Co-operative Training in Classifier Ensembles

Nayer Wanas Rozita Dara Mohamed S. Kamel
PAMI Lab
University of Waterloo
Waterloo, N2L 3G1
Canada
{nwanas,rdara,mkamel}@uwaterloo.ca

Abstract – *As the possibilities of combining experts become a more important direction in intelligent systems, difficulties arise in ways of generating these various experts and how to effectively use them concurrently. This is very evident in designing multiple classifier systems. The degree and method by which multiple classifier systems share training resources among their components can be a measure of co-operation. Training resources that are sharable in a multiple classifier system are training patterns, algorithms or information. In this paper we present co-operative training as a means of sharing training information amongst an ensemble during training. Improved classification accuracy demonstrates that sharing, or co-operation, amongst classifiers during their training is useful in a multiple classifier system.*

Keywords: Decision fusion, Co-operative training, Sharing, Multiple classifiers

1 Introduction

There has been a growing interest in the area of combining classifiers ensembles in recent years. This is motivated by the fact that using the classification capabilities of multiple experts tend to yield an improved performance over that of a single expert. It also improves the classifiers reliability and generalization ability [1]. Most research in Multiple Classifier Systems (MCS) has focused on developing different approaches or architectures to perform combining by selection and/or fusion. Diversity among members of an ensemble has always been emphasized as a key to the sucess of the combining approach.

Sharkey [1, 2] presented an account of categorization of the various multiple classifier approaches. The basis of the categorization is the distinction between ensemble and modular approaches, as well as differentiating between co-operative and competitive approaches. Kuncheva et al. [3] introduced a measure of diversity, the Q measure, and compared the performance of two ensemble creating approaches, boosting and bagging. They concluded that diversity is generally beneficial, and that it should be used to establish and investigate ensemble building methodologies. However, they couldn't find a relationship between the level of diversity and the accuracy of MCS. Along with the shift to more complex and incremental training algorithms for MCS, the perspective of diversity is no longer sufficient. The degree or method by which components of the ensemble *share* their training resources, or in other words the level of *co-operation*, becomes important [4]. Training resources in MCS are the training patterns, training algorithms, or training information. A measure of sharing at training can be used to categorize how systems utilize training resources. This would lead to more intelligent choices when designing MCS. In the following we will shed some light on this concept of sharing training resources.

2 Shared Training Resources

2.1 Sharing Training Patterns

Components within MCS may share training patterns by using identical or overlapping training sets. While MCS that do not exhibit sharing of training patterns are trained using disjoint sets. In both cases, the generated ensemble might have a degree of diversity; however, the diversity is generated from different sources in both cases. If members of the ensemble are to be trained on disjoint data sets, the bulk of the diversity would be expected to be generated from the diversity within the data patterns. While the main component of the diversity in ensembles that share training patterns is expected from the different generalization abilities of the ensemble members. Most approaches use some sort of sharing of training patterns, among the methods are Bagging, Boosting, and k-fold cross validation. Brieman [5] introduced the concept of Bagging, which presents each classifier with a set of data points sampled uniformly with replacement from the original data points. This creates a resampled data set in which some points might appear multiple times or not at all. In unstable classifiers, small changes in the training data will produce a large change in the output generalization. As a result, bagging will produce a diverse ensemble. The final classification decision is attained by taking the majority vote of the generated ensemble. Another method that helps force diversity, is by using k-fold cross validation. Here, the training set is randomly divided into k subsets. Each classifier is presented with $k-1$ subsets to use in training, leading to some diversity within the ensemble. Other methods belonging to this category are random forests [6], random trees [7], as well as general data manipulating techniques.

2.2 Sharing Training Algorithms

Sharing of training algorithms can be used to distinguish between ensembles of classifiers that use homogenous and heterogeneous classifiers. The diversity in homogenous classifiers is generated by varying the set of initial conditions or classification parameters. While heterogenous classifiers would employ the use of different training algorithms to produce different generalizations. This is especially useful when MCS use data from different input sources. This can be achieved when different sensors are used, and when these sensors collect different kinds of information. Although, in principle, a set of classifiers can vary in terms of their weights, initial conditions, or architecture, yet they constitute the same solution [1]. Therefore, sharing training algorithms might not be effective unless the different classifiers result in different patterns of generalization when they are tested.

2.3 Sharing Training Information

Finally, sharing training information in MCS can be identified as exchanging information among components to improve their accuracy. If each classifier is constructed independently, then such an ensemble will not exhibit sharing of training information. Although this ensemble model has many benefits, there are still some disadvantages. In practice, we see that although the individual classification accuracy of some of the classifiers may be high, the final classification accuracy can be much lower [1]. This is due in part to the fact that the decision fusion mechanism may not have enough information about the accuracy expected from a module during the testing phase. This lack of interaction among the different classifiers may generate individual classifiers that do not contribute to the ensemble. The sharing of training information would allow the aggregation module to carry out a more informed fusion process. Hence, the relative importance of each of the modules' accuracy is just as significant as its behavior amongst the ensemble members. This can be attained by using coupled or *co-operative training* [8].

Among the approaches that generate ensembles using co-operative training, is Boosting. Schapire [9] has demonstrated that a series of weak learners can be converted to strong learners as a result of training the members of an ensemble on previously filtered patterns. These patterns are filtered by previously trained members of the ensemble. A number of empirical studies [10, 11] have supported the efficiency of the boosting algorithm, although one problem is that it requires large amounts of data. Freund and Schapire [12] have proposed an algorithm, Adaboost, that largely avoids this problem. Essentially, in this algorithm, the training sets are adaptively re-sampled, so that the weights in the re-sampling are increased for those cases which are most often misclassified. Boosting and AdaBoost can be referred to as *generalized additive models* [13]. *Pasting votes* proposed by Brieman [14] is among the approaches that construct additive models. Generally, these methods add classifiers as necessary, while selecting their training set, to improve the overall accuracy. Liu and Yao [15] proposed a co-operative ensemble learning system (CELS) to encourage several neural network classifiers to learn different aspects of the training data. In this approach, the problem is decomposed into smaller more specialized sub-problems through training. CELS emphasizes interaction among the individual networks, through encouraging specialization by using an unsupervised penalty term. This penalty term is included in the error function produces biased individual networks with negative correlation. This approach would allow members of the ensemble to learn the whole training set better.

Auda and Kamel [16] presented the EVOL architecture which tries to achieve autonomous training of an ensemble. In this approach, members of the ensemble utilize voting results to direct the training process. During training, EVOL gates each individual training input to one of the classifiers in the ensemble. This approach enhances the diversity between the ensemble modules, but lacks any guarantee of convergence. Wanas et. al. [17] and Kamel and Wanas [18] presented an adaptive training algorithm that addresses the two observations made concerning classifier ensembles. This algorithm is capable of autonomously directing the training of each classifier in the ensemble by selecting the training patterns to be used in re-training. This re-training is based on both the performance of the ensemble and the performance of the individual classifier. This incremental learning algorithm allows members of the ensemble to enhance their diversity during training. In the following, we will shed some light upon this algorithm within the context of *sharing*. We will also investigate the effect of sharing training algorithms on the accuracy of MCS.

3 Sharing Training Information in Classifier Ensembles: An Algorithm

In this algorithm, we introduce a level of sharing amongst the members of the ensemble during training. The adaptive training allows the final classification by the aggregation layer to determine whether or not further training should be carried out at the module level [19]. This will facilitate the exchange and sharing of information regarding the training of each classifier within the ensemble. This information would increase the accuracy of the individual classifiers and help improve the final and overall accuracy.

Figure 1 shows the flowchart of this training algorithm. We note here that the data must be divided into training, testing and evaluation sets. All these sets should be distinct from one another. They must also include representative vectors from all the classes. Let C_i represent classifier i, CF_i, the confidence factor of the classification from C_i. The confidence factor acts as a measure of this ability since it is based on the proportion of incorrectly classified records relative to the total number of records in the evaluation set. Let CF_{i_best} be the best confidence factor obtained for C_i during training. We also define Err as the number of evaluation samples in error,and K is defined as the number of classifiers. The constants Γ, a user defined threshold such that $0.0 < \Gamma < 1.0$, and P, the base percentage such that $0 < P < 100\%$, and δ the modifier for P, are all selected for the algorithm.

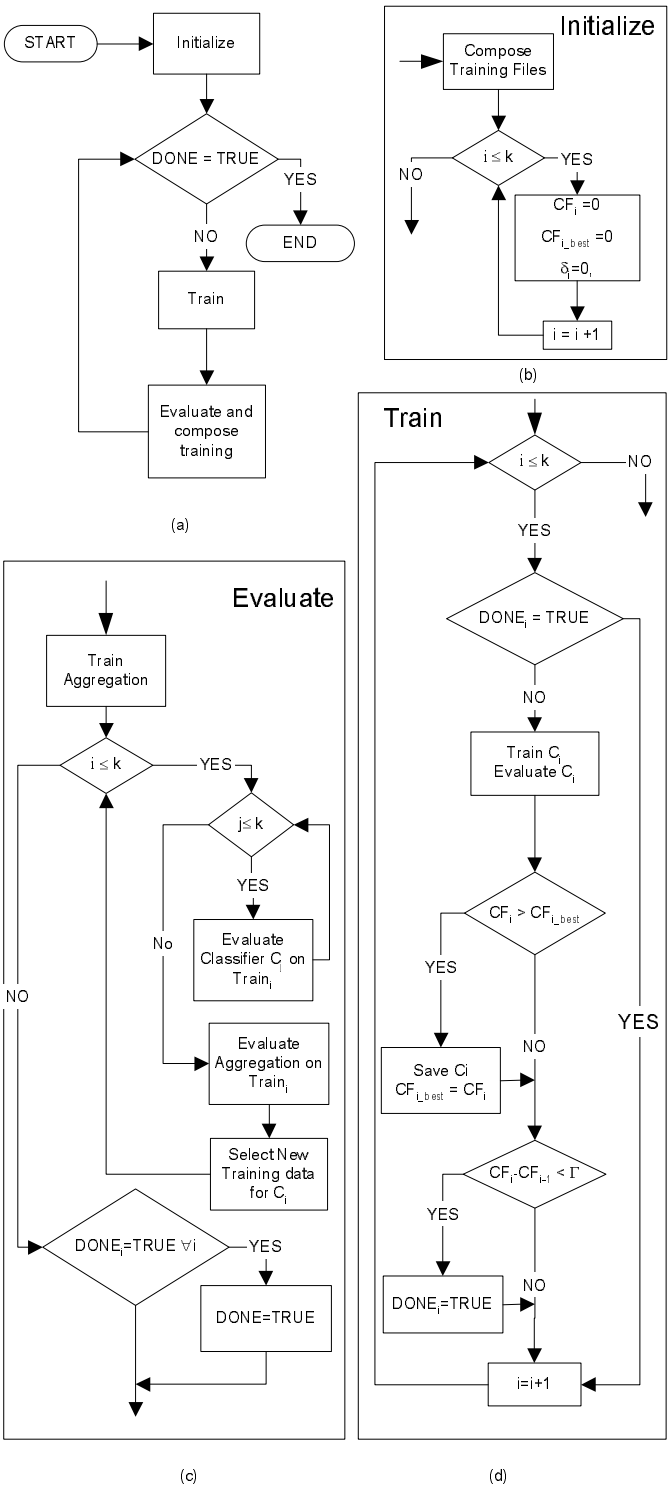

Fig. 1: Adaptive training algorithm, (a) Basic Algorithm which included (b) An Initialization Loop, (c) An Evaluation Algorithm, (d) A Training algorithm

The training of the classifiers is done iteratively. A sequence of training and testing cycles are performed. The algorithm utilizes the evaluation set to measure how well each classifier is performing. During testing, we utilize the best weight set obtained for each of the modules. This assures the highest confidence factors for the modules' local decisions. The algorithm doesn't limit its evaluation of the performance with respect to each classifier alone, but also relative to the other modules. All the individual classifiers, C_j, are tested on the data ,$train_i$, used in training classifier C_i. This is followed by the evaluation of the aggregation module on the data. The mis-classifier and correctly classified patterns of $train_i$ after aggregation are used to compose the new training set for classifier C_i. Referring to the training algorithm, we continue training the network based on the classifications and votes that are in error. The algorithm randomly chooses records from the training set that represents the set of learned classes, in order not to destroy the classifications and votes already learnt by a module. The number of records chosen $\lceil Err \times P \times \delta/100 \rceil$, in this manner depends on the number of records in error and the user defined constant P. A higher error from the evaluation set results in more records being chosen from the training set that have been classified correctly. The number of records chosen to represent the correctly classified classes is therefore a function of the number of records chosen to represent those classes that were incorrectly classified.

The evaluation of the patterns used in retraining can be performed at different levels. If each classifier is evaluated based on its own performance, then this algorithm will reduce to a classifier improving method. The aggregated output of the classifier ensemble can be used to evaluate the performance of the classifiers and generate the new training sets. That is, the members of the ensemble are all tested using the training set of module C_i. The output of all the classifiers is then aggregated and the patterns are divided into two groups depending on the correct classification. This allows the members of the ensemble to share training information amongst each other, and couple their training. The algorithm may also be integrated into a hierarchal architecture [20], in this case sharing training information impacts both the classifiers and the subsequent levels of aggregation.

In the following experiments we study the advantage of sharing in the training algorithm. The objective of our experiments was to compare the classification abilities of the ensemble architecture using an independently trained members and one that exhibits sharing, namely the adaptive training algorithm presented.

4 Experimental Study

We implemented these experiments on the 20 class Gaussian data problem [21], and the Satimages database [22]. In all the tests presented in this investigation, we composed an ensemble of five classifiers. Each classifier was a one-hidden layer MLP neural network with ten sigmoidal hidden units. These classifiers are trained using the error back-propagation algorithm. The training sets were randomized before being presented to the networks. These classifiers are trained using disjoint training sets. Each classifier was trained for 1000 epochs, and was tested on the evaluation set every 100 epochs. Next, the network instance that provides the best classification is saved. The parameters of all the networks are maintained for all the classifiers that are trained. To reduce any external factors, the models are set up using the same platform, language, and implementation of the neural networks.

The performance of an ensemble classifiers trained independently and using the adaptive training algorithm are compared. The aggregation method used to evaluate the performance of the classifiers as the weighted average [23]. That is, the classifiers within the ensemble are trained on their respective training sets. Then, the weighting matrix was generated and used to aggregate the classifiers. This aggregated output is used to evaluate the classification accuracy of each classifier. In addition it was used to determine the patterns to be used in re-training and indicated in the algorithm.

In order to investigate the effect of sharing training information, the classifiers trained were combined using a variety of aggregation approaches. These approaches can be divided into two groups. The *voting approaches*, which include the majority, maximum, average Nash and Borda count, and the *trained approaches*, including the weighted average, Bayesian and fuzzy integrals. The classification error of these methods is also compared to the oracle of the ensemble. The oracle is defined as follows: assign the correct class label to x if at least one individual classifier produces the correct class label of x when its decision is hardened.

Table 1 compares the classification errors achieved with and without sharing for different combining schemes, as well as the corresponding variance. The results indicate a reduction in classification errors ranging from 1.5% to 7%. Although the improvement is consistent with all the aggregation approaches, the majority vote and Borda count experience the greatest improvement. The weighted average and Bayesian approach are among the best aggregation approaches. This improvement is due to the fact that, in their re-training, the classifiers are allowed to focus on improving the overall classification. The members of the ensemble are re-trained on patterns in their training sets that have been misclassified after aggregation. Hence, the ensemble increases its classification accuracy collectively not individually. Consequently, the diversity in the classifiers is enhanced by using coupled training. This diversity is reflected in the reduction of the error correlation from 0.9420 to 0.9404. Although the change is marginal, it is more meaningful when put in the light of an improved ensemble. The ensemble members themselves have improved (the classification error of the best classifier is reduced by 4%), and a reduced correlation implies that they produce errors on different patterns. This leads to an improved overall performance. It is worth mentioning that the large margin between the aggregation method and the oracle is due to the biased output of the classifiers when they are trained with the disjoint data. This can be in analogy to the poor performance attained by combining disjoint modular neural networks using averaging.

Table 2 compares the classification error achieved with standard voting and trained methods on an ensemble that is trained with and without sharing on the Satimage data base. Although, the accuracy of the individual classifiers is reduced, the classification error is reduced using most of the aggregation methods. The reduced performance of the members of the ensemble is apparent in the increase of classification error of the best classifier in the ensemble. Except for the Bayesian vote, the classification error is reduced by as much as 6%. With the increase in classification errors of the individual classifiers, we would expect an increase in the error correlation of the ensemble. The error correlation has slightly increased from 0.8836 to 0.8919.

Table 1: 20 Class Gaussian Problem: Comparison of ensemble approaches with and without sharing

Aggregation method	Without Sharing	With Sharing
Majority	14.25 ± 0.51	13.48 ± 0.32
Maximum	14.34 ± 0.99	14.00 ± 0.34
Average	13.88 ± 0.41	13.23 ± 0.09
Nash	13.75 ± 0.43	13.08 ± 0.19
Borda	14.00 ± 0.62	13.06 ± 0.25
Weighted Average	13.46 ± 0.56	12.84 ± 0.17
Bayesian	13.12 ± 0.20	12.66 ± 0.10
Choquet Integral	14.39 ± 0.97	14.12 ± 0.30
Best Classifier	16.20 ± 4.03	15.57 ± 3.27
Oracle	3.74 ± 0.32	3.88 ± 0.31

Table 2: Satimages Data: Comparison of ensemble approaches with and without sharing

Aggregation method	Without Sharing	With Sharing
Majority	13.42 ± 0.88	13.39 ± 0.90
Maximum	14.87 ± 1.90	14.59 ± 1.54
Average	13.31 ± 1.01	13.26 ± 0.93
Nash	17.36 ± 4.15	16.28 ± 4.44
Borda	13.97 ± 1.33	13.90 ± 1.03
Weighted Average	13.17 ± 0.94	13.09 ± 0.91
Bayesian	13.63 ± 0.80	13.76 ± 1.03
Choquet Integral	14.85 ± 2.07	14.57 ± 1.43
Best Classifier	16.52 ± 2.80	17.13 ± 1.03
Oracle	5.08 ± 0.13	5.41 ± 0.23

5 Discussions and Conclusion

In this work we have discussed the concept of sharing training resources in classifier ensembles. The exchange of information between members of the ensemble during training would allow the aggregation module to carry out a more informed fusion process. This leads to more diversity amongst the members of the ensemble. In this work we have presented an algorithm that couples the training of an ensemble. The algorithm presents a method to increase the useful diversities among the classifiers, and improve the over all classification accuracy through sharing of training information. The diversity can be seen in the performance of the individual classifiers, as well as the error correlation within the ensemble. The algorithm is capable of autonomously directing the training of each of the individual modules by selecting the training records used in re-training based on the performance of the ensemble. The introduction of sharing has shown to improve the performance of the ensemble. However, the adaptive algorithm presented, like all iterative algorithms, is more suitable for large data sets. We have demonstrated that our method can

produce results that are as good as or better than independently trained members of the ensemble over a number of aggregation methods.

References

[1] A. Sharkey, "Multi-net systems", *Combining artificial neural nets*, A. Sharkey (Ed.), Springer-Verlag, London, pp. 1-30, 1999.

[2] A. Sharkey, "Types of multinet systems", *Multiple Classifier Systems, 3rd international workshop, MCS2002*, F. Roli and J. Kittler (Eds.), Cagliari, Italy, 24-26 pp. 108-117, June, 2002, Springer-Verlag, Berlin, LNCS2364.

[3] L. Kuncheva, M. Skurichina, and R. Duin, "An experimental study on diversity for bagging and boosting with linear classifiers", *Information Fusion* 3(4), pp. 245-258, 2002.

[4] R. Dara, "Cooperative multi-classifiers", Ph.D. Thesis Proposal, University of Waterloo, Ontario, Canada, June 2003.

[5] L. Breiman, "Bagging predictors", *Machine Learning* 26(2), pp. 123-140, 1996.

[6] L. Breiman, "Random forests", *Machine Learning* 45(1), pp. 5-32, 2001.

[7] T. Dietterich, "An experimental comparision of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization", *Machine Learning* 40(2), pp. 139-158, 2000.

[8] T. Dietterich, "Ensemble learning", *The handbook of brain theory and neural networks*, M. Arbib (Ed.), MIT Press, Cambridge, 2002.

[9] R. Schapire, "The strength of weak learnability", *Machine Learning* 5, pp. 197-227, 1990.

[10] H. Druker, C. Cortes, L. Jackel, Y. LeCum, and V. Vaprik, "Boosting and other ensemble methods", *Neural Computation* 6, pp. 1289-1301, 1994.

[11] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/variance dilemma", *Neural Computation* 4(1), pp. 1-58, 1992.

[12] Y. Freund, and R. Schapire, "Experiments with a new boosting algorithm", *Proceedings of the 13th international conference on machine learning*, San Fransiciso, CA, USA, pp. 148-146, 1996, Morgan Kaufmann, New York, 1996.

[13] T. Hastie, and R. Tibshirani, "*Generalized additive models*", Chapman and Hall, London, 1990.

[14] L. Breiman, "Combining predictors", *Combining artificial neural nets*, A. Sharkey (Ed.), Springer-Verlag, London, 1999.

[15] Y. Liu, and X. Yao, "A cooperative ensemble learning system", *Proceedings of the international joint conference on neural networks (IJCNN'98)*, Anchorage, AK, USA, 4-9, pp.2202-2207, May 1998.

[16] G. Auda, and M. Kamel, "EVOL, Ensemble Voting OnLine", *International conference on neural networks (ICNN'98)*, Vol. 2, pp. 1356-1360, 1998.

[17] N. Wanas, L. Hodge, and M. Kamel, "Adaptive training algorithm for an ensemble of networks", *International joint conference on neural networks (IJCNN'01)*, Washington, DC, USA, 15-19, pp. 2590-2595, July 2001.

[18] M. Kamel and N. Wanas, "Data dependence in combining classifiers", *Multiple Classifier Systems, 4th international workshop, MCS2003*, T. Windeatt and F. Roli (Eds.), Guilford, UK, 11-13, pp. 1-14, June 2003, Springer-Verlag, Berlin, LNCS2709.

[19] L. Hodge, G. Auda, and M. Kamel,"Learning decision fusion in cooperative modular neural networks,", *Proceedings of the international joint conference on neural networks (IJCNN'99)*, Washington, DC, USA, 10-16, pp. , July, 1999.

[20] N. Wanas and M. Kamel, "Weighted combination of neural network ensembles", *International joint conference on neural networks (IJCNN'02)*, Honolulu, HI, USA, 12-17 Vol. 2, pp. 1748-1752, May 2002.

[21] G. Auda, and M. Kamel, "CMNN: Cooperative modular neural networks for pattern recognition", Pattern Recognition Letters Vol. 18:11-13, pp. 1391-1398, 1997.

[22] C. Blake and C. Merz, "UCI repository of machine learning databases [http://www.ics.uci.edu/∼mlearn/MLRepository.html]", 1998.

[23] S. Hashem, "Optimal linear combinations of neural networks", *Neural Networks* Vol. 10:4, pp. 599-614, 1997.