89.12	89.23	88.43	85.31	88.73	87.74 4	**91.03** *ALL*
Cleveland	52.38	57.23	57.43	55.64	**58.43**	55.02 1	57.14 1,4,6
Dermatology	93.23	95.75	95.05	92.87	93.17	94.67 1	**95.83** 1,4,6
E. coli	71.02	77.43	75.36	71.61	**80.18**	79.62 1,2,3,4,6	77.60 1,3,4
Flare	74.31	72.69	75.72	73.12	75.18	73.90 2,4	**77.12** *ALL*
Isolet	77.92	82.19	81.05	80.22	86.02	85.18 1,2,3,4	**87.81** *ALL*
Led7digit	70.02	70.98	68.34	68.92	70.98	73.19 1,2,3,4,5	**75.28** *ALL*
Letter-2	81.76	84.92	82.77	80.15	83.12	85.14 1,3,4,5	**88.03** *ALL*
Lymphography	82.27	80.32	80.87	79.32	**83.46**	81.12 2,4	80.32 4
Nursery	82.17	82.34	83.05	80.18	81.31	**84.56** *ALL*	83.22 1,4,5
Penbased	93.28	94.58	93.52	92.38	93.37	95.19 1,3,4,5	**96.66** *ALL*
Satimage	79.86	82.03	80.43	78.12	**83.28**	80.75 4	81.25 1,3,4
Segment	86.23	86.74	87.54	85.62	88.42	86.89 3,4	**91.21** *ALL*
Shuttle	93.18	95.32	94.72	92.06	93.06	94.89 1,4	**95.15** 1,4
Vehicle	66.43	74.03	72.63	67.90	69.62	70.12 1,3,4	**73.87** 1,3,4,5,6
Vowel	75.52	76.39	75.30	73.28	75.96	78.24 1,2,3,4,5	**80.03** *ALL*
Yeast	43.41	52.36	49.78	45.02	47.33	50.11 1,3,4,5	**57.98** *ALL*
Zoo	83.22	86.15	84.98	83.06	85.66	**88.14** *ALL*	86.28 1,3,4,5
Avg. ranks	5.60	3.25	4.25	6.55	3.52	3.00	1.83

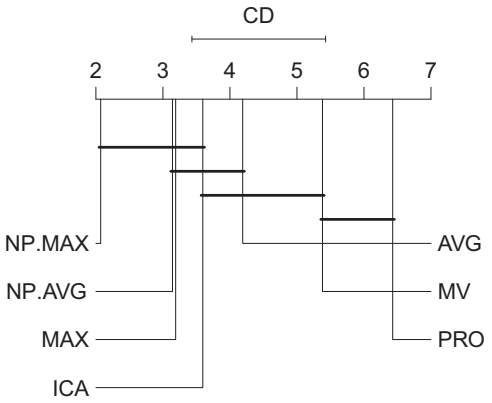


Fig. 1. Demsar's critical difference plot for the proposed untrained fusers with embedded pruning and reference untrained combiners over 20 benchmark datasets.

Table 4
Shaffer test for comparison between the proposed combination methods and reference untrained combiners. Symbol '=' stands for classifiers without significant differences, '+' for situation in which the method on the left is superior and '−' vice versa.

Hypothesis	*p*-Value
NP-AVG vs MV	+ (0.000141)
NP-AVG vs MAX	=(0.714393)
NP-AVG vs AVG	+ (0.047278)
NP-AVG vs PRO	+ (0.000000)
NP-AVG vs ICA	=(0.442178)
NP-MAX vs MV	+ (0.000000)
NP-MAX vs MAX	+ (0.03698)
NP-MAX vs AVG	+ (0.000385)
NP-MAX vs PRO	+ (0.000000)
NP-MAX vs ICA	+ (0.012827)
NP-AVG vs NP-MAX	=(0.085428)

Table 5
Comparison with trained classifier combination methods, with respect to their accuracy (%). Small numbers under accuracies stand for indexes of methods, from which the considered one is statistically superior. Last line presents the ranks obtained from the Friedman test.

Dataset	BKS[1]	DT[2]	NC[3]	NP-AVG[4]	NP-MAX[5]
Auslan	77.98	83.43	84.75	86.36 1,2,3	**87.48** *ALL*
Autos	60.19	63.28	63.62	**67.54** *ALL*	66.32 1,2,3
Car	84.85	91.54	**92.17**	87.74 1	91.03 1,4
Cleveland	47.12	53.34	52.65	55.02 1,2,3	**57.14** *ALL*
Dermatology	92.47	**95.83**	95.18	94.67 1	**95.83** 1
E. coli	68.38	75.31	73.88	**79.62** *ALL*	77.60 1,2,3
Flare	74.12	75.32	**78.04**	73.90 –	77.12 1,2,4
Isolet	82.34	88.02	**88.16**	85.18 1	87.81 1,4
Led7digit	67.58	72.05	70.84	73.19 1,2,3	**75.28** *ALL*
Letter-2	80.28	**88.44**	87.92	85.14 1	88.03 1,4
Lymphography	72.39	76.65	77.09	**81.12** 1,2,3	80.32 1,2,3
Nursery	82.58	**86.83**	85.64	84.56 1,5	83.22 –
Penbased	91.72	**96.90**	95.77	95.19 1	96.66 1,4
Satimage	71.30	**84.58**	83.62	80.75 1	81.25 1
Segment	81.77	**92.14**	91.60	86.89 1	91.21 1,4
Shuttle	92.91	95.02	**95.45**	94.89 1	95.15 1
Vehicle	72.67	74.40	**75.87**	70.12 –	73.87 –
Vowel	77.62	**83.36**	82.18	78.24 –	80.03 1
Yeast	53.49	59.05	**61.38**	50.11 –	57.98 1
Zoo	78.45	84.56	83.89	**88.14** *ALL*	86.28 1,2,3
Avg. rank	4.85	2.27	2.40	3.20	2.28

for computation of meaningful statistics for each combination in the lookup table. Therefore, for the proposed setting where only 5% of training objects are used for training combiners BKS cannot deliver satisfactory performance.

Table 6
Average time complexities (s) of the proposed method and examined classifier combination methods. Measurements include all operations required to prepare the combination block.

Dataset	BKS	DT	NC	NP-AVG	NP-MAX
Auslan	28.34	10.23	16.43	2.18	2.19
Autos	9.34	1.43	3.02	0.65	0.65
Car	21.32	5.73	13.90	1.77	1.77
Cleveland	7.05	2.03	3.23	0.73	0.74
Dermatology	7.21	2.41	3.50	0.90	0.90
E. coli	7.37	2.62	3.74	0.92	0.92
Flare	11.38	4.37	7.22	2.39	2.39
Isolet	37.54	14.36	26.82	5.75	5.75
Led7digit	8.19	2.86	4.03	0.96	0.97
Letter-2	53.86	19.89	30.77	5.93	5.93
Lymphography	4.87	1.02	1.28	0.33	0.33
Nursery	10.15	3.96	6.38	1.84	1.84
Penbased	19.48	8.52	14.07	1.81	1.80
Satimage	7.43	2.20	3.61	0.72	0.72
Segment	25.89	9.32	22.47	2.38	2.40
Shuttle	26.36	9.97	23.16	2.46	2.46
Vehicle	14.72	5.68	11.85	1.26	1.26
Vowel	19.63	8.28	15.76	1.53	1.53
Yeast	23.55	9.30	18.01	2.66	2.66
Zoo	6.54	3.66	4.85	2.54	2.55

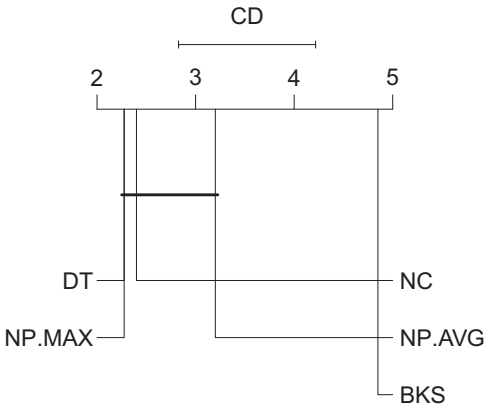


Fig. 2. Demsar's critical difference plot for the proposed untrained fusers with embedded pruning and reference trained combiners over 20 benchmark datasets.

Table 7
Shaffer test for comparison between the proposed combination methods and reference trained combiners. Symbol '=' stands for classifiers without significant differences, '+' for situation in which the method on the left is superior and '−' vice versa.

Hypothesis	*p*-Value
NP-AVG vs KBS	+ (0.000967)
NP-AVG vs DT	− (0.044314)
NP-AVG vs NC	=(0.109599)
NP-MAX vs KBS	+ (0.000000)
NP-MAX vs DT	=(0.999998)
NP-MAX vs NC	=(0.802587)

Remaining two methods (DT and NC) are able to outperform the proposed untrained operators on several datasets, showing that training combination rules may indeed lead to improved adaptation to the considered problem and boosted performance. When taking a look at the statistical tests we can observe that differences between these two trained combiners and proposed untrained operators were found to be statistically insignificant, which further proves the high quality of our methods. However, one must note that by increasing the size of the combiner training

set (the above used 5% of entire training set) we may be able to further boost the performance of trained combination blocks. Yet this can be often unenforceable in many real-life scenarios where amount of training data is limited and we do not have an abundance of objects to spare for improving combination rules.

When comparing averaged computation times we can see that both proposed operators offer rapid procedure for simultaneous weight establishment and ensemble pruning. Obtained times are often several times smaller than in the case of trained combiners. This shows that our methods are able to deliver high quality classification rates with very low computational complexity.

5. Lessons learned

In this section we want to summarize the main findings of our manuscript.

1. *Weighting and pruning*: The proposed untrained operators extend popular AVG and MAX combiners by utilizing weighting module for base classifiers. This way we are able to differentiate influence of base learners in the NP-AVG approach (as models do not have identical influence on the computed final class support), or to utilize information from more than a single classifier in the NP-MAX approach (as the highest support function will have still high weight, but winner-takes-all approach is no longer used). These modifications significantly increase robustness to the influence of incompetent base classifiers in the pool, or to cases where supports for several classes are similar. Additionally, we showed how to extend these combiners with embedded pruning scheme that allows for discarding classifiers with low weights assigned. This way we obtain sparse ensembles and reduce their computational complexities.
2. *Classification accuracy*: The proposed operators offer a statistically significant boost to classification accuracy when compared with their canonical versions. Additionally, they are able to outperform state-of-the-art untrained classifier combination methods. When compared to trainable combination rules our methods still display satisfactory performance. For small datasets they are able to outperform their trainable counterparts, while for remaining cases they are not statistically worser. Of course trained combiners are expected to perform better than untrained ones, but our operators offer better trade-off between accuracy and computational complexity.
3. *Requirements for additional objects*: Trained combiners require a separate set of objects for establishing combination rules. Our untrained operators may take advantage of a very small separate set of objects for optimizing pruning threshold (as seen in Table 2), but may also work without any additional objects. In such cases pruning threshold can be set heuristically. This shows that our combiners are especially useful for small-size datasets or for cases when we do not have an abundance of objects in given time window and we cannot store them to form a separate combiner training set (like in online learning from massive and high-speed data streams).
4. *Time complexity*: Our combiners offer very low computational complexity required for weight calculation and pruning. It is on the same level as in other untrained methods, while being several times smaller than in the case of trained fusers. Our methods, due to their fast computation seem as an attractive proposition for real-life problems with limitations on processing time, e.g., mining non-stationary data streams in which we need to quickly update our ensemble to current state of the concept after occurrence of a drift.

6. Conclusions

The paper presented two novel untrained aggregation operators which could be used in the case of the absence of additional learning material to train the combination rule. Additionally, we show how to directly embed a pruning step into the aggregation schema.

The proposed methods could be valuable alternatives for the traditional aggregating operators which do not require learning and should be used in the mentioned above case instead of voting methods, of course in the case that we can access the support function values of individual classifiers. Additionally, these combiners are attractive for machine learning scenarios where time and complexity factors are crucial.

The computer experiments confirmed that performances of the proposed methods are highly satisfactory compared to the traditionally untrained operators, especially for tasks when the number of possible classes is high. Therefore, we are going to continue the work on the proposed models, especially we would like to apply our combination methods to mining high-speed data streams with concept drift and to one-class classification.

Acknowledgments

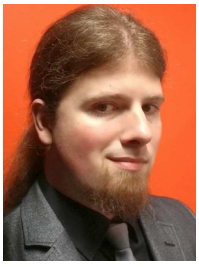
This work was supported by the Polish National Science Center under the Grant no. DEC-2013/09/B/ST6/02264.

All experiments were carried out using computer equipment sponsored by EC under FP7, Coordination and Support Action, Grant Agreement Number 316097, ENGINE—European Research Centre of Network Intelligence for Innovation Enhancement (http://engine.pwr.wroc.pl).

References

[1] Ethem Alpaydin, Combined 5 × 2 cv F test for comparing supervised classification learning algorithms, Neural Comput. 11 (8) (1999) 1885–1892.
[2] Bruno Baruque, Santiago Porras, Emilio Corchado, Hybrid classification ensemble using topology-preserving clustering, New Generat. Comput. 29 (2011) 329–344.
[3] Battista Biggio, Giorgio Fumera, Fabio Roli, Bayesian analysis of linear combiners, in: Proceedings of the 7th International Conference on Multiple Classifier Systems, MCS'07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 292–301.
[4] R. Burduk, Classifier fusion with interval-valued weights, Pattern Recognit. Lett. 34 (14) (2013) 1623–1629.
[5] R. Burduk, K. Walkowiak, Static classifier selection with interval weights of base classifiers, in: Intelligent Information and Database Systems—7th Asian Conference, ACIIDS 2015, Bali, Indonesia, Proceedings, Part I, March 23–25, 2015, pp. 494–502.
[6] C.K. Chow, Statistical independence and threshold functions, IEEE Trans. Electron. Comput. EC-14 (February (1)) (1965) 66–68.
[7] B. Cyganek, Ensemble of tensor classifiers based on the higher-order singular value decomposition, in: Hybrid Artificial Intelligent Systems—Proceedings of the 7th International Conference, HAIS 2012, Salamanca, Spain, Part II, March 28–30, 2012, pp. 578–589.
[8] B. Cyganek, K. Socha, Novel parallel algorithm for object recognition with the ensemble of classifiers based on the higher-order singular value decomposition of prototype pattern tensors, in: VISAPP 2014—Proceedings of the 9th International Conference on Computer Vision Theory and Applications, vol. 2, Lisbon, Portugal, 5–8 January 2014, pp. 648–653.
[9] W.M. Czarnecki, J. Tabor, Two ellipsoid support vector machines, Expert Syst. Appl. 41 (18) (2014) 8211–8224.
[10] Janez Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (18) (2006) 1–30.
[11] Thomas G. Dietterich, Ghulum Bakiri, Solving multiclass learning problems via error-correcting output codes, J. Artif. Intell. Res. 2 (January) (1995) 263–286.
[12] R.P.W. Duin, The combining classifier: to train or not to train? in: 2002 Proceedings of the 16th International Conference on Pattern Recognition, vol. 2, 2002, pp. 765–770.
[13] A. Frank, A. Asuncion. UCI machine learning repository, ⟨http://archive.ics.uci.edu/ml⟩, 2010.

[14] M. Galar, A. Fernández, E. Barrenechea, F. Herrera, DRCW-OVO: distance-based relative competence weighting combination for one-vs-one strategy in multi-class problems, Pattern Recognit. 48 (1) (2015) 28–42.
[15] M. Galar, A. Fernández, E. Barrenechea Tartas, H. Bustince Sola, F. Herrera, Dynamic classifier selection for one-vs-one strategy: avoiding non-competent classifiers, Pattern Recognit. 46 (12) (2013) 3412–3424.
[16] S. García, A. Fernández, J. Luengo, F. Herrera, 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) (2010) 2044–2064.
[17] G. Giacinto, F. Roli, G. Fumera, Design of effective multiple classifier systems by clustering of classifiers, in: 2000 Proceedings of the 15th International Conference on Pattern Recognition, vol. 2, 2000, pp. 160–163.
[18] L.K. Hansen, P. Salamon, Neural network ensembles, IEEE Trans. Pattern Anal. Mach. Intell. 12 (October (10)) (1990) 993–1001.
[19] Tin Kam Ho, The random subspace method for constructing decision forests, IEEE Trans. Pattern Anal. Mach. Intell. 20 (August) (1998) 832–844.
[20] Y.S. Huang, C.Y. Suen, A method of combining multiple experts for the recognition of unconstrained handwritten numerals, IEEE Trans. Pattern Anal. Mach. Intell. 17 (January (1)) (1995) 90–94.
[21] K. Jackowski, Fixed-size ensemble classifier system evolutionarily adapted to a recurring context with an unlimited pool of classifiers, Pattern Anal. Appl. 17 (4) (2014) 709–724.
[22] K. Jackowski, B. Krawczyk, M. Woźniak, Improved adaptive splitting and selection: the hybrid training method of a classifier based on a feature space partitioning, Int. J. Neural Syst. 24 (3) (2014).
[23] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, Geoffrey E. Hinton, Adaptive mixtures of local experts, Neural Comput. 3 (March) (1991) 79–87.
[24] W. Kmiecik, R. Goscien, K. Walkowiak, M. Klinkowski, Two-layer optimization of survivable overlay multicasting in elastic optical networks, Opt. Switch. Netw. 14 (2014) 164–178.
[25] B. Krawczyk, M. Woźniak, Experiments on simultaneous combination rule training and ensemble pruning algorithm, in: 2014 IEEE Symposium on Computational Intelligence in Ensemble Learning, CIEL 2014, Orlando, FL, USA, December 9–12, 2014, pp. 1–6.
[26] B. Krawczyk, M. Woźniak, Untrained method for ensemble pruning and weighted combination, in: Advances in Neural Networks—ISNN 2014—Proceedings of the 11th International Symposium on Neural Networks, ISNN 2014, Hong Kong and Macao, China, November 28–December 1, 2014, pp. 358–365.
[27] L. Kuncheva, J.C. Bezdek, R.P.W. Duin, Decision templates for multiple classifier fusion: an experimental comparison, Pattern Recognit. 34 (2) (2001) 299–314.
[28] L.I. Kuncheva, C.J. Whitaker, Ten measures of diversity in classifier ensembles: limits for two classifiers, in: A DERA/IEE Workshop on Intelligent Sensor Processing (Ref. No. 2001/050), February 2001, pp. 10/1-1010.
[29] I. Ludmila, Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Second ed., Wiley, Hoboken, New Jersey, 2014, ISBN-13: 978-1118315231.
[30] C. Lin, W. Chen, C. Qiu, Y. Wu, S. Krishnan, Q. Zou, Libd3c: ensemble classifiers with a clustering and dynamic selection strategy, Neurocomputing 123 (2014) 424–435.
[31] C.-L. Liu, H. Hao, H. Sako, Confidence transformation for combining classifiers, Pattern Anal. Appl. 7 (1) (2004) 2–17.
[32] R. Lysiak, M. Kurzynski, T. Woloszynski, Optimal selection of ensemble classifiers using measures of competence and diversity of base classifiers, Neurocomputing 126 (2014) 29–35.
[33] Gian Luca Marcialis, Fabio Roli, Fusion of appearance-based face recognition algorithms, Pattern Anal. Appl. 7 (2) (2004) 151–163, http://dx.doi.org/10.1007/s10044-004-0212-7.
[34] Nageswara S.V. Rao, A generic sensor fusion problem: classification and function estimation, in: Fabio Roli, Josef Kittler, Terry Windeatt (Eds.), Multiple Classifier Systems, Lecture Notes in Computer Science, vol. 3077, Springer, 2004, pp. 16–30.
[35] Lior Rokach, Oded Maimon, Feature set decomposition for decision trees, Intell. Data Anal. 9 (March (2)) (2005) 131–158.
[36] J.A. Sáez, M. Galar, J. Luengo, F. Herrera, Analyzing the presence of noise in multi-class problems: alleviating its influence with the one-vs-one decomposition, Knowl. Inf. Syst. 38 (1) (2014) 179–206.
[37] A. Szczurek, M. Badura, K. Banaszkiewicz, M. Maciejewska, T. Marcinkowski, Monitoring volatile organic compound emission based on semiconductor gas sensors, Environ. Eng. Sci. 31 (10) (2014) 533–540.
[38] J. Tabor, P. Spurek, Cross-entropy clustering, Pattern Recognit. 47 (9) (2014) 3046–3059.
[39] Grigorios Tsoumakas, Ioannis Partalas, Ioannis Vlahavas, An ensemble pruning primer, in: Applications of Supervised and Unsupervised Ensemble Methods, Springer, 2009, pp. 1–13.
[40] Kagan Tumer, Joydeep Ghosh, Analysis of decision boundaries in linearly combined neural classifiers, Pattern Recognit. 29 (2) (1996) 341–348.
[41] M. Van Erp, L. Vuurpijl, L. Schomaker, An overview and comparison of voting methods for pattern recognition, in: 2002 Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition, 2002, pp. 195–200.
[42] K. Woods Jr., W.P. Kegelmeyer, K. Bowyer, Combination of multiple classifiers using local accuracy estimates, IEEE Trans. Pattern Anal. Mach. Intell. 19 (4) (1997) 405–410.
[43] M. Woźniak, M. Graña, E. Corchado, A survey of multiple classifier systems as hybrid systems, Inf. Fusion 16 (2014) 3–17.
[44] M. Woźniak, K. Jackowski, Some remarks on chosen methods of classifier fusion based on weighted voting, in: Emilio Corchado, Xindong Wu, Erkki Oja, Alvaro Herrero, Bruno Baruque (Eds.), Hybrid Artificial Intelligence Systems, Lecture Notes in Computer Science, vol. 5572, Springer, Berlin, Heidelberg, 2009, pp. 541–548.
[45] M. Woźniak, M. Zmyślony, Designing combining classifier with trained fuser—analytical and experimental evaluation, Neural Netw. World 20 (7) (2010) 925–934.
[46] L. Xu, A. Krzyzak, C.Y. Suen, Methods of combining multiple classifiers and their applications to handwriting recognition, IEEE Trans. Syst. Man Cybern. 22 (May/June (3)) (1992) 418–435.
[47] X.-C. Yin, K. Huang, C. Yang, H.-W. Hao, Convex ensemble learning with sparsity and diversity, Inf. Fusion 20 (2014) 49–59.
[48] H. Zhang, L. Cao, A spectral clustering based ensemble pruning approach, Neurocomputing 139 (2014) 289–297.



Bartosz Krawczyk received the M.Sc. and Ph.D. degrees in Computer Science with distinctions from the Wrocław University of Technology, Poland, in 2012 and 2015, respectively. Currently he is an Assistant Professor at the Department of Systems and Computer Networks of the same university. His research is focused on machine learning, ensemble learning, one-class classification, class imbalance, data streams and interdisciplinary applications of these methods. He has published 26 international journal papers and over 75 contributions to conferences. Dr. Krawczyk was awarded with numerous prestigious awards for his scientific achievements like IEEE Outstanding Leadership Award, PRELUDIUM and ETIUDA grants from Polish National Science Center, Scholarship of Polish Minister of Science and Higher Education or START award from Foundation for Polish Science among others. He served as a Guest Editor in four journal special issues and organized ten special sessions at international conferences. He is a member of Program Committee for over 40 international conferences and serves as a reviewer for 22 international journals.

Michał Woźniak is a professor of computer science at the Department of Systems and Computer Networks, Wroclaw University of Technology, Poland. He received M.Sc. degree in biomedical engineering from the Wroclaw University of Technology in 1992, and Ph.D. and D.Sc. (habilitation) degrees in computer science in 1996 and 2007, respectively, from the same university. In 2015 he was awarded professor titular by the President of Poland. His research focuses on compound classification methods, hybrid artificial intelligence and medical informatics. Prof. Wozniak has published over 260 papers and three books. His recent one *Hybrid classifiers: Method of Data, Knowledge, and Data Hybridization* was published by Springer in 2014. He has been involved in several research projects related to the above-mentioned topics and has been a consultant of several commercial projects for well-known Polish companies and public administration. Prof. Wozniak is a senior member of the IEEE.