An Investigation into How ADABOOST Affects Classifier Diversity

Catherine A. Shipp
School of Informatics,
University of Wales, Bangor,
Bangor, Gwynedd,
LL57 1UT, United Kingdom
map802@bangor.ac.uk

Ludmila I. Kuncheva
School of Informatics,
University of Wales, Bangor,
Bangor, Gwynedd,
LL57 1UT, United Kingdom
mas00a@bangor.ac.uk

Abstract

AdaBoost is a method for incrementally creating a classifier ensemble. We investigate how the diversity of an ensemble of classifiers created by AdaBoost varies as the number of classifiers in the ensemble increases. We consider two data sets from the UCI machine learning repository and use ten different measures of diversity. We show that AdaBoost does indeed initially increase the diversity but after the first few classifiers the diversity begins to gradually tail off. These results suggest that useful classifier ensembles can be recovered at an early stage of AdaBoost training, perhaps using a more sophisticated combination method than the weighted voting.

Keywords
Combining classifiers, AdaBoost, diversity, dependence.

1 Introduction

Combining classifiers is an established research area in the field of statistical pattern recognition. If we have many different classifiers at our disposal it is sensible to combine them in the hope of increasing the overall accuracy [10]. It has been proved theoretically that a group of independent classifiers improve upon the single best classifier when majority vote combination is used. It is assumed that the improvement holds for other combination methods as well. Intuitively, the classifiers to be combined should be different from each other, or 'diverse'. Using different classifiers can result in both better performance and worse performance, depending on the differences. Thus diversity can be both beneficial or harmful [8, 14]. Understanding and measuring these differences in diversity is an important issue in classifier combination [15] and there are several different measures of diversity being used. These measures aim to quantify the dependence between classifiers.

Several techniques exist which aim to improve the performance of classifier ensembles by manipulating the data set which classifiers are trained on. These include Bagging, Boosting, and Arcing [2, 3]. We are particularly interested in the AdaBoost[1] algorithm which has had considerable success with artificial and real-world data problems [1]. AdaBoost builds a classifier ensemble by starting with one classifier and adding new classifiers, one at a time. The new classifier is constructed by modifying the training set according to the previous classifier's performance [2,4,18]. It has been found that Boosting can be paralysed [21], i.e., no further improvement is achieved when adding new classifiers to the team.

Here we study how diversity changes in the ensemble as the number of classifiers produced by AdaBoost increases and try to establish whether or not AdaBoost works by focusing on altering the diversity. The next

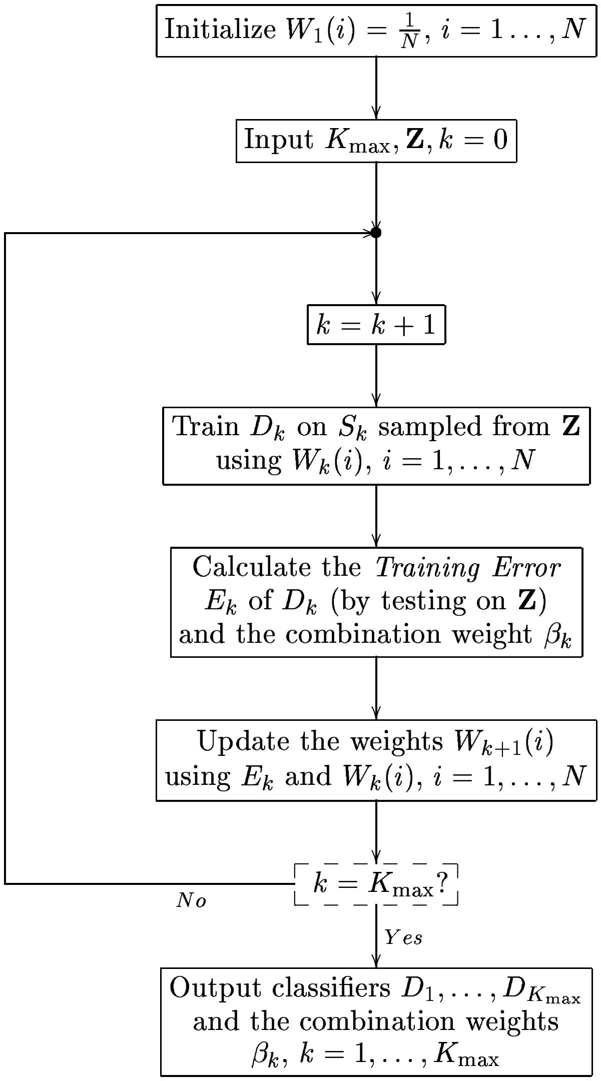
[1] The name comes from **Ada**ptive **Boost**strapping.

section contains a detailed account of the AdaBoost algorithm. Section 3 contains a brief description of the diversity measures. Section 4 contains details of the experimental set-up, Section 5 contains the results and Section 6 offers a conclusion.

2 AdaBoost

AdaBoost constructs classifiers by modifying the training set based on the previous classifier's performance. It does this by getting the new classifier to put more emphasis on those objects which the previous classifier found difficult to classify accurately. This is achieved by maintaining a distribution of weights over the training set, which can be modified as required on each iteration. Some implementations of AdaBoost use a resampling method [2] and others use re-weighting [1]. These differ according to whether you resample from the original training set or attach weights to each data point and re-use the whole training set to build the next classifier. The choice of implementation does not affect AdaBoost too much although boosting with re-weighting is a more direct implementation of the theory [1]. Research by Breiman suggests that there is very little difference in the results obtained using the two methods [2]. We have used a resampling method because this allows for any type of basic classifiers to be used.

For the resampling implementation, each weight determines the probability of its associated object being selected for the training set for an individual component classifier [4]. Initially all weights are set equal. On each round if a training object is not accurately classified then its chances of being selected again for a subsequent training set are increased by increasing the value of its associated weight [18]. In this way the next classifier is forced to concentrate on the more difficult examples in the training set. In Figure 1 we show the basic algorithm for AdaBoost using the resampling implementation.


Initialize $W_1(i) = \frac{1}{N}$, $i = 1\ldots, N$
Input $K_{\max}, \mathbf{Z}, k = 0$
$k = k + 1$
Train D_k on S_k sampled from $\mathbf{Z}$ using $W_k(i)$, $i = 1, \ldots, N$
Calculate the *Training Error* E_k of D_k (by testing on $\mathbf{Z}$) and the combination weight β_k
Update the weights $W_{k+1}(i)$ using E_k and $W_k(i)$, $i = 1, \ldots, N$
$k = K_{\max}?$
No
Yes
Output classifiers $D_1, \ldots, D_{K_{\max}}$ and the combination weights β_k, $k = 1, \ldots, K_{\max}$

KEY:

$\mathbf{Z}$ the training set;
S_k the training set for iteration k;
D_k the classifier trained at iteration k;
E_k the training error at iteration k;
$W_k(i)$ the weight for object i at iteration k;
$W_k = \{W_k(1), \ldots, W_k(N)\}$ the set of weights used at iteration k.

Figure 1: THE ADABOOST ALGORITHM: THE RESAMPLING IMPLEMENTATION

The combination weights β_k are obtained as follows

$$\beta_k = \frac{1 - E_k}{E_k}. \qquad (1)$$

When $E_k = 0$, the weights $W_k(i)$ are reinitialized to $1/N$. The weights at iteration k are calculated as

$$W_{k+1}(i) = \frac{W_k(i)\beta_k^{d(i)}}{\sum_{j=1}^{N} W_k(j)\beta_k^{d(j)}}, \qquad (2)$$

where $d(i) = 1$ if the current classifier (D_k) gives the incorrect label of object i, and $d(i) = 0$, otherwise. The final decision for a new object $\mathbf{x}$ is made by weighted voting between the $K_{\max}$ classifiers. First, all classifiers label $\mathbf{x}$ and then for all D_k that gave label ω_t, we calculate the support for that class by

$$\mu_t(\mathbf{x}) = \sum_{D_k(\mathbf{x})=\omega_t} \ln(\beta_k). \qquad (3)$$

The class with the maximal support is chosen for $\mathbf{x}$.

Typically, the generalization error of AdaBoost decays exponentially. In this study we were interested in discovering patterns of diversity as the building of the classifier ensemble progresses, so the problem of accuracy was left out. By the idea of its design, AdaBoost would enforce diversity in the team, making new classifiers focus on different parts of the data set. However, it was found out that the method might get to an inefficient stage where adding new classifiers will not improve the performance but on the contrary, deteriorate it [21]. This is caused by sampling repetitively from similar distributions whose resultant training set poses the same difficulty to the chosen classifier model. Then the ensemble will be overpopulated with “semi-good” and almost identical experts, and the effect of some good quality members of the team will be diluted. Our study of diversity aims at showing a pattern of diversity during training of the ensemble to support this hypothesis.

3 Measures of Diversity

There are different diversity measures available from different fields of research. Some of these measures, such as the Q-statistic and the correlation coefficient have come directly from mainstream statistics whilst others have developed through the field of statistical pattern recognition, specifically for the problems of multiple classifier systems. The first four of these measures are called pair-wise because they consider the $K_{\max}$ classifiers two at a time and then average the calculated pair-wise diversity. The other measures work on the whole group of $K_{\max}$ classifiers. The ten measures of diversity used in this study are taken from [12, 13] and described in Table 1. Those measures which have a (↓) type are such that the *lower* the value, the more diverse the classifiers, whilst those which have a (↑) type are such that the *higher* the value, the more diverse the classifiers.

Table 1: MEASURES OF DIVERSITY

Name	Notation	Type	Source
Q-statistic	Q	(↓)	[22]
Correlation coefficient	ρ	(↓)	[20]
Disagreement	D	(↑)	[9,19]
Double-fault	DF	(↓)	[6]
Kohavi-Wolpert variance	kw	(↑)	[11]
Measurement of interrater agreement	κ	(↓)	[5]
Entropy	Ent	(↑)	[13]
Measure of difficulty	θ	(↓)	[7]
Generalised diversity	GD	(↑)	[17]
Coincident failure diversity	CFD	(↑)	[16]

4 Experimental Set-up

For our experiments we used the Pima Indian Diabetes database and the Haberman Survival database, both taken from the UCI Repository of Machine Learning Database[2].

In summary: Pima Diabetes Database: 2 classes, 768 patients, 8 features; Haberman

[2] http://www.ics.uci.edu/~mlearn/MLRepository.html

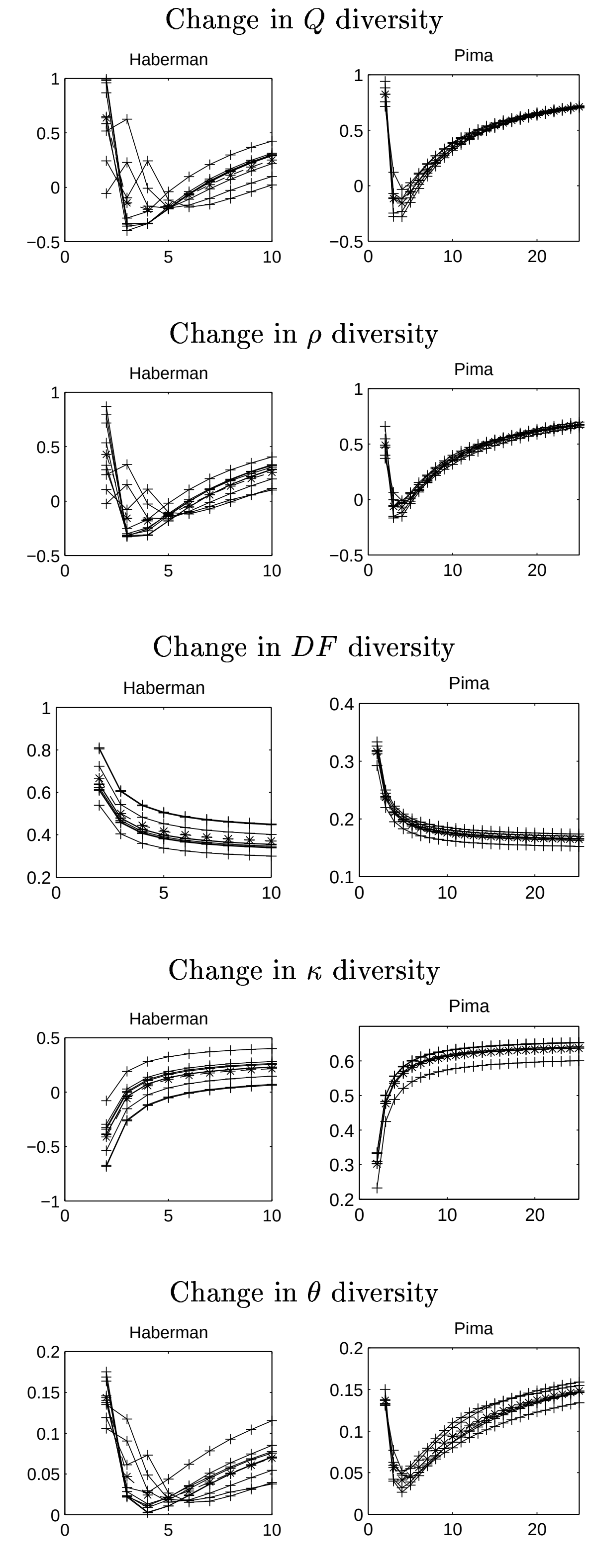


Figure 2: Change in the (↓) measures of diversity as we add classifiers to the ensemble

Survival Database: 2 classes, 306 patients, 3 features. Since generalization accuracy was not an issue here, we used the whole data sets for training and measuring diversity.

We used AdaBoost (Section 2) to generate ensembles of 10 linear classifiers for the Haberman data (the smaller data set), and 25 linear classifiers for the Pima data. As each new classifier was added to the ensemble we measured the diversity of the ensemble using the ten measures (Section 3). Finally we plotted the change in diversity against the number of classifiers for each dataset and each diversity measure. It was clear from our preliminary investigation that the change in diversity followed a similar pattern for each run. Therefore, we only repeated each experiment ten times to get an overall result.

5 Results

Figures 2 and 3 show the changes in diversity as we add classifiers to the ensemble. Figure 2 shows those diversity measures for which the lower the value the more diverse the classifiers (↓), whilst Figure 3 shows those diversity measures for which the higher the value the more diverse the classifiers (↑).

As Figure 2 shows, Q, ρ, and θ all have very similar 'tick-shaped' graphs which indicate that the diversity of the ensemble increases sharply as the first few classifiers are added to the ensemble, but then gradually decreases as we add more classifiers to the ensemble. DF and κ both have monotonic curves but whilst κ is steadily increasing, indicating that diversity gradually reduces, the DF's pattern suggests that the diversity is increasing gradually as we add more classifiers to the ensemble. Figure 3 shows curves with a similar 'tick-shape' (but inverted) to those of Q, ρ and κ, namely those of Ent, GD, and CFD. These, given that they are of the (↑) type, also indicate that the diversity of the ensemble increases sharply as the first few classifiers are added, but then gradually decreases as we add more classifiers. D on the other hand has a curve similar to an inverted version of κ indicating, like κ, that diversity gradually reduces as we add classifiers to the ensemble. kw has a slightly gentler, less-sharp curve for the Haberman data set but the familiar tick-shape for the Pima data set, both of which indicate an increase and then a decrease in diversity. As expected, due to the smaller sample size, the variability among the ten repetitions was greater with the Haberman set.

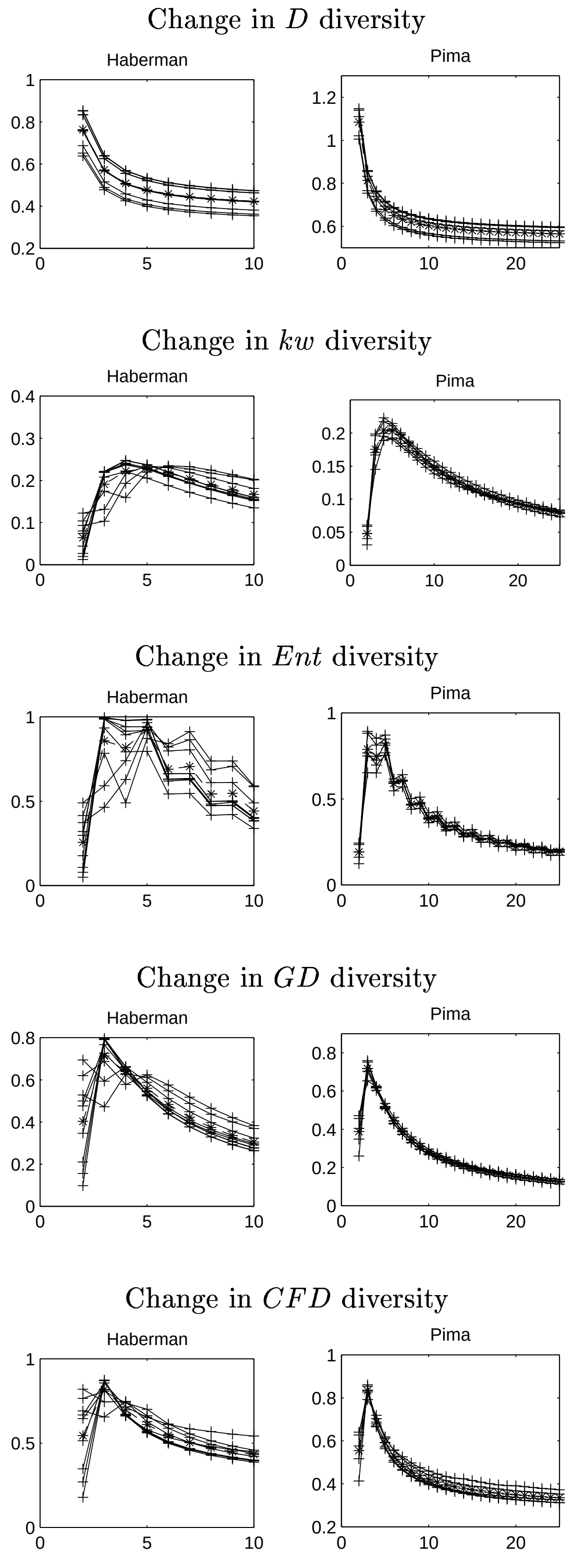
As discussed earlier, we would expect diversity to decline as the ensemble grows beyond a certain point. The AdaBoost resampling procedure will keep adding new classifiers built on similar samples of training objects which appear to be equally difficult for the successive members of the ensemble. The results with both data sets supported this hypothesis except for the Double Fault (DF) measure which showed increasing diversity throughout the training. The reason for that could be that DF is related to the accuracy of the team more than the other measures and is therefore not exactly a “true” measure of *diversity* [12].

It is interesting to see how the dramatic increase in diversity as the first few classifiers are added to the ensemble (tick-shaped curves), can be exploited to its potential. Perhaps stopping the training there and applying different combination rules will be beneficial.

6 Conclusions

We study the diversity of classifier ensembles built through the AdaBoost algorithm. The algorithm is explained in Section 2 and experimental results with two data sets from UCI ML repository are given. We found a consistent pattern of diversity showing that at the beginning, the new classifiers are highly diverse but as the training progresses, the diversity gradually returns to its starting level. This suggests that it could be beneficial to stop AdaBoost before diversity drops, and try to enhance the performance of the ensemble further by applying different combination methods on a reasonably accurate and diverse ensemble.

We obtained consistent patterns of diversity with almost all diversity measures and both data sets using *linear* base classifiers. However, the patterns might change if other classifier models are used.


Change in D diversity

Change in kw diversity

Change in Ent diversity

Change in GD diversity

Change in CFD diversity

Figure 3: Change in the (↑) measures of diversity as we add classifiers to the ensemble

References

[1] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–139, 1999.

[2] L. Breiman. Combining predictors. In A. J. C. Sharkey, editor, *Combining Artificial Neural Nets*, chapter 2, pages 31–50. Springer-Verlag, 1999.

[3] H. Drucker. Boosting neural networks. In A. J. C. Sharkey, editor, *Combining Artificial Neural Nets*, chapter 3, pages 51–78. Springer-Verlag, 1999.

[4] R.O. Duda, P.E.Hart, and D.G. Stork. *Pattern Classification*, chapter 9, pages 453–516. John Wiley & sons, New York, 2nd edition, 2001.

[5] J.L. Fleiss. *Statistical methods for Rates and Proportions.* John Wiley & Sons, 1981.

[6] G. Giancinto and F. Roli. Design of effective neural network ensembles for image classification processes. *Image, Vision and Computing Journal*, (to appear).

[7] L.K. Hansen and P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, 1990.

[8] S. Hashem. Treating harmful collinearity in neural network ensembles. In A. J. C. Sharkey, editor, *Combining Artificial Neural Nets*, chapter 5, pages 101–125. Springer-Verlag, 1999.

[9] T.K Ho. The random space method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[10] A.K. Jain, R.P.W. Duin, and J. Mao. Statistical pattern recognition: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.

[11] R. Kohavi and D.H. Wolpert. Bias plus variance decomposition for zero-one loss functions. In L. Saitta, editor, *Machine Learning: Proc. 13th International Conference*, pages 275–283. Morgan Kaufman, 1996.

[12] L.I. Kuncheva and C.J. Whitaker. Measures of diversity in classifier ensembles. *(submitted)*, 2001. 'available at http://www.bangor.ac.uk/~mas00a/papers/lkml.ps.gz'.

[13] L.I. Kuncheva and C.J. Whitaker. Ten measures of diversity in classifier ensembles: limits for two classifiers. In *IEEE Workshop on Intelligent Sensor Processing*, Birmingham, UK, 10 2001. ISP2001.

[14] L.I. Kuncheva, C.J. Whitaker, C.A. Shipp, and R.P.W. Duin. Is independence good for combining classifiers? In *Proc. 15th International Conference on Pattern Recognition*, volume 2, pages 169–171, Barcelona, Spain, 2000.

[15] L. Lam. Classifier combinations: implementations and theoretical issues. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, pages 78–86, Cagliari, Italy, 2000. Springer.

[16] D. Partridge and W.J. Krzanowski. Distinct failure diversity in multiversion software. (personal communication 1999).

[17] D. Partridge and W.J. Krzanowski. Software diversity: practical statistics for its measurement and exploitation. *Information & Software Technology*, 39:707–717, 1997.

[18] R.E. Schapire. Theoretical views of boosting. In *Computational Learning Theory: Fourth European Conference*, pages 1–10. EuroCOLT'99, 1999.

[19] D.B. Skalak. The sources of increased accuracy for two proposed boosting algorithms. In *Proc. American Association for Artificial Intelligence*, Integrating Multiple Learned Models Workshop. AAAI, 1996.

[20] P.H.A. Sneath and R.R. Sokal. *Numerical Taxonomy.* W.H. Freeman & Co., 1973.

[21] J. Wickramaratna, S. Holden, and B. Buxton. Performance degradation in boosting. In J. Kittler and F. Roli, editors, *Multiple Classifier Systems*, volume 2096 of *Lecture Notes in Computer Science*, pages 11–21. Springer, 2001.

[22] G.U. Yule. On the association of attributes in statistics. *Phil. Trans. A*, 194:257–319, 1900.