Competent Undemocratic Committees.

Włodzisław Duch, Łukasz Itert, and Karol Grudziński

Department of Informatics, Nicholas Copernicus University,
Grudziądzka 5, 87-100 Toruń, Poland. http://www.phys.uni.torun.pl/kmk

Abstract. Committees of models are frequently employed to improve accuracy and decrease the variance of individual models. Each model has an equal right to vote (democratic procedure), despite obvious differences in model competence in different regions of the feature space. Adding competence factors to different models before calculation of the committee decision (undemocratic procedure) improves the quality of the committee. A method for creation of a committee of competent models is described and empirical tests presented.

1 Introduction

Combining information from different classifiers is an important and quite popular subject in machine learning. Whole conferences and special issues of journals are devoted to this subject (see references in [1]), know as ensemble learning, mixture of experts, voting classification algorithms, or committees of models [2]. Ortega, Koppel and Argamon [1] point out that there are problems, such as predicting the glucose levels of diabetic patients, that a large number of different learning algorithms have been applied.

Mixture of models may not only improve the accuracy of a single model, but also may decrease the variance, stabilizing and improving generalization of the whole system [3]. Two sources contribute to the variability of models created for use in a committee: different samples taken from the same data, for example in crossvalidation training or in such methods as boosting, bagging or arcing [2,3] and different bias of models due to the change of their complexity, such as the number of neurons or other parameters. Recently we have developed a framework for similarity based methods (SBM) [4] and used some methods that belong to this framework to create voting committees [5], obtaining significant improvements of results and decrease of variance of committee errors.

Typical voting techniques follow the democratic majority decision, linear combination or selecting the most confident models. In the mixture of experts neural architecture Jacobs [6] has introduced a gating network to select the most competent model. Very recently Ortega et al [1] used similar idea, a “referee meta-model” deciding which model should contribute to the final decision. These undemocratic procedures exploit the fact that different models may have different areas of competence. The idea of competent voting was also mentioned in [7] but has not been developed further. Global selection of competent models has recently been introduced [8]. Instead of training a meta-model each area of the input space in which a given model makes a number of errors is identified and a penalty factor is used to decrease the influence of this model during the voting.

In the next section methods for model combination are briefly discussed and algorithms for creating committees of competent models are described. In the third section results of a numerical experiment are presented. Finally some conclusions and plans for further work are given.

2 Combining models.

Individual models are frequently unstable [3], i.e. quite different models are created as a result of repeated training (if learning algorithms are stochastic) or if the training set is slightly perturbed [9]. The mixture of models allows to approximate complicated probability distributions quite accurately. With $l = 1..m$ models providing estimation of probabilities $P(C_i|\mathbf{X};M_l)$ for $i = 1..K$ classes, one can use the majority voting, average results of all models, select one model that has highest confidence (i.e. gives the largest probability), set a threshold to select a subset of models with highest confidence and use majority voting for these models.

An empirical comparison of voting algorithms, including bagging and boosting, has been published by Bauer and Kohavi [10]. Tests were made using decision trees and naive Bayes method. The bagging algorithm uses classifiers trained on bootstrap samples, created by randomly drawing a fixed number of training data vectors from the pool which always contains all training vectors (i.e. drawing does not remove them from the pool). Results are aggregated by voting. AdaBoost (Adaptive Boosting) creates a sequence of training sets and determines weights of the training instances, with higher weights for those that are incorrectly classified. The arcing method uses a simplified procedure for weighting of the training vectors. Bauer and Kohavi [10] provided an interesting decomposition of bias and variance components of errors for these algorithms. Renormalized product of different predictors has been advocated recently by Hinton [11] in context of unsupervised probability density estimation.

A linear meta-model

$$p(C_i|\mathbf{X};M) = \sum_{l=1}^{m} W_{i,l} P(C_i|\mathbf{X};M_l) \tag{1}$$

provides additional mK linear parameters for model combination, determined using the standard Least Mean Squares (LMS) procedure.

3 Committees of Competent Models (CCM)

So far all models selected to the ensemble were allowed to vote on the final result. Krogh and Vedelsby [12] showed that ensemble generalization error is small if highly accurate classifiers disagreeing with each other are used. Contrary to this idea Xin Yao has used averaging of results with negative correlation between individual models to diversify their pool [13]. Each model does not need to be accurate for all data, but should account well for a different (overlapping) subset of data.

The Similarity Based Models [4] use reference vectors (selected from a training set) and it is relatively easy to determine the areas of the input space where a given model is competent (makes a few errors) and where it fails. Vectors that cannot be correctly classified show up as errors that all model make, but some vectors that are erroneously classified by one model may be correctly handled by another. This information may be used in several ways. A simple algorithm that includes information on the competence of different models is presented below.

1. Optimize parameters for all models $M_l, l = 1 \ldots m$ on the training set using a cross-validation procedure.
2. For each model $l = 1 \ldots m$
   (a) for all training vectors $\mathbf{R}_i$ generate predicted classes $C_l(\mathbf{R}_i)$;
   (b) if $C_l(\mathbf{R}_i) \neq C(\mathbf{R}_i)$, i.e. model M_l makes an error for vector R_i, determine the area of incompetence of the model, finding the distance $d_{i,j}$ to the nearest vector that M_l has correctly classified;
   (c) set parameters of the incompetence factor $F(||\mathbf{X}-\mathbf{R}_i||;M_l)$ in such a way that its value decreases significantly for $||\mathbf{X}-\mathbf{R}_i|| \geq d_{i,j}/2$.
3. The incompetence function for the model $F(\mathbf{X};M_l)$ is a product of factors $F(||\mathbf{X}-\mathbf{R}_i||;M_l)$ for all training vectors that have been incorrectly handled.

The incompetence function $F(\mathbf{X};M_l) \approx 1$ in all areas where the model has worked well and $F(\mathbf{X};M_l) \approx 0$ near the training vectors where errors were made. A number of functions may be used for that purpose: a Gaussian function $F(||\mathbf{X}-\mathbf{R}_i||;M_l) = 1-G(||\mathbf{X}-\mathbf{R}_i||^a;\sigma_i)$, where $a \geq 1$ coefficient is used to flatten the function, a simpler $F(||\mathbf{X}-\mathbf{R}_i||;M_l) = 1/(1+||\mathbf{X}-\mathbf{R}_i||^{-a})$ function or a sum of two logistic functions $\sigma(-||\mathbf{X}-\mathbf{R}_i|| - d_{i,j}/2) + \sigma(||\mathbf{X}-\mathbf{R}_i|| - d_{i,j}/2)$. Since a number of factors enters the incompetence function of the model each factor should quickly reach 1 outside the incompetence area. This is achieved either by using large a values, high slopes of sigmoids or defining a cut-off values where a value 1 is taken.

Such committee of competent models may be used in several ways. In the voting phase nearest neighbor reference vectors should be determined and only those classifiers that are competent should be included in the voting procedure. If no competent models are found the vector given for classification is probably an outlier and should be left as 'rejected' or 'impossible to classify'. Sometimes it helps if all such vectors are removed from the training set, but this is achieved automatically by competent classifiers.

Even simpler way of creating competent committee is introduced if linear combinations are used instead of majority voting. For class C_i coefficients of linear combination are determined from the least-mean square solution of:

$$p(C_i|\mathbf{X};M) = \sum_{l=1}^{m} \sum_{m} W_{i,l} F(\mathbf{X};M_l) P(C_i|\mathbf{X};M_l) \tag{2}$$

The incompetence factors simply modify probabilities $F(\mathbf{X};M_l)P(C_i|\mathbf{X};M_l)$ that are used to set linear equations for all training vectors $\mathbf{X}$, therefore the solution is done in the same way as before. After renormalization $p(C_i|\mathbf{X};M)/\sum_j P(C_j|\mathbf{X};M)$ give

final probability of classification. In contrast to AdaBoost and similar procedures [2] explicit information about competence, or quality of classifier performance in different feature space areas, is used here.

4 Numerical experiments

Numerical experiments were made on Telugu vowel data [14] containing intensities of 3 formants, for 6 vowels. The classes overlap strongly and 871 samples are given in the dataset. The results of different tests are collected in Table 1. Other methods discussed in ref. [14] gave worse results.

Table 1. Comparison of results on Telugu vowel data. 2xCV means 2-fold stratified crossvalidation test.

System	Accuracy	Remarks
CUC committee	88.2± 0.6%	2xCV (our calculation)
kNN	86.1± 0.6%	k=3, Euclidean, 2xCV (our calculation)
MLP	84.6 %	2xCV, 10 neurons [14]
Fuzzy MLP	84.2 %	2xCV, 10 neurons [14]
Bayes Classifier	79.2 %	2xCV, [14]
Fuzzy Kohonen	73.5 %	2xCV, [14]

Since our calculations with the nearest neighbor classifier gave quite promising results we have divided the dataset randomly into two parts and trained a committee of kNN models on each part, treating the other part as test data. The committee included the following models: M_1 with k=10, Euclidean, M_2 with k=13, Manhatan, M_3 with k=5, Euclidean and M_4 with k=5 and Manhatan.

Table 2. Accuracy of 4 models for each class, in %.

Class	M_1	M_2	M_3	M_4
C_1	50.0	45.8	65.3	62.5
C_2	88.8	91.0	87.6	89.9
C_3	84.3	84.3	84.9	84.7
C_4	85.4	84.8	90.1	88.1
C_5	91.3	88.4	90.3	90.1
C_6	90.6	92.8	90.1	90.4
Average	85.1	84.6	86.1	86.0

Accuracy of each model is given in Table 2. Although the overall accuracy may be similar these models significantly differ in the accuracy for different classes. If

one could select the best model for a given class, for example model 3 for class 1, accuracy would grow to 87.9%, but of course the class cannot be selected, it should be predicted. 86 vectors were assigned to their classes incorrectly by all 4 models, giving a chance to account correctly for the remaining 785 vectors, or 90.1 % of all vectors.

The Table 3 contains results obtained from 4 types of committees, created by majority voting, selecting the model with highest confidence, linear combination (1) and a linear combination with competence factors (2). Results obtained with committees are usually better than results of a single model, with majority voting is the worst, the highest confidence and linear combination on the same level (slightly better) and a significant improvement for the linear combination of competent models.

Table 3. Results from committees created in 4 ways: by majority voting, highest confidence, linear combination and a linear combination with competence factors.

Class	Majority	Confidence	Combination	+ Competence
C_1	54.2	58.3	62.5	65.3
C_2	88.8	88.8	88.8	89.9
C_3	84.3	84.9	84.3	84.9
C_4	86.8	88.1	88.1	88.1
C_5	92.3	92.8	92.3	93.8
C_6	90.6	92.2	91.7	93.3
Average	85.9	87.0	87.0	88.2

5 Conclusions

Although more empirical tests are needed, assigning incompetence factors in various voting procedures, including linear combination of models, is an attractive idea that may significantly improve analysis of difficult problems. Since there is no need to create a single model that handles all data correctly learning may become modular, with each model specializing in different subproblems. A constructive approach to committee growth may be used: after creating initial committee by combining competent models created so far new models should be searched that classify correctly just those vectors, that the committee has still problems with.

Ideas presented here may be developed in a number of directions. So far we have tried to aggregate only a few models generated with different parameters. The same procedure may be applied to models generated using adaptive boosting or similar algorithms [2]. An interesting possibility is to train a neural network, providing input vectors and predicting competent models. A combination of classifiers gives Receiver Operator Characteristic (ROC) curves that cover a convex combination of all individual ROC curves, allowing to reach better operating points, i.e. detection

rates for a given false alarm rate [17]. Models that end up with small effective coefficients for all training data may be pruned. Diversification of models by adding explicit negative correlation is also worth considering [13]. A lot of other options remains to be investigated.

Acknowledgments: Support by the Polish Committee for Scientific Research, grant 8 T11C 006 19, is gratefully acknowledged.

References

1. Ortega J, Koppel M, Argamon S. (2001) Arbitrating Among Competing Classifiers Using Learned Referees. Knowledge and Information Systems **3**, 470-490
2. Bauer E, Kohavi R. (1999) An empirical comparison of voting classification algorithms: bagging, boosting and variants. *Machine learning* **36**, 105-142
3. Breiman, L. (1998): Bias-Variance, regularization, instability and stabilization. In: Bishop, C. (Ed.) Neural Networks and Machine Learning. Springer, Berlin, Heidelberg, New York
4. Duch W. (2000) Similarity based methods: a general framework for classification, approximation and association, Control and Cybernetics **29**, 937-968
5. Duch W, Grudziński K. (2001) Ensembles of Similarity-Based Models. Inteligent Information Systems 2001, Advances in Soft Computing, Physica Verlag (Springer), pp. 75-85
6. Jacobs R. A. (1997) Bias/Variance Analyses of Mixtures-of-Experts Architectures. Neural Computation **9**, 369-383
7. Duch W, Adamczak R, Diercksen G.H.F. (2000) Classification, Association and Pattern Completion using Neural Similarity Based Methods. Applied Mathematics and Computer Science **10**, 101-120
8. Giacinto G, Roli F. Dynamic Classifier Selection Based on Multiple Classifier Behaviour. Pattern Recognition, 34 (2001) 179-181
9. Avnimelech R, Intrator N. (1999) Boosted Mixture of Experts: An Ensemble Learning Scheme. Neural Computation **11**, 483-497
10. Bauer E, Kohavi R. (1999) An empirical comparison of voting classification algorithms: Bagging, Boosting and variants. Machine Learning 36, 105-139
11. Hinton, G. (2000): *Training products of experts by minimizing contrastive divergence.* Gatsby Computational Neuroscience Unit Technical Report 2000-004
12. Krogh A, Vedelsby J. (1995) Neural Network Ensembles, Cross Validation, and Active Learning. Advances in Neural Information Processing Systems, MIT Press, **7**, 231–238.
13. Yao, X., Liu, Y. (1997): A New Evolutionary System for Evolving Artificial Neural Networks. IEEE Transaction on Neural Networks 8, 694–713
14. Pal, S.K. and Mitra S. (1999) *Neuro-Fuzzy Pattern Recognition*. J. Wiley, New York
15. Duch W, Adamczak R, Grabczewski K, A new methodology of extraction, optimization and application of crisp and fuzzy logical rules. IEEE Transactions on Neural Networks 12 (2001) 277-306
16. D. Michie, D.J. Spiegelhalter and C.C. Taylor, "Machine learning, neural and statistical classification". Elis Horwood, London 1994
17. Swets J.A. (1988) Measuring the accuracy of diagnostic systems. Science **240**, 1285-93