Pattern Anal Applic (2017) 20:981–990
DOI 10.1007/s10044-017-0655-2

ORIGINAL ARTICLE

The deterministic subspace method for constructing classifier ensembles

Michał Koziarski[1] · **Bartosz Krawczyk**[2] · **Michał Woźniak**[1]

Received: 8 April 2017 / Accepted: 26 September 2017 / Published online: 3 October 2017

Abstract Ensemble classification remains one of the most popular techniques in contemporary machine learning, being characterized by both high efficiency and stability. An ideal ensemble comprises mutually complementary individual classifiers which are characterized by the high diversity and accuracy. This may be achieved, e.g., by training individual classification models on feature subspaces. *Random Subspace* is the most well-known method based on this principle. Its main limitation lies in stochastic nature, as it cannot be considered as a stable and a suitable classifier for real-life applications. In this paper, we propose an alternative approach, *Deterministic Subspace* method, capable of creating subspaces in guided and repetitive manner. Thus, our method will always converge to the same final ensemble for a given dataset. We describe general algorithm and three dedicated measures used in the feature selection process. Finally, we present the results of the experimental study, which prove the usefulness of the proposed method.

Keywords Machine learning · Classifier ensemble · Feature subspaces · Classifier diversity

✉ Michał Koziarski
michal.koziarski@pwr.edu.pl

Bartosz Krawczyk
bkrawczyk@vcu.edu

Michał Woźniak
michal.wozniak@pwr.edu.pl

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

[2] Department of Computer Science, School of Engineering, Virginia Commonwealth University, 401 West Main Street, P.O. Box 843019, Richmond, VA 23284-3019, USA

1 Introduction

Contemporary machine learning has to deal with escalating complexity of problems appearing with the increasing prevalence of data. Standard classification methods are often unable to capture complex patterns or cannot maintain their generalization capabilities, leading to either under- or overfitting. Therefore, models that can capture multi-dimensional data properties while avoiding mentioned pitfalls are very desirable. Classifier ensemble, known also as multiple classifier system or classifier committee [37], is an example of such an approach. By combining predictions of a number of simpler models, the classifier ensembles can produce a more efficient and flexible recognition systems, at the same time benefiting from the high generality of the base classifiers [17].

To ensure a satisfactory performance, several requirements have to be met by an ensemble. Perhaps the most important one is the need to supply a pool of a diverse classifiers [34]. This could be achieved in a various ways, e.g., by training every learner on the basis of a different features. The motivation behind this approach is twofold: firstly, to simplify the training procedure by reducing the number of features used by each learner and at the same time allowing the learner to explore different properties of the supplied subspaces [11]. The most notable examples of techniques relying on partitioning features into subspaces include *Random Subspace* (RS) [15] and *Random Forest* (RF) [5] methods. While simple and computationally efficient, such approaches have a major drawback. Due to the random nature of the feature subset construction process, they are inherently unstable, prone to producing models of a poor quality in the worst-case scenarios. Additionally, for each run of these algorithms one will obtain different feature subsets and thus different base classifiers. It should

be underlined that in the real-life applications the stability is often a necessary component, which significantly limits the use of the mentioned methods, particularly the *Random Subspace* approach.

To overcome this limitation, we consider how the feature subspaces can be constructed in a guided way, in order to make the resulting classifier ensemble more stable and less prone to producing under-performing base learners. As a result of our investigation, we present a novel approach, which allows to form feature subspaces in a fully deterministic manner. It is based on the same idea as the previously mentioned methods, namely creating an ensemble of a simpler, diverse classifiers, each trained on a feature subset. However, instead of a stochastic subspace generation procedure, the proposed approach employs a guided search strategy. It assigns the new features greedily in a round-robin fashion, based on the defined feature quality and the subspace diversity measures. The final decision of the ensemble is made using the majority voting rule. Proposed method is as flexible as the *Random Subspace* approach and can work with any type of base learners. However, the created ensemble is more stable and uses the information about the predictive power of individual features.

The main contributions of the work are as follows:

- Proposition of the new classifier ensemble learning algorithm, *Deterministic Subspace* (DS), which allows forming a set of diverse classifiers trained on selected features, where the attributes used by the base classifiers are being chosen by a guided search strategy.
- Improvement over random subspace selection, leading to stable ensemble forming procedure.
- Extensive experimental results on a set of benchmark datasets, which evaluate the dependency among the quality of DS algorithm, its parameters and chosen model of base classifiers.
- Proving the high usefulness of the proposed DS approach for specific classifiers that take advantage of creating less correlated subspaces.

The rest of this paper is organized as follows. Section 2 presents related works on the subject of ensemble classification. In Sect. 3, *Deterministic Subspace* method is discussed. Section 4 describes conducted experimental study and obtained results. Finally, Sect. 5 summarizes our findings.

2 Related work

Ensemble classification methods have several properties that make them one of the most prevalent techniques in supervised learning domain. Perhaps most importantly, models produced using this paradigm tend to be capable of approximating complex decision boundaries while remaining resilient to overfitting. Performance gain is possible by exploiting local competencies of base learners. At the same time, ensembles incorporate mechanisms that prevent from choosing the worst learner from the pool [20]. Additionally, this approach is highly flexible. The parameters of learners are often easily adjustable, and the training procedure can be parallelized without much effort.

Even though ensembles solve some of the issues related to the classification problems, at the same time they introduce several new challenges related to their design. Necessity of providing a diverse pool of learners while preserving their individual accuracies [36] is the one on which we will focus in this paper. This issue is especially severe due to the ambiguity of this term. Proposing an agreed upon definition of diversity remains an important open question [6], particularly in context of classification task. Existing diversity measures are therefore only approximations. The exact extent of diversity influence on ensemble performance remains unclear [12]. At the same time, it is hard to argue that learners should display some differing characteristics, since adding identical models would not contribute to efficacy of formed ensemble.

Diversity may be introduced to the ensemble on many different levels, often as a combination of factors. Main approaches include:

- Varying learning models by using either completely different classification algorithms or the same algorithms with modified hyperparameters.
- Varying outputs of learners by decomposing classification task, for instance into binary problems.
- Varying inputs of learners by supplying different partitions of dataset or different feature subspaces during training.

In the case of heterogeneous ensembles, one assumes that using different learners will be sufficient to ensure diversity. Indeed, in many situations modifying learning paradigm may lead to obtaining significantly different decision boundaries. Selection process is crucial when applying this approach, to reduce the chance of using similar outputs produced by different models. Entire family of dynamic classifier selection methods is worth mentioning, as they offer a flexible ensemble line-up for each incoming sample [33].

Alternatively, instead of using entirely different models one may rather train them using different set of hyperparameters or initial conditions. This approach is based on the assumption of existence of complex search space during the training, which could lead to reaching different local extrema. Most notable examples of this method include ensembles of neural networks with early stopping condition [35] or *Support Vector Machines* with varying kernels [32].

Another approach is based on manipulating the classifier outputs. Common techniques rely on a multi-class decomposition, after which specialized learners, trained to recognize reduced number of classes, are obtained. Dedicated combination method, such as *Error-Correcting Output Codes* [25], is being used to reconstruct original multi-class task. Most notable examples of this type of technique include binarization [13], hierarchical decomposition [27] and classifier chains [22].

Diversity may be also induced by input manipulation in either data or feature space. The former approach relies on assumption that variance is being introduced into the training instances, which enables learners to capture properties of different subsets of objects [29]. *Bagging* [30] and *Boosting* [2] are most significant realizations of this approach, but ensembles may also be trained on the basis of clusters to preserve spatial relations among instances [10].

Finally, one may increase the diversity by manipulating the feature space. This could be done in either randomized [21] or guided manner, using feature selection [8] or global optimization methods [7, 24]. The most notable techniques based on this paradigm are previously mentioned *Random Subspace* and *Random Forest* methods. Both of them rely on randomly created feature subspaces to increase the diversity of the ensemble.

Despite its simplicity, the *Random Subspace* method has gained a popularity in the machine learning community. Its main advantage over *Random Forest* lies in flexibility, as it can be used with an any type of base learners [31]. There are several interesting variations of this approach that appeared in recent years. Polikar et al. [28] proposed Learn^{++}.MF, a modification of popular Learn^{++} algorithm that utilized *Random Subspaces* in order to handle missing values in classified instances. In case of incomplete information, only classifiers trained on available features were used for the classification phase. Li et al. [18] used *Random Subspaces* together with distance-based lazy learners and combined them using Dempster's rule. Mert et al. [23] developed a weighted combination of *Random Subspaces*, where weight assigned to each of them was based on their ability to provide a good class separability. *Random Subspaces* have also been successfully used for semi-supervised learning. Yaslan and Cataltepe [39] used randomized set of features for co-training an ensemble of classifiers, while Yu et al. [40] used them for a semi-supervised dimensional reduction using graphs. Recent work by Carbonneau et al. [9] proved that this method offers very good performance in multi-instance learning. Another reason behind popularity of *Random Subspaces* approach lies in many successful applications of this technique to solving real-life problems. Plumpton et al. [26] used it for real-time classification of fMRI data, Xia et al. [38] for hyperspectral image analysis, while Zhu and Xue [41] combined it with tensor analysis for face recognition.

However, significant drawback of *Random Subspace* approach (as well as of *Random Forest*) lies in its purely random nature. Therefore, it lacks any deterministic element that would allow to maintain stability or ensure that the same model will be trained for given data in every repetition. This is especially crucial for real-life applications, where a final model is required for some purposes, e.g., being embedded in a hardware unit. Some researchers tried to improve the stability of this method [19]; however, no fully deterministic solution was proposed so far.

3 Deterministic subspace method

Due to the random subspace creation procedure, RS and RF are conceptually simple and computationally efficient methods. However, there is a probability that due to their stochastic components, the produced subspaces may lack the discriminative power necessary for a proper separation of classes. Additionally, even if individually strong subspaces are created, the lack of diversity among them may deteriorate performance of the ensemble. In that sense, these methods may be viewed as unstable. Furthermore, random techniques could be somewhat unsatisfying: Even if they produce highly accurate models, it might be unclear what make them good.

To overcome these limitations, we propose an alternative to the RS method, a *Deterministic Subspace* (DS) approach. Our main goal here is to offer a stable and deterministic substitute for the RS method. In this section, a detailed description of the algorithm will be given, along with a discussion of several feature quality and subspace diversity measures that may be used as its components.

3.1 Algorithm

The proposed algorithm is based on the idea of creating subspaces incrementally, in a manner guided by both the quality of individual subspaces and the diversity of the whole ensemble. The preference toward either quality or diversity can be adjusted by modifying the algorithms hyperparameter α. For the approach to be computationally feasible, we had to make several simplifications. Firstly, we create the subspaces in a greedy manner based on a round-robin strategy, which may produce a non-optimal solution. Secondly, we make a strong assumption that a subspace consisting of individually strong features is itself of a high quality. This assumption does not have to hold in practice; in fact, it can be easily shown that two weak features can together have a high discriminant power [14]. However, it was necessary to make training on a highly dimensional data feasible. The proposed algorithm has three parameters:

- the number of subspaces to be created k,
- the number of features selected for every subspace n,
- the weight coefficient α, indicating preference toward either the feature quality or the diversity.

Lower values of α lead to creation of more diverse subspaces, with features allocated close to evenly among them. On the other hand, by choosing a higher value we force the algorithm to pick the individually strong features more often. Setting α to 0 would make the algorithm disregard feature quality completely, whereas setting it to 1 would result in creation of a single subspace, consisting of individually strongest features.

Smaller number of features per subspace n should, in principle, result in producing weaker base learners. Additionally, the subspaces created in that case are more diverse, since there is less overlap between their features. Larger number of subspaces k leads to creation of bigger ensemble, at the same time decreasing the diversity of the subspaces.

3.2 Diversity measure

It is intuitive that increasing the classifier ensemble diversity should lead to a better accuracy, but on the other hand there is no formal proof of this dependency [4]. Several different approaches to measuring the diversity of classifier ensemble have been proposed in the existing literature. However, most of them rely on predictions made by classifiers [3] and as a result are computationally expensive. We propose a naive, yet fast approach based on measuring evenness of the feature spread among the subspaces.

Let S denote the set of the existing subspaces and S_j stand for the jth subspace. Let $\mathcal{X}$ be a set of the available features $\mathcal{X} = \{x^{(1)}, x^{(2)}, \ldots, x^{(d)}\}$. Consider inserting additional feature $x^{(c)}$ into the currently considered subspace S_j. We define a diversity metric $div\_m(S, S_j, x^{(c)})$ as an average of two components: the proportion of existing subspaces already containing the considered feature $div\_m_x(S, x^{(c)})$ and the distance to the most similar subspace $div\_m_s(S, S_j)$:

$$div\_m\left(S, S_j, x^{(c)}\right) = \frac{div\_m_x\left(S, x^{(c)}\right) + div\_m_s(S, S_j)}{2}, \tag{1}$$

where:

$$div\_m_x\left(S, x^{(c)}\right) = 1 - \frac{\left|\left\{S_j : x^{(c)} \in S_j\right\}\right|}{|S|}, \tag{2}$$

and:

$$div\_m_s(S, S_j) = 1 - \max_{j \neq l} \frac{\left|S_j \cap S_l\right|}{|S_j|}. \tag{3}$$

By minimizing the proposed metric, we ensure that the features are spread evenly among the subspaces, which should contribute toward creation of a diverse set of learners. We make an underlying assumption that large groups of features are not highly correlated, in which case the proposed dissimilarity would be too simplistic. In practice, the situations that would lead to a complete failure of the proposed metric are very rare.

Table 1 Details of datasets used throughout the experiment

No.	Name	Features	Objects	Classes
1	winequality	11	6497	11
2	vowel	13	990	11
3	vehicle	18	846	4
4	segment	19	2310	7
5	ring	20	7400	2
6	thyroid	21	7200	3
7	mushroom	22	5644	2
8	chronic kidney	24	157	2
9	wdbc	30	569	2
10	ionosphere	33	351	2
11	dermatology	34	358	6
12	texture	40	5500	11
13	biodegradation	41	1055	2
14	spectfheart	44	267	2
15	spambase	57	4597	2
16	sonar	60	208	2
17	splice	60	3190	3
18	optdigits	64	5620	10
19	mice protein	80	552	8
20	coil2000	85	9822	2
21	movement libras	90	360	15

Table 2 Values of base classifiers hyperparameters

Classifier	Parameters
kNN	$k = 5$
SVM	kernel = linear, $C = 1.0$
ParzenKDE	window size = 1.0
NNKDE	kernel = Gaussian, bandwidth = 1.0
GMM	covariance = diagonal, iterations = 100

3.3 Quality measures

As mentioned before, the estimation of subspace quality is based on the strength of the individual predictors. Using only the individually strong features not necessarily will improve the discriminative power of the subspace, or even more so of the whole ensemble. However, we claim that reducing frequency of the occurrence of the weak predictors will, on average, result in an increased performance.

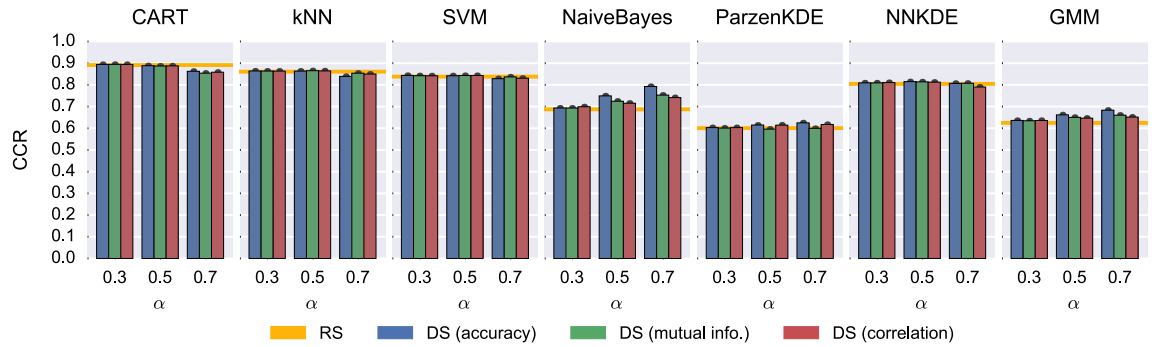

Fig. 1 Correct classification rates (CCR) averaged over all datasets and examined number of subspaces for *Random Subspace* and *Deterministic Subspace* algorithms. Three different feature quality measures were considered in combination with *Deterministic Subspace* method, namely accuracy, mutual information and correlation

Let us denote the ith class label, encoded as an integer, as $i \in \mathcal{M} = \{1, 2, \ldots, M\}$. Furthermore, let $\mathcal{LS} = \{(x_1, i_1), (x_2, i_2), \ldots, (x_n, i_n)\}$ be the learning set consisting of n observations, $\overline{x}^{(c)} = [x_1^{(c)}, x_2^{(c)}, \ldots, x_n^{(c)}]$ be the vector of observations of feature $x^{(c)}$, and $\overline{i} = [i_1, i_2, \ldots, i_n]$ be the vector of class labels associated with observations. We define the classification accuracy on kth fold, obtained by using a single feature $x^{(c)}$, as $Acc(x^{(c)}, k)$. Marginal probabilities of $\overline{x}^{(c)}$ and $\mathcal{M}$ as $p(x_j^{(c)})$ and $p(i)$, respectively, and their joint probability as $p(x_j^{(c)}, i)$. Covariance of $\overline{x}^{(c)}$ and $\overline{i}$ as $\mathrm{cov}(\overline{x}^{(c)}, \overline{i})$, and standard deviation as $\sigma_{\overline{x}^{(c)}}$ and $\sigma_{\overline{i}}$. We propose three different measures. First and foremost, a twofold cross-validation accuracy on the training data $qual\_m_{acc}(x_c)$ was obtained while training on the individual features:

$$qual\_m_{acc}(x_c) = \frac{1}{2} \sum_{k=1}^{2} Acc(x_c, k). \tag{4}$$

It provides a conceptually simple metric with an important property of being adaptable to the type of chosen learner. However, depending on the dimensionality of the data it might require training a large number of classifiers. Because of that, we propose two alternative measures.

The first one is the mutual information between the feature and the target $qual\_m_{mi}(x_c)$:

$$qual\_m_{mi}(x_c) = \sum_{i=1}^{M} \sum_{j=1}^{n} p(x_j^{(c)}, i) \log \left(\frac{p(x_j^{(c)}, i)}{p(x_j^{(c)})\, p(i)} \right), \tag{5}$$

while the second one is the population Pearson correlation between the cth feature and the labels:

$$qual\_m_{corr}(x_c) = \frac{\mathrm{cov}(\overline{x}^{(c)}, \overline{i})}{\sigma_{\overline{x}^{(c)}} \sigma_{\overline{i}}}. \tag{6}$$

Because the probability characteristics of the classification tasks are usually unknown, therefore we use the appropriate estimators as the sample correlation coefficient, which is used to estimate the population Pearson correlation.

The idea behind using the proposed measures is to accelerate the learning process without a significant loss of the accuracy.

The pseudocode of the *Deterministic Subspace* algorithm is presented in Algorithm 1.

Algorithm 1 Deterministic Subspace algorithm

Input: set of features $\mathcal{X} = \{x^{(1)}, x^{(2)}, \ldots, x^{(d)}\}$
Parameters: number of subspaces k, number of features per subspace n, weight coefficient α, feature quality measure $qual\_m(x^{(c)})$
Output: feature subspaces S
for $i = 1$ to k **do**
  $S_i \longleftarrow \emptyset$
end for
repeat
  for $i = 1$ to k **do**
    for $c = 1$ to d **do**
      if $x_c \notin S_i$ **then**
        $f_{score}(x^{(c)}) \longleftarrow \alpha \times qual\_m(x^{(c)}) + (1 - \alpha) \times div\_m(S, S_i, x^{(c)})$
      end if
    end for
    $x_{best} \longleftarrow \mathrm{argmax}_{x^{(c)}} f_{score}(x^{(c)})$
    $S_i \longleftarrow S_i \cup x_{best}$
  end for
until every subspace consists of n features
return S

Fig. 2 Correct classification rates for specific datasets and base learners. The accuracy was used as a feature quality measure

4 Experimental study

In this section, we present a detailed description of the conducted experimental study and perform an analysis of the obtained results. The main goal was to evaluate whether proposed deterministic approach is capable of achieving at least as high accuracy as the RS method. Secondly, we tried to establish whether, and if so under what conditions, DS can actually outperform the RS method. Finally, we compared the both approaches with another popular method relying on the creation of a random feature subspaces, *Random Forest*.

4.1 Set-up

All experiments were implemented in Python programming language; code sufficient to repeat them was made publicly available at.[1] Whenever possible, the existing implementations of the classification algorithms from the scikit-learn machine learning library[2] were used to limit the possibility of a programming errors.

Performance of the considered algorithms was evaluated on the basis of 21 benchmark datasets with varying number of objects and features. All of them were taken from the UCI[3] and the KEEL[4] repositories and, as such, are publicly available. The detailed parameters of the datasets are presented in Table 1. For every dataset, 5×2-fold partitions were randomly created and used during the experiments. These partitions are available together with the code.

Seven classifiers were evaluated during the experiments, namely CART, k-nearest neighbors (kNN), linear support vector machine (SVM), Naïve Bayes, Parzen window kernel density estimation (ParzenKDE), nearest neighbor kernel density estimation (NNKDE) and Gaussian mixture model (GMM). First four were tested in the initial stage of the experiment, with partial results published in [16], and were chosen to cover different types of algorithms. After obtaining the results, the remaining three classifiers were evaluated to establish whether trends observable for Naïve Bayes extend to other types of nonparametric classifiers.

The hyperparameters specific for the particular classification methods were constant throughout the experiments. Whenever possible, their default values provided in the corresponding scikit-learn modules were used. The most significant parameters are presented in Table 2. Different numbers of subspaces $k \in \{5, 10, \ldots, 50\}$ were evaluated in all cases. Number of features per subspace used by RS and DS methods was fixed at half the total number of the features. Additionally, the quality coefficients $\alpha \in \{0.0, 0.1, \ldots, 1.0\}$ and three different quality metrics, namely twofold cross-validation accuracy on training set, mutual information between the features and the labels, and absolute value of correlation between the two were tested for the DS approach. Finally, varying number of trees $\in \{5, 10, ..., 50\}$ was used in combination with the *Random Forest* method.

4.2 Results

Average accuracy of proposed method for different classification algorithms, quality measures and quality coefficients α is presented in Fig. 1. Only selection of α parameters was shown to improve the clarity of the presentation. Most significant changes in the algorithms behavior were observed for the selected values. The classification accuracy of the RS method is used as a baseline. The average scores for specific base learners and datasets are to be found in Fig. 2, where accuracy was used as a feature quality metric. Results of the combined 5×2 cross-validation F test [1] are depicted in Fig. 3. It presents, for all k and α parameters, difference between the number of datasets on which proposed method achieved statistically significantly better and worse results than its random counterpart. Finally, the average rankings obtained from Friedman N × N test are given in Table 3. Once again, only a subset of considered α parameters and a single quality measure, twofold cross-validation accuracy, was presented for clarity. In addition to deterministic and random feature subspace methods, performance of *Random Forest* classifier was also reported in this step. The complete results of the experimental study can be found at.[5]

4.3 Discussion

Presented results indicate that DS can achieve not only similar performance as random methods but also, depending on the type of classifier used, significantly outperform it. Observed accuracy gain was especially high with certain types of nonparametric classifiers, namely Naïve Bayes, Gaussian mixture models and Parzen kernel density estimation. In all of these cases, large values of α returned the best performance, which corresponds to favoring individually strong features over higher diversity. This may be caused by the nature of considered classifiers that assume low or no correlation among features and thus directly benefit from the way DS algorithm creates feature subspaces for its base learners.

In the second group of classifiers consisting of CART, k-nearest neighbors, SVM and nearest neighbors kernel density estimation choosing large values of α actually led to

[1] github.com/michalkoziarski/DeterministicSubspace.

[2] scikit-learn.org/stable.

[3] archive.ics.uci.edu/ml/datasets.html.

[4] sci2s.ugr.es/keel/datasets.php.

[5] mkoziarski.com/deterministic-feature-subspace-method.

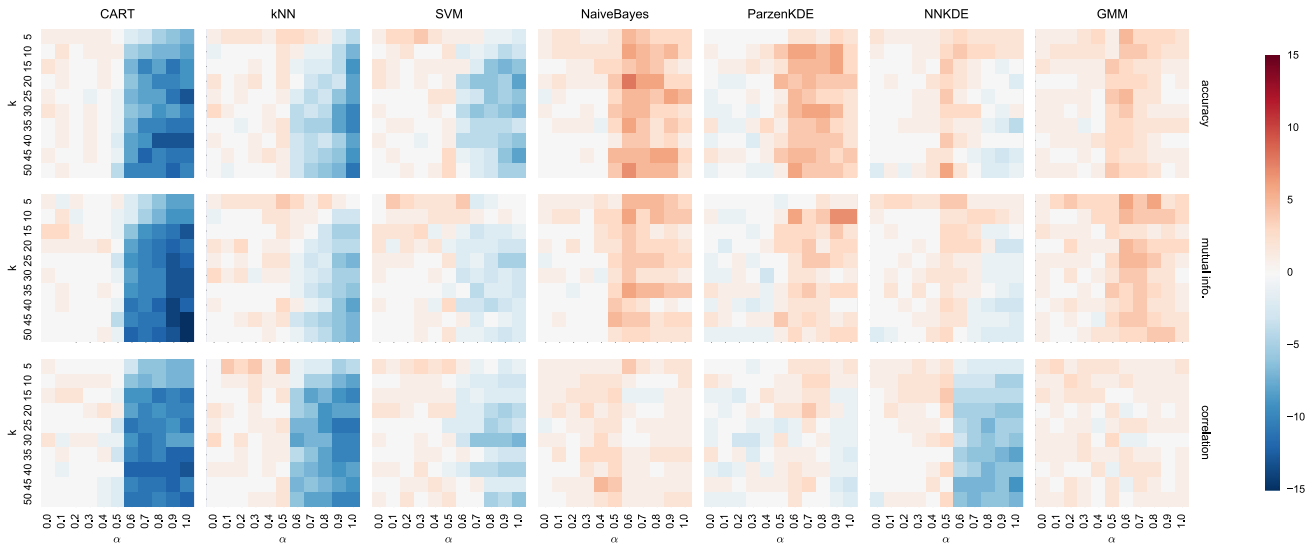

Fig. 3 Differences between the number of datasets on which the *Deterministic Subspace* algorithm achieved statistically significantly better (positive values) and worse (negative values) results than the *Random Subspace* algorithm

performance drop. Behavior of proposed method was more stable with small values of α: For all tested classifiers, average rank was slightly higher than with RS method in that setting. Most notably, when combined with decision tree, proposed method achieved highest average rank for a particular choice of α, higher than *Random Forest* classifier.

Despite lower computational complexity, alternative measures of feature quality, namely mutual information and correlation, resulted on average in degradation of performance.

Overall, the proposed method operated in two distinguishable modes. The first one, in which small values of α parameter were applied, presents comparable alternative to RS approach offering slightly higher performance and sought-after stability. The second one, with setting α to higher values, requires greater care but can lead to significantly better results when combined with particular types of classifiers.

5 Conclusions and future work

The novel classification method *Deterministic Subspace* based on the feature subspaces was proposed and evaluated throughout this study. During the experiments, we established that it presents stable alternative to the *Random Subspace* approach, also capable of outperforming the *Random Forest* method with a proper choice of classifiers. However, in contrast to the random methods, the *Deterministic Subspace* algorithm always returns the same model for a given learning dataset. Additionally, we observed that the proposed method can significantly outperform the random approach when used in combination with some types of classifiers.

The main limitation of the proposed algorithm lies in its high computational complexity. Subspace creation procedure can take significantly greater amount of time compared to random approaches, especially when the number of features is large. In the future, we plan to improve the computational efficiency of our method in order to offer a speedup of the training process. Additionally, two less computationally expensive feature quality metrics were proposed in the course of this paper to try to remedy that issue, but at the expense of the classification accuracy. Finding better estimators of the feature quality could significantly improve practical usefulness of the proposed method. Furthermore, the possibility of parallelization of the algorithm could be investigated to make the algorithm more suitable for larger datasets.

Additionally, the exact conditions under which *Deterministic Subspace* method is capable of achieving significantly higher performance remain unknown. Determining what types of classifiers benefit more from individual feature quality than the diversity of produced subspaces was done partially in this paper; more extensive evaluation would be, however, necessary. This remains for the further study.

Table 3 Average rankings of the algorithms

Position	Algorithm	Ranking
1	DS(CART, 0.3)	5.7619
2	RandomForest	6.4286
3	RS(CART)	7.4762
4	DS(CART, 0.5)	8
5	DS(SVM, 0.3)	9.5238
6	DS(SVM, 0.5)	9.6429
7	DS(kNN, 0.5)	9.8571
8	DS(kNN, 0.3)	9.9048
9	RS(SVM)	10.7619
10	RS(kNN)	10.9524
11	DS(CART, 0.7)	11.7143
12	DS(SVM, 0.7)	12.9762
13	DS(kNN, 0.7)	13.1905
14	DS(NNKDE, 0.5)	14.1429
15	DS(NNKDE, 0.3)	14.381
16	RS(NNKDE)	15.7143
17	DS(NNKDE, 0.7)	15.8571
18	DS(NaiveBayes, 0.7)	16.8571
19	DS(NaiveBayes, 0.5)	17.5714
20	DS(NaiveBayes, 0.3)	19.3333
21	DS(ParzenKDE, 0.7)	20.2619
22	RS(NaiveBayes)	20.4762
23	DS(ParzenKDE, 0.5)	20.5952
24	DS(GMM, 0.5)	21.4286
25	DS(ParzenKDE, 0.3)	21.7857
26	DS(GMM, 0.7)	21.8095
27	RS(ParzenKDE)	22.0714
28	DS(GMM, 0.3)	22.4762
29	RS(GMM)	24.0476

Random Subspace (RS), *Deterministic Subspace* (DS) with the accuracy as a feature quality measure and *Random Forest* algorithms were considered. Name of the base learner was specified for both RS and DS methods. Additionally, the value of the feature quality coefficient α was specified for the DS

Acknowledgements This work was supported by the Polish National Science Center under the grant no. UMO-2015/19/B/ST6/01597 as well as the PLGrid Infrastructure.

References

1. Alpaydin E (1999) Combined 5 x 2 cv F test for comparing supervised classification learning algorithms. Neural Comput 11(8):1885–1892
2. Álvarez PM, Luengo J, Herrera F (2016) A first study on the use of boosting for class noise reparation. In: Proceedings of the 11th international conference on hybrid artificial intelligent systems, HAIS 2016, Seville, Spain, 18–20 April 2016, pp 549–559
3. Banfield RE, Hall LO, Bowyer KW, Kegelmeyer WP (2005) Ensemble diversity measures and their application to thinning. Inf Fusion 6(1):49–62
4. Bi Y (2012) The impact of diversity on the accuracy of evidential classifier ensembles. Int J Approx Reason 53(4):584–607
5. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
6. Brown G, Wyatt J, Harris R, Yao X (2005) Diversity creation methods: a survey and categorisation. Inf Fusion 6(1):5–20
7. Cano A, García-Martínez C, Ventura S (2017) Extremely high-dimensional optimization with mapreduce: scaling functions and algorithm. Inf Sci 415:110–127
8. Canuto AMP, Vale KMO, Neto AF, Signoretti A (2012) Reinsel: a class-based mechanism for feature selection in ensemble of classifiers. Appl Soft Comput 12(8):2517–2529
9. Carbonneau M, Granger E, Raymond AJ, Gagnon G (2016) Robust multiple-instance learning ensembles using random subspace instance selection. Pattern Recognit 58:83–99
10. Cyganek B (2012) One-class support vector ensembles for image segmentation and classification. J Math Imaging Vis 42(2–3):103–117
11. Czarnecki WM, Józefowicz R, Tabor J (2015) Maximum entropy linear manifold for learning discriminative low-dimensional representation. In: Proceedings of the European conference on machine learning and knowledge discovery in databases, ECML PKDD 2015, Part I, Porto, Portugal, 7–11 Sept 2015, pp 52–67
12. Didaci L, Fumera G, Roli F (2013) Diversity in classifier ensembles: fertile concept or dead end? In: International workshop on multiple classifier systems. Springer, pp 37–48
13. Galar M, Fernández A, Barrenechea E, Herrera F (2015) DRCW-OVO: distance-based relative competence weighting combination for one-vs-one strategy in multi-class problems. Pattern Recognit 48(1):28–42
14. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
15. Ho TK (1998) The random subspace method for constructing decision forests. IEEE Trans Pattern Anal Mach Intell 20(8):832–844
16. Koziarski M, Krawczyk B, Woźniak M (2016) Forming classifier ensembles with deterministic feature subspaces. In: Ganzha M, Maciaszek L, Paprzycki M (eds) Proceedings of the 2016 federated conference on computer science and information systems. Annals of computer science and information systems, vol 8. IEEE, pp 89–95. doi:10.15439/2016F552
17. Krawczyk B, Woźniak M (2016) Untrained weighted classifier combination with embedded ensemble pruning. Neurocomputing 196:14–22
18. Li H, Wen G, Yu Z, Zhou T (2013) Random subspace evidence classifier. Neurocomputing 110:62–69
19. Liu Z, Yang Z, Liu S, Shi Y (2013) Semi-random subspace method for writeprint identification. Neurocomputing 108:93–102
20. Marcialis GL, Roli F (2003) Fusion of face recognition algorithms for video-based surveillance systems. Springer, Boston, pp 235–249. doi:10.1007/978-1-4615-0371-2_13
21. Maudes J, Diez JJR, García-Osorio CI, García-Pedrajas N (2012) Random feature weights for decision tree ensemble construction. Inf Fusion 13(1):20–30

22. Melki G, Cano A, Kecman V, Ventura S (2017) Multi-target support vector regression via correlation regressor chains. Inf Sci 415:53–69
23. Mert A, Kiliç NZ, Bilgili E (2016) Random subspace method with class separability weighting. Expert Syst 33(3):275–285
24. Nag K, Pal NR (2016) A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification. IEEE Trans Cybern 46(2):499–510
25. Özögür-Akyüz S, Windeatt T, Smith RS (2015) Pruning of error correcting output codes by optimization of accuracy-diversity trade off. Mach Learn 101(1–3):253–269
26. Plumpton CO, Kuncheva LI, Oosterhof NN, Johnston SJ (2012) Naive random subspace ensemble with linear classifiers for real-time classification of fMRI data. Pattern Recognit 45(6):2101–2108
27. Podolak IT, Roman A (2013) Theoretical foundations and experimental results for a hierarchical classifier with overlapping clusters. Comput Intell 29(2):357–388
28. Polikar R, DePasquale J, Mohammed HS, Brown G, Kuncheva LI (2010) Learn $^{++}$.mf: a random subspace approach for the missing feature problem. Pattern Recognit 43(11):3817–3832
29. Porwik P, Orczyk T, Lewandowski M, Cholewa M (2016) Feature projection k-nn classifier model for imbalanced and incomplete medical data. Biocybern Biomed Eng 36(4):644–656. doi:10.1016/j.bbe.2016.08.002
30. Rokach L (2016) Decision forest: twenty years of research. Inf Fusion 27:111–125
31. Skurichina M, Duin RPW (2002) Bagging, boosting and the random subspace method for linear classifiers. Pattern Anal Appl 5(2):121–135
32. Sun T, Jiao L, Liu F, Wang S, Feng J (2013) Selective multiple kernel learning for classification with ensemble strategy. Pattern Recognit 46(11):3081–3090
33. Trajdos P, Kurzynski M (2016) A dynamic model of classifier competence based on the local fuzzy confusion matrix and the random reference classifier. Appl Math Comput Sci 26(1):175
34. Wang S, Yao X (2013) Relationships between diversity of classification ensembles and single-class performance measures. IEEE Trans Knowl Data Eng 25(1):206–219
35. West D, Dellana S, Qian J (2005) Neural network ensemble strategies for financial decision applications. Comput Oper Res 32(10):2543–2559
36. Windeatt T (2006) Accuracy/diversity and ensemble MLP classifier design. IEEE Trans Neural Netw 17(5):1194–1211
37. Woźniak M, Graña M, Corchado E (2014) A survey of multiple classifier systems as hybrid systems. Inf Fusion 16:3–17
38. Xia J, Mura MD, Chanussot J, Du P, He X (2015) Random subspace ensembles for hyperspectral image classification with extended morphological attribute profiles. IEEE Trans Geosci Remote Sens 53(9):4768–4786
39. Yaslan Y, Cataltepe Z (2010) Co-training with relevant random subspaces. Neurocomputing 73(10–12):1652–1661
40. Yu G, Zhang G, Domeniconi C, Yu Z, You J (2012) Semi-supervised classification based on random subspace dimensionality reduction. Pattern Recognit 45(3):1119–1135
41. Zhu Y, Xue J (2017) Face recognition based on random subspace method and tensor subspace analysis. Neural Comput Appl 28(2):233–244