Combining MF Networks: A Comparison Among Statistical Methods and Stacked Generalization

Joaquín Torres-Sospedra, Carlos Hernández-Espinosa, and Mercedes Fernández-Redondo

Departamento de Ingenieria y Ciencia de los Computadores, Universitat Jaume I, Avda. Sos Baynat s/n, C.P. 12071, Castellon, Spain
{jtorres, espinosa, redondo}@icc.uji.es

Abstract. The two key factors to design an ensemble of neural networks are how to train the individual networks and how to combine the different outputs to get a single output. In this paper we focus on the combination module. We have proposed two methods based on *Stacked Generalization* as the combination module of an ensemble of neural networks. In this paper we have performed a comparison among the two versions of *Stacked Generalization* and six statistical combination methods in order to get the best combination method. We have used the mean increase of performance and the mean percentage or error reduction for the comparison. The results show that the methods based on *Stacked Generalization* are better than classical combiners.

1 Introduction

The most important property of a neural network is its generalization capability. The ability to correctly respond to inputs which were not used in the training set.

It is clear from the bibliography that the use of an ensemble of neural networks (figure 1) increases the generalization capability, [1,2], for the case of *Multilayer Feedforward* (MF) and other classifiers. The two key factors to design an ensemble are how to train the individual networks and how to combine them.

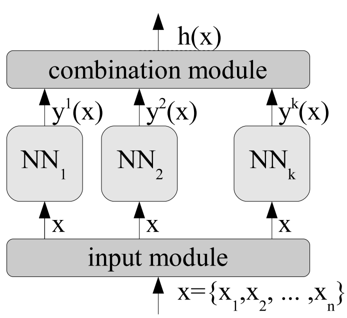

Fig. 1. The basic diagram of an Ensemble of Neural Networks

F. Schwenker and S. Marinai (Eds.): ANNPR 2006, LNAI 4087, pp. 210–220, 2006.

Among the methods of training the individual networks there are an important number of alternatives. Our research group has performed a comparison among methods of building ensembles which shows that the *Simple ensemble* method provides a reasonable performance with a lower computational cost [3,4].

Moreover, our research group has performed another comparison among combination methods of ensembles which shows that the *Output Average* is the simpler method but it is one of the best combination methods [5,5].

In this paper, we present some results of two versions of *Stacked Generalization* and we compare them with six *classic* combination methods. We have built ensembles of 3, 9, 20 and 40 networks with *Simple Ensemble* on six databases from the *UCI repository* to test the performance of the combination methods.

The methods are described in 2. The results we have obtained on these six databases are in subsection 3.2. We have also calculated general measurements of the combination methods to compare them, these results appear in subsetion 3.3.

2 Theory

In this section, firstly we briefly review the methods of combination that we have used in our experiments in subsections 2.1-2.6. Finally we describe two new methods based on *Stacked Generalization* in subsections 2.7 and 2.8.

2.1 Output Average

This approach simply averages the individual classifier outputs across the different classifiers.

$$\overline{y}_{class}(x) = \frac{1}{k} \cdot \sum_{net=1}^{k} y_{class}^{net}(x) \tag{1}$$

The output yielding the maximum of the averaged values is chosen as the correct class.

$$h_{average}(x) = \arg \max_{class=1,\ldots,q} \overline{y}_{class}(x) \tag{2}$$

Where q is the number of classes, k is the number of networks in the ensemble.

2.2 Majority Vote

Each classifier provides a vote to a class, given by the highest output. The correct class is the one most often voted by the classifiers.

$$vote_{class}^{net}(x) = \begin{matrix} 1 & \text{if } h^{net}(x) = d(x) \\ 0 & \text{otherwhise} \end{matrix} \tag{3}$$

$$h_{voting}(x) = \arg \max_{class=1,\ldots,q} \left(\sum_{net=1,\ldots,k} vote_{class}^{net}(x) \right) \tag{4}$$

2.3 Winner Takes All

In this method, the class with overall maximum output across all classifier and outputs is selected as the correct class.

$$\overline{y}_{class}(x) = \max_{net=1,\dots,k} y^{net}_{class}(x) \quad (5)$$

$$h_{wta}(x) = \arg \max_{class=1,\dots,q} \overline{y}_{class}(x) \quad (6)$$

2.4 Borda Count

For any class c, the Borda count is the sum of the number of classes ranked below c by each classifier. The Borda count for class $class$ is:

$$Borda_{class}(x) = \sum_{net=1}^{k} Borda^{net}_{class}(x) \quad (7)$$

Where $Borda^{net}_{class}(x)$ is the number of classes ranked below the class $class$ by the net-th classifier. The final hipothesys is given by the class yielding the highest Borda count.

$$h_{borda}(x) = \arg \max_{class=1,\dots,q} B_{class}(x) \quad (8)$$

2.5 Bayesian Combination

This combination method is based on the belief value, the class with maximum belief value is selected as the correct class. According to [6] this value is the conditional probability that the pattern x belongs to class i, it can be approximated by:

$$Belief_{class}(x) = \frac{\prod_{net=1}^{k} p(x \in class|h(y^{net}) = j)}{\sum_{i=1}^{q} \prod_{net=1}^{k} p(x \in i|h(y^{net}) = j)} \quad (9)$$

$$h_{bayesian}(x) = \arg \max_{class=1,\dots,q} Belief_{class}(x) \quad (10)$$

Where the conditional probability that sample x actually belongs to class i, given that classifier k assign it to class j can be estimated from the values of the confusion matrix [7].

$$p(x \in i|class(y^{net}) = j) = \frac{c^{net}_{i,j}}{\sum_{m=1}^{q} c^{net}_{m,j}} \quad (11)$$

2.6 Dinamically Averaged Networks

It is proposed in reference [8]. It is a weighted output average which introduces weights to the outputs of the different networks prior to averaging. The weights values are derived from the network output of the pattern we are classifying.

$$\overline{y}_{class}(x) = \sum_{net=1}^{k} w_{class}^{net} \cdot y_{class}^{net}(x) \tag{12}$$

Where the weights are calculated by:

$$w_{class}^{net}(x) = \frac{C_{class}^{net}(x)}{\sum_{i=1}^{k} C_{class}^{k}(x)} \tag{13}$$

$$C_{class}^{net}(x) = \begin{cases} y_{class}^{net}(x) & \text{if } y_{class}^{net}(x) \geq 0.5 \\ 1 - y_{class}^{net}(x) & \text{otherwise} \end{cases} \tag{14}$$

$$h_{dan}(x) = \arg \max_{class=1,\ldots,q} \overline{y}_{class}(x) \tag{15}$$

2.7 Stacked Generalization

Stacked Generalization was introduced by Wolpert [9]. The combination method we propose in this paper is based on the idea of *Stacked Generalization* and it consist on training a neural network to combine the output vectors provided by the networks of the ensemble. The neural network used for combination is called *Combination network*, the networks of the ensemble are also known as *expert networks*. In Figure 2 we can see a diagram of the *Stacked Generalization*.

2.8 Stacked Generalization Plus

The use of the original pattern input vector is the difference between *Stacked Generalization* and *Stacked Generalization Plus*. The outputs of the expert networks on patterns from training set and the original pattern input vector are used to train the combination network. In Figure 3 we can see a diagram of the *Stacked Generalization Plus*.

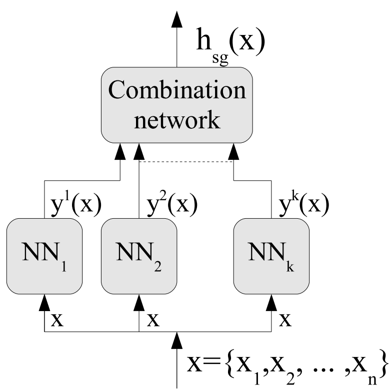

Fig. 2. Stacked Generalization diagram

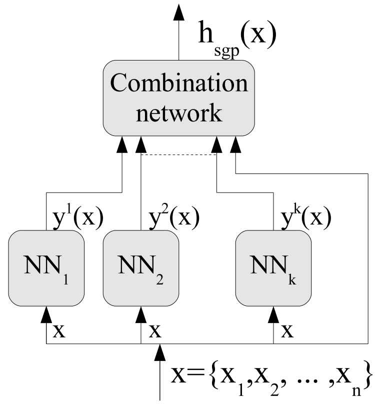

Fig. 3. Stacked Generalization Plus diagram

3 Experimental Testing

In this section we describe the experimental setup and the datasets we have used in our experiments. Finally, we show and compare the results we have obtained with the combination methods on the different datasets.

3.1 Datasets

We have used six different classification problems from the *UCI repository of machine learning databases* [10] to test the performance of methods. The databases we have used are:

Arrhythmia Database (aritm)
The aim is to distinguish between the presence and absence of cardiac arrhythmia and to classify it in one of the 16 groups. This dataset contains 443 instances, 277 attributes and 3 classes.

Glass Identification Database (glas)
The aim of the dataset is to determinate if the glass analysed was a type of 'float' glass or not for Forensic Science. This dataset contains 2311 instances, 34 attributes and 2 classes.

Ionosphere Database (ionos)
Classification of radar returns from the ionosphere. This dataset contains 351 instances, 34 attributes and 2 classes.

The Monk's Problems 1 (mok1)
Artificial problem with binary inputs. This datasets contain 432 instances, 6 attributes and 2 classes.

The Monk's Problems 2 (mok2)
Artificial problem with binary inputs. This datasets contain 432 instances, 6 attributes and 2 classes.

Vowel Database (vowel)
There is no description about it in the repository. This dataset contains 990 instances, 11 attributes and 11 classes.

Table 1 shows the training parameters (Step, Momentum, Number of Hidde Units and Number of iterations) we have used to train the combination networks for *Stacked Generalization*. Table 2 shows the training parameters for *Stacked Generalization Plus*. Finally, Table 3 shows the training parameters and the performance of expert networks.

All these values has been determinated by trial and error.

3.2 Results

In this subsection we present the experimental results. Table 4 shows the results we have obtained with ensembles of 3 networks. Tables 5, 6, 7 show the results we have obtained for ensembles of 9, 20 and 40 networks respectively.

3.3 Interpretations of Results

Comparing tables 4-7 we can see that both methods based on *Stacked Generalization* are more accurate than the classical methods.

Table 1. MF training parameters for Gating Network (Stacked)

Database	Networks	Hidden	Step	Momentum	Iterations
aritm	3	0.01	0.05	3	10000
	9	0.01	0.05	20	500
	20	0.01	0.05	1	100
	40	0.01	0.05	5	100
glas	3	0.01	0.05	3	10000
	9	0.01	0.05	3	10000
	20	0.01	0.05	5	10000
	40	0.01	0.05	5	10000
ionos	3	0.01	0.05	7	10000
	9	0.01	0.05	1	10000
	20	0.01	0.05	5	10000
	40	0.01	0.05	5	10000
mok1	3	0.01	0.05	1	10000
	9	0.01	0.05	1	10000
	20	0.01	0.05	1	10000
	40	0.01	0.05	1	10000
mok2	3	0.01	0.05	15	100
	9	0.01	0.05	5	100
	20	0.01	0.05	5	250
	40	0.01	0.05	25	250
vowel	3	0.01	0.05	19	10000
	9	0.01	0.05	6	7500
	20	0.01	0.05	20	500
	40	0.01	0.05	10	5000

We have calculated the increase of performance of *Stacked Generalization* and *Stacked Generalization Plus* with respect to *Output Average* to see more clearly if *Stacked* combination methods performs better. A positive value of the increase of performance means that the performance is better. A negative value means that the performance of the method on the dataset is worse. The results appear in tables 8 and 9.

Comparing the results showed in tables 8-9 we can see that the improvement in performance using our method depends on the database and the number of networks used

Table 2. MF training parameters for Gating Network (Stacked Plus)

Database	Networks	Hidden	Step	Momentum	Iterations
aritm	3	0.01	0.05	4	2500
	9	0.01	0.05	6	1500
	20	0.01	0.05	17	1500
	40	0.01	0.05	5	1500
glas	3	0.01	0.05	5	10000
	9	0.01	0.05	4	10000
	20	0.01	0.05	15	10000
	40	0.01	0.05	15	10000
ionos	3	0.01	0.05	1	10000
	9	0.01	0.05	1	10000
	20	0.01	0.05	4	10000
	40	0.01	0.05	5	10000
mok1	3	0.01	0.05	5	10000
	9	0.01	0.05	5	10000
	20	0.01	0.05	5	10000
	40	0.01	0.05	5	10000
mok2	3	0.01	0.05	4	2500
	9	0.01	0.05	5	250
	20	0.01	0.05	5	250
	40	0.01	0.05	1	250
vowel	3	0.01	0.05	30	2500
	9	0.01	0.05	13	5000
	20	0.01	0.05	10	2500
	40	0.01	0.05	7	5000

Table 3. MF training parameters for Expert Networks

Database	Hidden	Iterations	Step	Momentum	Performance
aritm	9	2500	0.1	0.05	75.6 ± 0.7
glas	3	4000	0.1	0.05	78.5 ± 0.9
ionos	8	5000	0.1	0.05	87.9 ± 0.7
mok1	6	3000	0.1	0.05	74.3 ± 1.1
mok2	20	7000	0.1	0.05	65.9 ± 0.5
vowel	15	4000	0.2	0.2	83.4 ± 0.6

in the ensemble. We can see that, in general the methods based on *Stacked Generalization* are better than *Output Average*.

We have also calculated the percentage of error reduction (PER) of the ensembles with respect to a single network to get a general value for the comparison among all the methods we have studied. We have used equation 16 to calculate its value.

$$PER = 100 \cdot \frac{Error_{singlenetwork} - Error_{ensemble}}{Error_{singlenetwork}} \quad (16)$$

Table 4. Results for the ensemble of three networks

	aritm	glas	ionos	mok1	mok2	Vowel
Average	73.5 ± 1.1	94 ± 0.8	91.1 ± 1.1	98.3 ± 0.9	88 ± 2.5	88 ± 0.9
Vote	73.1 ± 1	93.6 ± 0.9	91.3 ± 1	98.3 ± 0.9	88 ± 2.2	86.9 ± 0.9
WTA	73.6 ± 1	94 ± 0.6	91.1 ± 1.1	98.1 ± 1	88 ± 2.4	86.7 ± 0.8
Borda	73.1 ± 1	94.4 ± 0.9	91.3 ± 1	98.3 ± 0.9	88 ± 2.2	85.9 ± 1
Bayesian	73.6 ± 0.9	94.2 ± 1	91.4 ± 1.1	98.4 ± 0.9	88.8 ± 2.4	86.4 ± 1
DAN	73.2 ± 1.1	92.8 ± 1.6	90 ± 1.2	97.1 ± 1	87 ± 2.2	84.6 ± 1.2
Stacked	75.4 ± 1.4	95.2 ± 0.9	92 ± 0.8	98.4 ± 0.9	88.8 ± 2.3	89.4 ± 0.8
Stacked +	74.4 ± 1.4	95.6 ± 0.9	92 ± 0.9	99.8 ± 0.3	88.5 ± 2.5	89.8 ± 0.8

Table 5. Results for the ensemble of nine networks

	aritm	glas	ionos	mok1	mok2	Vowel
Average	73.8 ± 1.1	94 ± 0.7	90.3 ± 1.1	98.8 ± 0.8	90.8 ± 1.8	88 ± 0.9
Vote	73.3 ± 0.9	93.2 ± 0.8	90.6 ± 1.2	98.3 ± 0.9	90.3 ± 1.8	88 ± 0.9
WTA	73.3 ± 1.1	93.8 ± 0.6	90.9 ± 1.3	99.5 ± 0.5	90 ± 1.2	88 ± 0.9
Borda	73.3 ± 0.9	94.2 ± 0.7	90.6 ± 1.2	98.3 ± 0.9	90.3 ± 1.8	88 ± 0.9
Bayesian	73.6 ± 0.9	92.2 ± 0.9	93.1 ± 1.4	99.8 ± 0.3	89.6 ± 1.7	88 ± 0.9
DAN	73.6 ± 1	92.8 ± 1.1	90 ± 1.1	98.8 ± 0.9	86.8 ± 2.8	88 ± 0.9
Stacked	75.1 ± 1.2	96 ± 0.7	92.9 ± 1	99.8 ± 0.3	92.1 ± 1.2	88 ± 0.9
Stacked +	73.6 ± 1.7	95.6 ± 0.8	92.7 ± 1	100 ± 0	91.9 ± 1.3	92.3 ± 0.6

Table 6. Results for the ensemble of twenty networks

	aritm	glas	ionos	mok1	mok2	Vowel
Average	73.8 ± 1	94 ± 0.7	90.4 ± 1	98.3 ± 0.9	91.1 ± 1.1	91.4 ± 0.8
Vote	73.3 ± 1	93.4 ± 0.9	90 ± 1.2	98.1 ± 1	90.4 ± 1.8	90.6 ± 0.6
WTA	73.1 ± 1.2	94.4 ± 0.7	91.3 ± 1.1	100 ± 0	90 ± 1.1	89.7 ± 0.7
Borda	73.3 ± 1	94.4 ± 0.8	90 ± 1.2	98.1 ± 1	90.4 ± 1.8	88 ± 0.9
Bayesian	73.8 ± 1	90.6 ± 0.9	93.1 ± 1.4	100 ± 0	89.9 ± 1.6	74.9 ± 1
DAN	72.8 ± 1.2	94.2 ± 1.2	89.6 ± 1.1	97.6 ± 1	86.6 ± 2.1	85.3 ± 1.1
Stacked	73.8 ± 1.3	96.6 ± 0.8	92.7 ± 1.1	100 ± 0	91.5 ± 1.1	93.3 ± 0.6
Stacked +	74.7 ± 1.1	96.6 ± 0.8	92.9 ± 1.2	100 ± 0	91.5 ± 1.1	93.3 ± 0.7

Table 7. Results for the ensemble of forty networks

	aritm	glas	ionos	mok1	mok2	Vowel
Average	73.8 ± 1.1	94.2 ± 0.6	90.3 ± 1	98.3 ± 0.9	91.1 ± 1.2	92.2 ± 0.7
Vote	73.5 ± 1	94 ± 0.8	90.1 ± 1.2	98.3 ± 0.9	91 ± 1.6	90.5 ± 0.7
WTA	73.1 ± 1.2	93.8 ± 0.9	91.6 ± 1.1	99.6 ± 0.4	90 ± 1.6	89.5 ± 0.7
Borda	73.5 ± 1	94.4 ± 0.8	90.1 ± 1.2	98.3 ± 0.9	91 ± 1.6	88.7 ± 0.8
Bayesian	74.1 ± 1.1	90.2 ± 0.9	93.4 ± 1.4	100 ± 0	90.3 ± 1.5	67.7 ± 1.3
DAN	73.2 ± 1	93.2 ± 0.9	89 ± 1.2	98.8 ± 0.8	86.4 ± 2.8	84.3 ± 1.2
Stacked	73.9 ± 1.4	95.8 ± 0.6	92.4 ± 1	100 ± 0	92.4 ± 1.2	94.2 ± 0.8
Stacked +	74.5 ± 1.3	96.6 ± 0.8	92.4 ± 1.2	100 ± 0	91.4 ± 1.2	94.1 ± 0.7

Table 8. *Stacked Generalization* increase of performance with respect to *Average*

Database	3 Nets	9 Nets	20 Nets	40 Nets
aritm	1.95	1.27	0	0.11
glas	1.2	2	2.6	1.6
ionos	0.85	2.56	2.27	2.14
mok1	0.12	1	1.75	1.75
mok2	0.75	1.38	0.37	1.25
vowel	1.41	1.36	1.92	2.02

Table 9. *Stacked Generalization Plus* increase of performance with respect to *Output Average*

Database	3 Nets	9 Nets	20 Nets	40 Nets
aritm	0.92	-0.24	0.91	0.68
glas	1.6	1.6	2.6	2.4
ionos	0.85	2.41	2.42	2.14
mok1	1.5	1.25	1.75	1.75
mok2	0.5	1.13	0.37	0.25
vowel	1.81	1.36	1.92	1.92

The PER value ranges from 0%, where there is no improvement by the use of a particular ensemble method with respect to a single network, to 100%. A negative value means that the performance of the ensemble is worse.

Furthermore, we have calculated the increase of performance with respect to *Single Network* (Table 10) and the mean PER (Table 11) across all databases for each method to get a global measurement.

According to these global measurement *Stacked Generalization* methods are the best performing methods. The highest difference between *Stacked Generalizacion* and *Output Average* is in the 40-network ensemble where the mean PER increase is 9.54%. Although, *Stacked Generalization Plus* is slitghly better than *Stacked Generalization* there are some cases where the second method is better.

Table 10. Mean increase of performance across all databases with respect to Single Network

Method	3 Nets	9 Nets	20 Nets	40 Nets
Average	11.2	12.15	12.23	12.38
Vote	10.91	11.6	11.7	11.95
WTA	10.98	12.03	12.14	11.99
Borda	10.88	11.42	11.44	11.72
Bayesian	11.18	10.85	9.45	8.35
DAN	9.85	10.34	10.07	9.88
Stacked	12.25	13.75	13.72	13.86
Stacked Plus	12.4	13.41	13.9	13.9

Table 11. Mean performance of error reduction across all databases

Method	3 Nets	9 Nets	20 Nets	40 Nets
Average	49.17	49.66	50.16	50.94
Vote	46.94	47.18	47.55	48.57
WTA	48.41	49.43	50.05	49.52
Borda	45.68	45.87	45.73	47.05
Bayesian	38.19	43.61	35.21	28.52
DAN	39.35	41.05	39.65	38.09
Stacked	56.78	58.3	58.56	58.98
Stcaked+	56.91	56.8	59.4	59.4

4 Conclusions

In the present paper we have analysed six classical combination methods and we have proposed two methods based on *Stacked Generalization*. We have used ensembles of 3, 9, 20 and 40 networks previously trained with *Simple Ensemble* on six databases from the *UCI Repository* to cover a wide spectrum of the number of networks in the classification system.

The results showed that the improvement by the use of *Stacked Generalization* depends on the database. Moreover, we have calculated the mean increase of performance and the mean percentage of error reduction across all databases with respect to a *Single Network* in order to get global measurements to compare the combination methods we have studied. According to the results of these global measurements *Stacked Generalization* methods perform better than the classical combination methods studied in this paper. In general, *Stacked Generalization* is the best performing combination method for ensembles of 9 networks and *Stacked Generalization Plus* is the best performing combination method for ensembles of 3, 20 and 40 networks.

We can conclude that the use of a *Combination Network* in the module combination of an ensemble increases the generalization capability of the ensemble.

Acknowlegments

This research was supported by project *P1·1B2004-03* of Universitat Jaume I - Bancaja in Castellón de la Plana, Spain.

References

1. Tumer, K., Ghosh, J.: Error correlation and error reduction in ensemble classifiers. Connection Science **8**(3-4) (1996) 385–403
2. Raviv, Y., Intratorr, N.: Bootstrapping with noise: An effective regularization technique. Connection Science, Special issue on Combining Estimators **8** (1996) 356–372
3. Hernandez-Espinosa, C., Fernandez-Redondo, M., Torres-Sospedra, J.: Ensembles of multilayer feedforward for classification problems. In: Neural Information Processing, ICONIP 2004. Volume 3316 of Lecture Notes in Computer Science. (2005) 744–749
4. Hernandez-Espinosa, C., Torres-Sospedra, J., Fernandez-Redondo, M.: New experiments on ensembles of multilayer feedforward for classification problems. In: Proceedings of International Conference on Neural Networks, IJCNN 2005, Montreal, Canada. (2005) 1120–1124
5. Torres-Sospedra, J., Fernandez-Redondo, M., Hernandez-Espinosa, C.: A research on combination methods for ensembles of multilayer feedforward. In: Proceedings of International Conference on Neural Networks, IJCNN 2005, Montreal, Canada. (2005) 1125–1130
6. Xu, L., Krzyzak, A., Suen, C.: Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Transactions on Systems, Man, and Cybernetics **22**(3) (1992) 418–435
7. Verikas, A., Lipnickas, A., Malmqvist, K., Bacauskiene, M., Gelzinis, A.: Soft combination of neural classifiers: A comparative study. Pattern Recognition Letters **20**(4) (1999) 429–444
8. Jimenez, D., Walsh, N.: Dynamically weighted ensemble neural networks for classification. In: IEEE World Congress on Computational Intelligence. Volume 1. (1998) 753–756
9. Wolpert, D.H.: Stacked generalization. Neural Networks **5**(6) (1994) 1289–1301
10. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI repository of machine learning databases (1998)
11. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: International Conference on Machine Learning. (1996) 148–156
12. Breiman, L.: Arcing classifiers. The Annals of Statistics **26**(3) (1998) 801–849
13. Kuncheva, L., Whitaker, C.J.: Using diversity with three variants of boosting: Aggressive. In: Proceedings International Workshop on Multiple Classifier Systems , Calgiari, Italy, June 2002. Springer. Volume 2364 of Lecture Notes in Computer Science., Springer (2002)