On Deriving the Second-Stage Training Set for Trainable Combiners

Pavel Paclík, Thomas C.W. Landgrebe, David M.J. Tax, and Robert P.W. Duin

Information and Communication Theory Group,
TU Delft, The Netherlands
{P.Paclik, T.C.W.Landgrebe, D.M.J.Tax, R.P.W.Duin}@ewi.tudelft.nl

Abstract. Unlike fixed combining rules, the trainable combiner is applicable to ensembles of diverse base classifier architectures with incomparable outputs. The trainable combiner, however, requires the additional step of deriving a second-stage training dataset from the base classifier outputs. Although several strategies have been devised, it is thus far unclear which is superior for a given situation. In this paper we investigate three principal training techniques, namely the re-use of the training dataset for both stages, an independent validation set, and the stacked generalization. On experiments with several datasets we have observed that the stacked generalization outperforms the other techniques in most situations, with the exception of very small sample sizes, in which the re-using strategy behaves better. We illustrate that the stacked generalization introduces additional noise to the second-stage training dataset, and should therefore be bundled with simple combiners that are insensitive to the noise. We propose an extension of the stacked generalization approach which significantly improves the combiner robustness.

1 Introduction

When designing pattern recognitions systems, it is often the case that multiple types of data representation and classifiers may be exploited. Instead of selecting the best single algorithm for a given problem, multiple classification systems combine a number of base algorithms to provide (in some cases) more accurate and robust solutions [6]. A second stage combiner is used to assimilate and process the outputs of the base classifier. Two types of combination strategies are typically considered - fixed and trainable combiners. While the fixed combiners operate directly on the outputs of the base classifiers, the trainable ones use the outputs of base classifier as a new feature representation.

Fixed combining rules assume that the responses of the base classifiers are comparable. This assumption does not necessarily hold if different classifier architectures are combined, such as Fisher Linear Discriminant (FLD), Support vector machines or neural networks. The trainable combiners are capable of overcoming this problem by learning the pattern in the outcomes of base classifiers.

Although benefits of trainable combiners have been demonstrated in number of studies [4, 2, 11, 5, 1, 9], it is still unclear which training strategy results in the best performance to suit a particular situation. Thus far, several strategies have been utilized. The first uses the same training set for both training of the base classifiers and (after processing by the trained base classifiers) also for the training of the combiner. Using the same set of examples for both stages, however, inevitably leads to a biased combiner. In order to remedy this situation, Raudys proposed to estimate the bias of the base classifier and correct for it [8]. This solution is, however, applicable only in the special case of linear base classifiers. A practical general recommendation, given by Duin in [2], is to train the base classifiers in a weak fashion. The combiner may then still have the flexibility to correct for their weakness as the bias is almost avoided.

Another training strategy avoids the biased dataset for training of the combiner by the use of a validation set. The available training data is split into two independent subsets. The first is utilized for training of the base classifiers. The second set, processed by the trained base classifiers, serves to train the combiner. The shortcoming of this approach is that the data available for training of each of the stages is typically severely limited, and thus leading to poorer overall performance. A compromise solution may be the construction of partially overlapping training and validation sets [1]. This observation might suggest that trainable combining rules are applicable only for large datasets [11, 9].

In this paper, we focus on an alternative approach which has a potential to improve the applicability of trainable combiners to smaller sample size problems, called *stacked generalization*, first introduced by Wolpert [13]. This is a general technique for construction of multi-level learning systems. In the context of classifier combination, it yields unbiased, full-size training sets for the trainable combiner in the following way:

- For each base classifier, an internal rotation-based cross-validation is performed such that all the fold test sets constitute the full original dataset.
- A classifier, trained in one of the internal folds, is applied to the respective fold test set and its outputs (such as class posteriori probability estimates or confidences) are stored.
- By collating the outputs of all the classifiers produced by the internal cross-validation, a full-size dataset consisting of the classifier outputs is constructed to be used for the second-stage training. Since each of these training examples was processed by the base classifier trained on an independent data set, the base classifier outputs in the resulting dataset are unbiased.

Stacked generalization therefore alleviates both of the aforementioned problems by providing the unbiased training set for the combiner, without sacrificing any of the available training examples.

Stacked generalization was discussed in a classifier combining context by Ting and Witten [12]. In their study, it outperformed both the model selection based on cross-validation, and the majority voting combiner. From a regression viewpoint, LeBlanc and Tibshirani [7] investigated the stacked generalizing combiner in a small artificial example using a linear classifier and the nearest neighbor rule. The existing studies on trainable combining using the stacked generalization focus on the selection of base classifiers and their outputs or viable combiner models. To our knowledge, the analysis of the relation between the performance of stacked combiners and training sizes has not been performed.

In this paper we investigate the behaviour of trainable combiners based on stacked generalization in comparison to two other primary approaches, namely re-using of the full training set (denoted the *re-use method*) by both stages, and the strategy based on the validation set (the *validation method*). We compare the behaviour of the three strategies across varying training set sizes in an attempt to understand their strengths and weaknesses.

In Section 2, the derivation of the combiner training set for the different approaches is formally introduced. We also discuss derivation of the base classifiers used in the stacked system and propose an alternative method increasing their robustness. In Section 3, we describe a set of experiments on several datasets and discuss our main findings. The final conclusions are given in Section 4.

2 Derivation of Training Set for the Trainable Combiner

We assume a training dataset X, with N examples $x_i \in R^D$, $i = 1, ..., N$ each assigned into one of C classes and a set of B untrained base classifiers A_b, $b = 1, ..., B$. The base classifiers are trained according to the fusion strategy used. Note, that if independent feature representations are available, the base classifier A_b will be trained on the corresponding dataset X_b. In the following, we denote such a trained classifier by $\hat{A}_b(X_b)$. By reapplying the trained classifier $\hat{A}_b(X_b)$ to the input set X_b, a dataset Y_b with C classifier outputs is created. The procedure, repeated for each of the base classifiers, yields a set Y, with N examples and BC features. The combiner A_{comb} is trained on this second-stage dataset Y. This procedure coincides with the *re-use method* for both the base classifiers and the combiner.

A new incoming observation $z \in R^D$ is assigned into one of the C classes using the trained combiner $\hat{A}_{\text{comb}}(Y)$ in the following way: First, the observation is subjected to each of B trained base classifiers $\hat{A}_b(X_b)$ using appropriate feature representations. The resulting outputs of the base classifiers are concatenated to form a feature vector in BC-dimensional space, and then finally the trainable combiner $\hat{A}_{\text{comb}}(Y)$ is applied.

The second approach (the *validation method*), investigated in this paper uses an independent validation set, in order to reduce the bias of the trainable combiner. The input dataset X is split into the training part Tr, and the mutually exclusive validation set V. The output dataset Y is composed of outputs of the trained classifiers $\hat{A}_b(Tr_b)$, obtained on the respective validation sets V_b, $b = 1, ..., B$. Note that both, base classifiers and and combiner, are trained on subsets of the original input datasets.

The application of *the stacked generalization* technique is illustrated in Figure 1, focusing on a single base classifier A_b. The dataset X_b is split into F

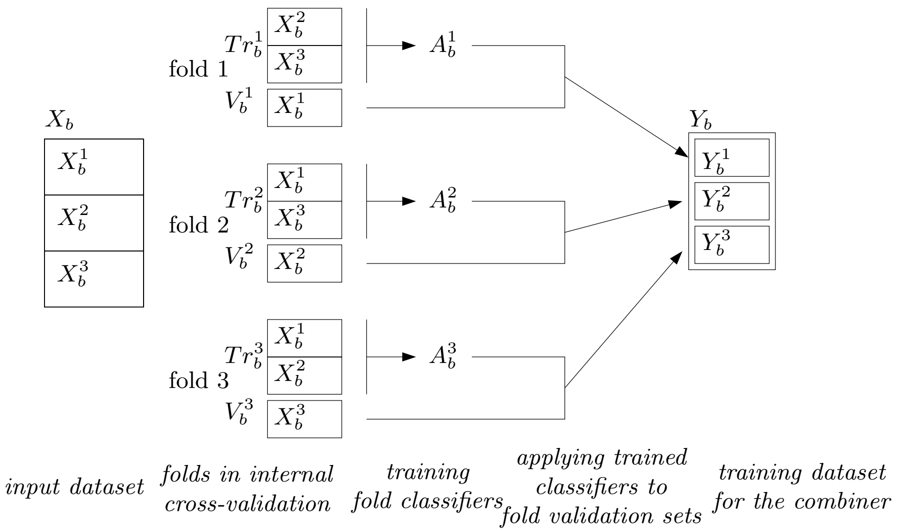


Fig. 1. Construction of a training set for the trainable combiner using stacked generalization, shown for the base classifier A_b

mutually exclusive parts X_b^f, $f = 1, ..., F$ of almost equal sizes. In an internal F-fold cross-validation procedure, trained classifiers $\hat{A}_b^f(Tr_b)$ are constructed using the per-fold training subsets $Tr_b^f = \cup_{j=1,j\neq f}^F X_b^j$. Each of the F trained fold classifiers are applied to the independent validation subset $V_b^f \equiv X_b^f$, yielding an output set Y_b^f. The full output set, specific to b-th base classifier is then constructed by concatenation $Y_b = \cup_{f=1}^F Y_b^f$. The process is concluded by re-training the base classifier on the full set X_b, i.e. by producing $\hat{A}_b(X_b)$ [12, 1].

Because the internal cross-validation derived F versions $\hat{A}_b^f(Tr_b^f), f = 1, ..., F$ for each base classifier A_b, we propose to use their fixed combination $\hat{A}_b^*(Tr_b)$ as the final base classifier, instead of a single re-trained one. Our motivation is to leverage the slight variations in the existing per-fold classifiers for the sake of increasing the robustness of the whole classification system. Because we assume a similar distribution of noise in the cross-validated training subsets, we propose to use the mean combiner [2].

3 Experiments

3.1 Experimental Setup

In this section the results of a number of experiments are shown, comparing the different approaches to deriving the training dataset for trainable combiners. The experiments were performed on the following datasets:

Handwritten Digits. The handwritten digits dataset (UCI repository[1]) contains 2000 objects from ten digit classes (200 objects per class). The dataset is composed of four different feature representations using the Fourier descriptors (76 features), Karhunen-Loeve coefficients (64 features), Zernike moments (47 features) and raw pixel values (240 features).

Spectra. A set of 988 measurements of fluorescent spectra in oral cavity labeled into two classes (856 healthy and 132 diseased tissue examples) [10]. Each measurement consists of six independent spectra measured in different locations in the oral cavity. Each spectrum is represented by of 199 wavelengths.

Waveform. An artificial dataset with three classes of waves each derived as a combination of two of three "base" waves (UCI repository). The dataset has in total 5000 examples and 21 features.

Sonar. A two-class real-world dataset with 208 data samples (111 examples of metal, and 97 examples of rock) and 60 continuous features (UCI repository).

The handwritten digits and spectra datasets contain different feature representations for each measurement. Therefore, we use a single type of a base classifier which is applied in different feature spaces. The waveform and sonar datasets are examples of problems with a single feature representation. Hence, we combine different base models in a single feature space.

In order to understand the behaviour of combiners utilizing different strategies for the construction of the second-stage training set, we estimate the learning curves in the following way. A training set of a desired size and an independent test set are drawn from the input dataset. On the training set, the base classifiers are trained and the second-stage dataset is constructed by the procedures described in Section 2. This dataset is then used for training of the combiner. The trained combiner is executed on the test set, and the mean classification error over the test examples is estimated. Note that all steps required for building the combiner, including the internal cross-validation of the stacked generalization, are performed on the training set only. For a given training set size, this procedure is repeated 10 times and the results are averaged. For the handwritten digits, spectra and waveform datasets the training set sizes vary from 5 to 100

Table 1. Experimental configurations

dataset	*base classifiers*	*test set size (per class)*
handwritten digits	FLD	50
spectra	FLD	30
waveform	FLD,Parzen,1-NN	500
sonar	FLD,Parzen,1-NN,NMC	40

[1] http://www.ics.uci.edu/˜mlearn/MLRepository.html

examples per class, for the sonar dataset from 5 to 50 examples per class. The base algorithms and test set sizes used are summarized in Table 1.

The outputs of the Parzen base classifier consists of posterior probability estimates. The outputs of the non-density based classifiers (distances to a decision boundary or a distance to the closest prototype) were normalized to a $< 0; 1 >$ range by a sigmoid function [3]. We have considered three different types of trainable combiners, ranging from simple to complex models, namely the decision templates (DT) [5] (effectively the nearest mean classifier), the Fisher linear discriminant (FLD), and the 1st nearest neighbor rule (1-NN). Note that while the FLD internally scales its inputs, the DT and 1-NN combiners require the proper scaling as explained above.

The experiments compare the three combiner training strategies, namely the *re-use method*, *validation method* (50/50% split), and *stacked generalization* using 10-fold internal cross-validation. For stacked generalization, we distinguish two approaches for construction of the trained base classifiers, as described in Section 2. We refer to the case when base classifiers are re-trained on the complete training dataset as "method I" [12, 1]. The second method, denoted "method II", uses a mean combiner over the set of 10 classifiers, trained during stacked generalization process.

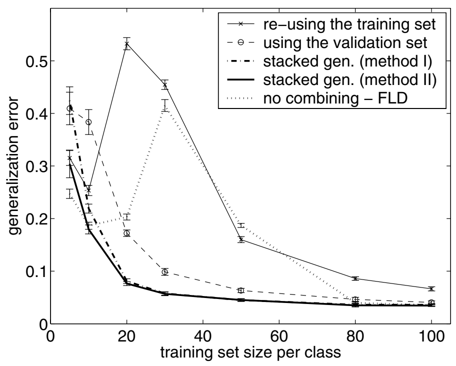
3.2 Results and Discussion

The handwritten digit results are presented in Figure 2, subfigures (a),(c), and (e). This subfigures (b), (d), and (f) correspond to the spectral dataset. The rows of the Figure 2 refer to FLD, DT, and 1-NN training combiners, respectively.

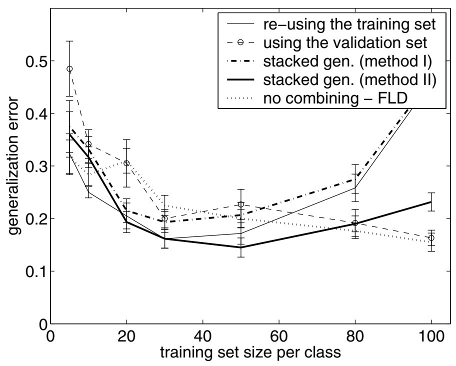
In the subfigure 2 (a), we can observe that the combiner re-using the training set (thin solid line) exhibits a drastic error increase between 10 and 50 examples per class. This is caused by superimposing the error peaks of the base FLD classifiers occurring when the sample size matches the feature space dimensionality. The combiner using the validation set (thin dashed line, circular markers) delivers significantly better results. The stacked generalizer using method I (thick dash-dotted line) significantly improves over these two traditional techniques for more than 10 examples per class. For the smallest training sample size, it is surprisingly outperformed by the re-use method. The proposed stacked generalizer using method II is the best of all remaining combiners even for the smallest training sample size. The dotted line represents a learning curve of a single FLD classifier directly applied to the full dataset. It illustrates that combining of different feature representation is beneficial. Employing the Parzen base classifier instead of FLD lead to analogous results (experiments not shown here).

For large sample sizes, the methods become comparable. This is understandable because a large training set diminishes the effects of bias when the training dataset is re-used and, at the same time, is sufficiently large for splitting into training and validation subsets.

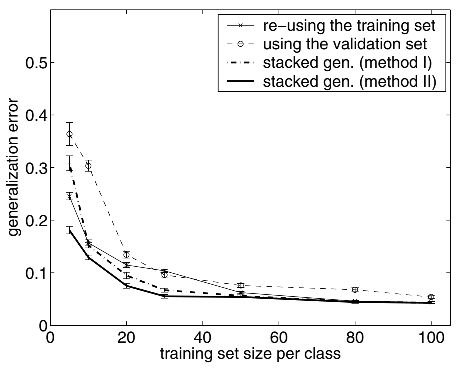
The results obtained on the spectral dataset depict the significant performance deterioration occurring for larger sample sizes. It is the result of a peaking effect where for 200 training examples a linear classifier is built in a 199 dimen-



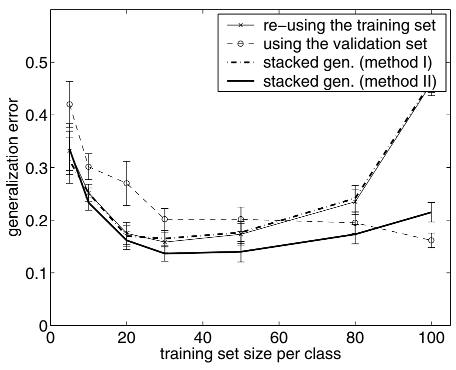
(a) digits, combiner: FLD



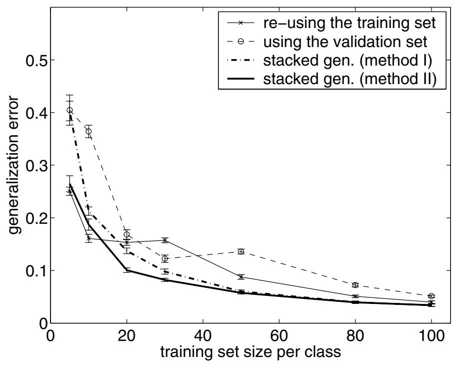
(b) spectra, combiner: FLD



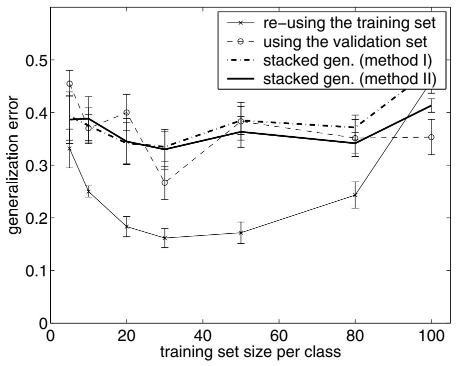
(c) digits, combiner: DT



(d) spectra, combiner: DT

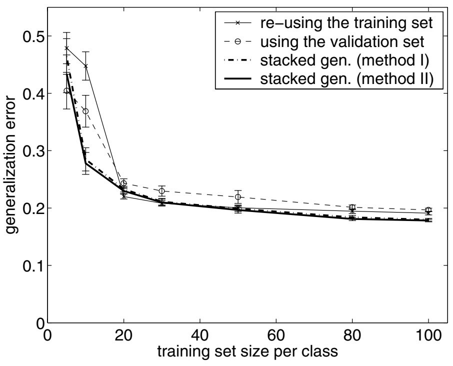


(e) digits, combiner: 1-NN

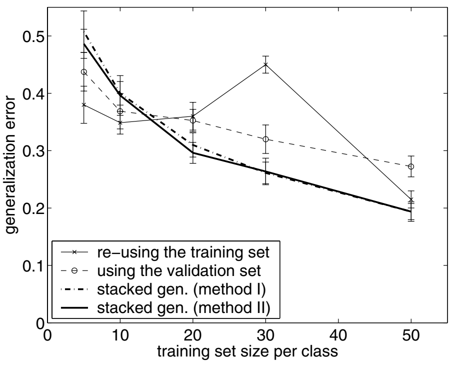


(f) spectra, combiner: 1-NN

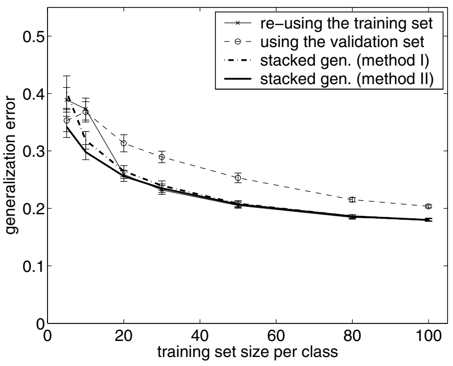
Fig. 2. Handwritten digit datasets (left column) and spectral dataset (right column). In all cases the FLD was used a base classifier



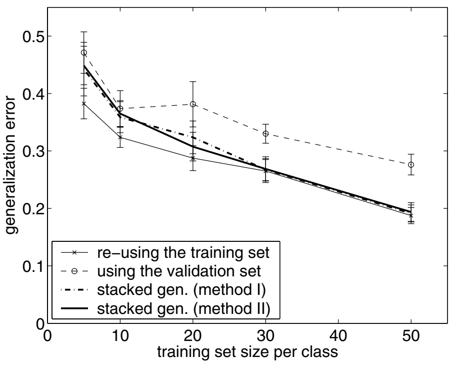
(a) waveform, combiner: FLD



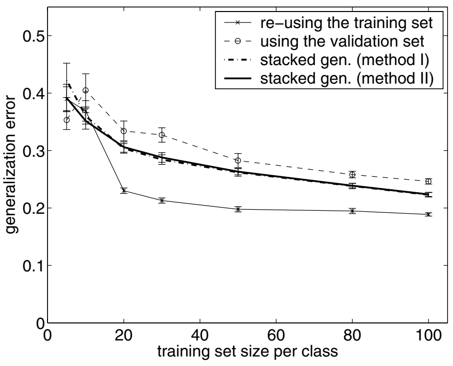
(b) sonar, combiner: FLD



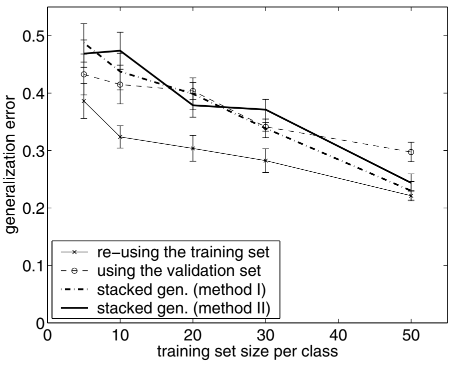
(c) waveform, combiner: DT



(d) sonar, combiner: DT



(e) waveform, combiner: 1-NN



(f) sonar, combiner: 1-NN

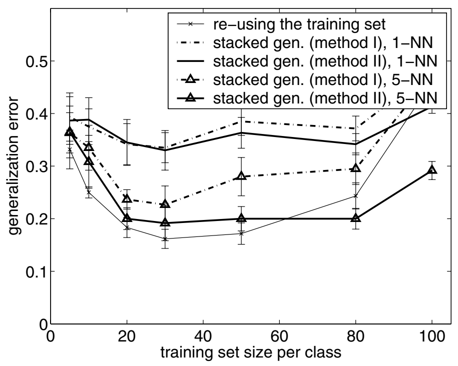
Fig. 3. Experiments with waveform and sonar datasets. The base classifiers are indicated in Table 1

sional feature space. Both the combiner re-using the training set and stacked generalizer with method I suffer from the peaking. The stacked generalizer using the method II benefits from a more robust base classifiers. Even better results are reached by the validation method. Its base classifiers are trained in a space with higher dimensionality then the number of training examples, using the pseudo-inverse. The dotted line denoting the single FLD classifier applied to the full dataset. It again illustrates that combining is beneficial for smaller sample sizes.

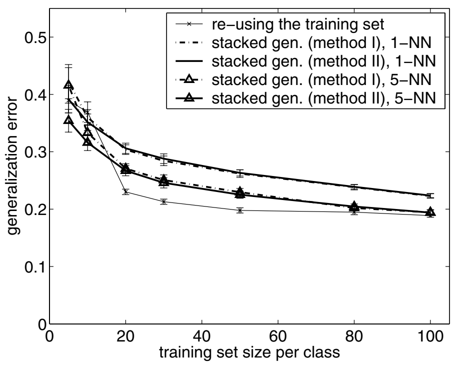
Similar trends may be also observed in Figure 3 depicting the experiments with Waveform and Sonar datasets and in additional experiments with other base classifiers such as nearest mean or Parzen we omit for the sake of brevity.

While FLD and DT combiners exhibit similar learning curve trends, the 1-NN combiner yields very different results. The most profound difference is apparent in subfigures 2 (f), 3 (e) and 3 (f), where the re-use method significantly outperforms all other strategies. Because the base classifiers are identical to those in experiments where FLD and DT combiners were used, we conclude that the inferior results are related to the 1st nearest neighbor combiner. This behaviour is understandable for the combiner using the validation set because already small amount of training examples is still cut in half. However, this cannot explain the failure of the stacked generalizers employing the second-stage datasets of identical size to the well-performing re-using approach.

We hypothesize that it is the presence of noise in the second-stage training set generated by the stacked generalization and the subsequent failure of the noise-sensitive 1-NN rule. We have performed an additional experiment with a combiner based on the 5-th nearest neighbor rule. The results are presented in Figure 4. The less noise-sensitive 5-NN combiner (triangular markers) results in a significant improvement eventually reaching the performance comparable with the 1-NN combiner trained on a re-used dataset. We conclude that the stacked generalization introduces additional level of noise in the second-stage dataset. It should be, therefore, used together with simple and robust combiners that are capable of averaging out this noise.



(a) spectra, base: FLD



(b) waveform

Fig. 4. The effect of less noise sensitive 5-NN combiners on the stacked generalizers

4 Conclusions

In this paper, we have compared three principal methodologies for the construction of the second-stage training sets for trainable combiners. Apart of the commonly employed re-using of the available training set by both stages and the validation set, we also investigate the stacked generalization technique which yields full-size unbiased second-stage training sets. However, the stacked generalization, inevitably introduces additional noise into the second-stage datasets.

Our experiments demonstrate that trainable combiners, derived using the stacked generalization exhibit significant performance improvements over the other two currently used methods for moderate training set sizes. For very small sample sizes, the best strategy is the re-use of a complete training set for both stages. Any split strategy sacrifices the scarce training data and should be, therefore, avoided. For a large number of training samples, the studied approaches to derivation of the second-stage training set do not differ and the re-using strategy may be again recommended as the simplest solution.

For the moderate training set sizes, the stacked generalization appears to significantly outperform the other two approaches. Commonly used combiners based on stacked generalization train the final base classifiers on the full training set. We have proposed to construct the base classifier from the available fold classifiers by a fixed mean combiner. This solution appears to bring additional robustness and should be preferred in applications where evaluation of all the fold classifiers for each new example does not pose a excessive speed burden.

We have noticed the performance drop of the stacked generalizers using the 1-NN combiner. We have shown that the stacked generalization introduces additional noise into the second-stage training dataset and therefore requires simple and robust combiners such as FLD or DT for a good performance. We conclude our study noting that the stack generalization provides a combiner training strategy superior to the validation set approach and for larger than very small sample sizes systematically outperforms the re-using approach.

Acknowledgements. This research is/was supported by the Technology Foundation STW, applied science division of NWO and the technology program of the Ministry of Economic Affairs.

References

1. C. Dietrich, G. Palm, and F. Schwenker. Decision templates for the classification of bioacustic time series. *Information Fusion*, 4:101–109, 2003.
2. R.P.W. Duin. The combining classifiers: to train or not to train? In *Proc. of 16th Int.Conf. on Pattern Recognition (Quebec City), vol. II*, pages 765–770, 2002.
3. R.P.W. Duin, P. Juszczak, D. de Ridder, P. Paclík, E. Pekalska, and D. M. J. Tax. PR-Tools 4.0, a Matlab toolbox for pattern recognition. Technical report, ICT Group, TU Delft, The Netherlands, January 2004. `http://www.prtools.org`.

4. R.P.W. Duin and D.M.J. Tax. Experiments with classifier combining rules (invited paper). In *Proc. of Multiple Classifier Systems MCS 2000*, volume LNCS, vol.1857, pages 16–29, 2000.
5. L.I. Kuncheva, J.C. Bezdek, and R.P.W. Duin. Decision templates for multiple classifier fusion: an experimental comparison. *Pattern Recog.*, 34(2):299–314, 2001.
6. Ludmila I. Kuncheva. *Combining Pattern Classifiers.* Wiley & Sons, 2004.
7. Michael LeBlanc and Robert Tibshirani. Combining estimates in regression and classification. *Journal of the American Statistical Association*, 91(436), 1996.
8. S. Raudys and A. Janeliunas. Reduction a boasting bias of linear experts. In *Proc. of Multiple Classifier Systems MCS 2002*, volume LNCS vol.2364, 2002.
9. F. Roli, S. Raudys, and G. L. Marcialis. An experimental comparison of fixed and trained fusion rules for crisp classifier outputs. In *Proc. of Multiple Classifier Systems MCS 2002*, volume LNCS vol.2364, pages 232–241, 2002.
10. M. Skurichina, P. Paclik, R.P.W. Duin, D.C.G. de Veld, H.J.C.M. Sterenborg, M.J.H. Witjes, and J.L.N. Roodenburg. Selection/extraction of spectral regions for autofluorescence spectra measured in the oral cavity. In *Proc.of SSSPR 2004*, volume vol. 3138 LNCS, Springer, Berlin, pages 1096–1104, 2004.
11. Ching Y. Suen and Louisa Lam. Multiple classifier combination methodologies for different output levels. In *Proc. of Multiple Classifier Systems MCS 2000*, volume LNCS, vol. 1857, Springer, Berlin, 2000.
12. Kai Ming Ting and Ian H. Witten. Issues in stacked generalization. *Journal of Artificial Intelligence Research*, 10:271–289, 1999.
13. D.H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.