Negative Correlation Learning and the Ambiguity Family of Ensemble Methods

Gavin Brown and Jeremy Wyatt

School of Computer Science, University of Birmingham
Edgbaston Park Road, Birmingham, B15 2TT
{g.brown,j.l.wyatt}@cs.bham.ac.uk
http://www.cs.bham.ac.uk/~gxb

Abstract. We study the formal basis behind Negative Correlation (NC) Learning, an ensemble technique developed in the evolutionary computation literature. We show that by removing an assumption made in the original work, NC can be shown to be a derivative technique of the Ambiguity decomposition by Krogh and Vedelsby. From this formalisation, we calculate parameter bounds, and show significant improvements in empirical tests. We hypothesize that the reason for its success lies in rescaling an estimate of ensemble covariance; then show that during this rescaling, NC varies smoothly between a single neural network and an ensemble system. Finally we unify several other works in the literature, all of which have exploited the Ambiguity decomposition in some way, and term them the *Ambiguity Family*.

1 Introduction

Error 'diversity' is now widely recognised as a desirable characteristic in multiple classifier systems. Though still an ill-defined concept, it is related to statistical correlation, and a number of methods designed to encourage low correlation between classifiers have matured over the last decade. Our framework for this investigation hinges on regarding these methods as dichotomous: *explicit* and *implicit* diversity methods. Explicit methods measure diversity (correlation) in some manner and directly incorporate this knowledge into the construction or combination of the estimators; for example Input Decimation Ensembles [10], which measure correlation between features before assigning them to particular networks. Implicit methods utilise purely stochastic perturbations to encourage diversity; for example, Bagging or similar data resampling techniques. In this paper we are concerned with explicit methods, in particular those which share a common root in the Ambiguity decomposition from [4], widely recognised as one of the most important theoretical results obtained for ensemble learning. It states that *the mean-square error of the ensemble estimator is guaranteed to be less than or equal to the average mean-square error of the component estimators*; the details of this will be expanded upon later.

T. Windeatt and F. Roli (Eds.): MCS 2003, LNCS 2709, pp. 266–275, 2003.

1.1 Negative Correlation Learning

After this initial branching, both explicit and implicit methods can be further divided as manipulating either: the initial weights of the networks, the network architectures, the training data, or the learning algorithm. Some authors, taking the latter approach, have found benefit from using a regularisation term in the learning. Negative Correlation[1] (NC) Learning [5], an extension of Rosen's decorrelated networks [11], is an ensemble learning technique which incorporates such a regularisation term into the backpropagation error function. The regularisation term is meant to quantify the amount of error correlation, so it can be minimised explicitly during training—as such, it is an *explicit* diversity method. In NC the error ϵ_i of network i is:

$$\epsilon_i = \frac{1}{2}(f_i - d)^2 + \lambda p_i \tag{1}$$

where f_i is the output of the i^{th} network on a single input pattern, d is the target, and λ is a weighting parameter on the penalty function p_i. Strictly, this notation should include input, so $f_i(n)$ and $d(n)$ for the n^{th} input pattern, but we omit this for notational simplicity. The λ parameter controls a trade-off between objective and penalty functions; when $\lambda = 0$, the penalty function is removed and we have an ensemble with each network training independently of the others, using plain backpropagation. NC has a penalty function of the form:

$$p_i = (f_i - \bar{f}) \sum_{j \neq i} (f_j - \bar{f}) \tag{2}$$

where $\bar{f}$ is the average output of the whole ensemble of M networks at the previous timestep, defined as $\bar{f} = \frac{1}{M} \sum_{i=1}^{i=M} f_i$. NC has seen a number of empirical successes [5,6,7], consistently *outperforming* a simple ensemble system, but so far has had very little formal analysis to explain why it works when it does; this leads naturally to our first question.

1.2 Why Does the Algorithm Work?

The mean-square error (MSE) of an ensemble system can be decomposed into bias, variance and covariance components [12]. The strength parameter λ in NC provides a way of controlling the trade-off between these three components: a higher value encourages a decrease in covariance, as has been demonstrated empirically [5]. However we do not yet have a clear picture of the exact dynamics of the algorithm.

When $\lambda = 1$, we have a special situation. This was described by Liu [5] to show a theoretical justification for NC-Learning. It should be noted that, in the calculation of the derivative, Liu has: *"... made use of the assumption that the output of the ensemble $\bar{f}$ has constant value with respect to f_i"* [5, p.29].

[1] So-called because it has demonstrated on a number of occasions that it is able to generate estimators with *negatively correlated* errors.

We have:

$$\epsilon_i = \frac{1}{2}(f_i - d)^2 + \lambda(f_i - \bar{f})\sum_{j \neq i}(f_j - \bar{f})$$

$$\begin{aligned}\frac{\partial \epsilon_i}{\partial f_i} &= f_i - d + \lambda \sum_{j \neq i}(f_j - \bar{f}) \\ &= f_i - d - \lambda(f_i - \bar{f}) \\ &= \bar{f} - d\end{aligned}$$

However, although the assumption of constant $\bar{f}$ is used, so is the property that $\sum_{j \neq i}(f_j - \bar{f}) = -(f_i - \bar{f})$, the sum of deviations around a mean is equal to zero; obviously the sum of deviations around a constant does not have this property. Using this apparently contradictory assumption, and the fact that the overall ensemble error function is defined as $\epsilon = \frac{1}{2}(\bar{f} - d)^2$, it was stated:

$$\frac{\partial \epsilon}{\partial f_i} = \frac{1}{M}\left[\frac{\partial \epsilon_i}{\partial f_i}\right] \tag{3}$$

showing that the gradient of the individual network error is directly proportional to the gradient of the ensemble error. Though this is obviously a useful property, the justification for the assumption is unclear. To understand this further, as with all algorithms, it would be useful to first understand a framework into which NC can fit. What is the theoretical grounding of NC? What are other similar algorithms? In the following sections we address these questions.

2 Formalising NC-Learning

In this section we show how NC can be related to the work by Krogh and Vedelsby [4], which showed the ensemble error could be broken down into two terms, one of which is dependent on the correlations between network errors.

2.1 NC Uses the Ambiguity Decomposition

Note that the penalty function is actually a sum of pairwise correlations; if we remember again that the MSE of an ensemble decomposes into bias plus variance plus covariance [12], then including some measure of correlation to be minimised seems like an intuitive thing to do (first noted by Rosen [11]). However this intuition is not enough. We note that the penalty function can be rearranged to:

$$p_i = -(f_i - \bar{f})^2 \tag{4}$$

which is again due to the property that the sum of deviations around a mean is equal to zero. This rearrangement is only possible if we remove Liu's assumption ([5], p29) of constant $\bar{f}$. As can be seen, each network minimises its penalty function by moving its output away from the ensemble output, the mean response

of all the other networks. So why should increasing distance from the mean, or optimising equation (1), necessarily lead to a decrease in ensemble error? An examination of the proof by Krogh and Vedelsby can answer this question, and also raise some new questions on the setting for the λ parameter. Their work showed that the following statement about ensemble error was true:

$$(\bar{f} - d)^2 = \sum_i w_i (f_i - d)^2 - \sum_i w_i (f_i - \bar{f})^2 \tag{5}$$

This stems from a number of definitions, one of which is the *ambiguity* of a single member of the ensemble:

$$v_i = (f_i - \bar{f})^2 \tag{6}$$

Remembering that the individual networks in NC-learning minimise the penalty function, and looking at equations (4) and (6) we can see $p_i = -v_i$ and so the networks are in fact maximising this ambiguity term, equation (6). This in turn of course affects the total ensemble error. Please note this result for NC only holds if the weightings on the networks are all $\frac{1}{M}$, as equation (2) cannot be rearranged to (4) without this constraint.

To understand this further we take equation (5), multiply through by $\frac{1}{2}$ and rearrange slightly assuming our ensemble is uniformly weighted, we then have:

$$\frac{1}{2}(\bar{f} - d)^2 = \frac{1}{M} \sum_i \left[\frac{1}{2}(f_i - d)^2 - \frac{1}{2}(f_i - \bar{f})^2 \right] \tag{7}$$

We see that the mean squared error of an ensemble can be decomposed into a weighted summation, where the ith term is the backpropagation error function plus the NC-Learning penalty function.

Now, since we have removed the constraint of assuming constant $\bar{f}$ to allow a link to the ambiguity decomposition, it seems more rigorous to differentiate the network error again without this assumption. What happens in this case? We have a partial derivative:

$$\frac{\partial \epsilon_i}{\partial f_i} = f_i - d - \lambda \Big[2 \frac{M-1}{M} (f_i - \bar{f}) \Big] \tag{8}$$

where M is the number of networks in the ensemble. Keeping the assumption of constant $\bar{f}$ causes this term $2\frac{M-1}{M}$ to disappear. However, it does seem sensible to retain this, as it takes account of the number of networks. In all of Liu's experiments [5,6,7], the λ parameter was thought to be problem dependent. Now we understand that it has a deterministic component, this $2\frac{M-1}{M}$. To avoid confusion, from this point on, we shall refer to the λ parameter in the following context, where γ is still a problem-dependent scaling parameter:

$$\lambda = \gamma \left[2 \frac{M-1}{M} \right] \tag{9}$$

In understanding the role of the strength parameter a natural question to ask is, what are the bounds?

2.2 What Are the Bounds of λ and γ?

As with any problem-dependent parameter, we would like to know bounds, to allow us to set it sensibly. Liu stated that the bounds of λ should be $[0, 1]$, based on the following calculation:

$$\begin{aligned}\frac{\partial \epsilon_i}{\partial f_i} &= f_i - d + \lambda \sum_{j \neq i} (f_j - \bar{f}) \\ &= f_i - d - \lambda(f_i - \bar{f}) \\ &= (1 - \lambda)(f_i - d) + \lambda(\bar{f} - d)\end{aligned}$$

He states: *"the value of parameter λ lies inside the range $0 \leq \lambda \leq 1$ so that both $(1-\lambda)$ and λ have non-negative values"* ([5], p29). However this justification is questionable, and again here we see the assumption of constant $\bar{f}$ is violated.

We have to ask therefore, why would it be a problem if $(1 - \lambda)$ and λ were negative values? Maybe the bounds of λ should not be $[0, 1]$. How can we determine what the true bounds should be? We can simply take the second partial derivative of ϵ_i with respect to f_i:

$$\frac{\partial^2 \epsilon_i}{\partial f_i^2} = 1 - \lambda(1 - \frac{1}{M})$$

If the second derivative becomes negative, then our function contains only local maxima or points of inflexion, and we have lost any useful gradient information from our original objective function. Rearranging this, to maintain a positive second derivative, we have an upper bound for λ and also γ:

$$\lambda_{upper} = \frac{M}{M-1} \qquad\qquad \gamma_{upper} = \frac{M^2}{2(M-1)^2}$$

Figure 1 plots λ_{upper} and the equivalent γ_{upper} for different numbers of networks. We see that in the infinite networks case, λ_{upper} converges to 1, and γ_{upper} converges to 0.5. It should be noted that with a smaller number of networks λ_{upper} is greater than 1.

2.3 An Empirical Study

With our new understanding of the bounds of the parameter in NC, we now perform an empirical evaluation, and show that it is critical to consider values for the strength parameter *outside* the originally specified range.

Table 1 shows the classification error rates and standard errors of two empirical tests, on the Wisconsin breast cancer data from the UCI repository (699 patterns), and the heart disease Statlog dataset (270 patterns). An ensemble consisting of two networks, each with five hidden nodes, was trained for 2000 iterations with and without NC. We use 5-fold cross-validation and 40 trials from uniform random weights in $[-0.5, 0.5]$ for each setup; in total 200 trials were conducted for each experimental configuration. It should be noted that with 2

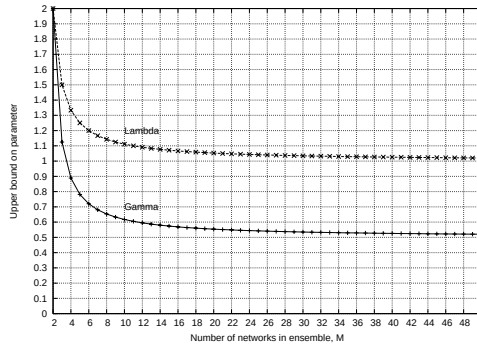

Fig. 1. The Upper bound on γ and λ

Table 1. Mean classification error rates (200 trials) using NC on two UCI datasets

	$\gamma = 0$	$\gamma = 0.5$	$\gamma = 1$
Breast cancer	0.0408 (0.0006)	0.0410 (0.0007)	0.0383 (0.0009)
Heart disease	0.2022 (0.0028)	0.1995 (0.0027)	**0.1802** (0.0020)

networks, $\gamma = \lambda$. The γ values tested are those considered in the original work on NC: 0.0, 0.5 and 1.0. When γ was set appropriately, results on the heart data showed NC significantly better than a simple ensemble (equivalent to $\gamma = 0$) at $\alpha = 0.05$ on a two-tailed t-test. On the breast cancer data, although the mean was lower, it was not statistically significant.

Figure 2 shows the results of repeating our experiment, but illustrating the full range of the strength parameter. Mean error rate over the 200 trials is plotted, and 95% confidence intervals shown. We see that performance on the breast cancer data *can be improved significantly* by considering the upper bounds beyond those previously specified; on the heart disease data (not shown due to space considerations), stable performance was observed beyond $\gamma = 1$.

As a further measure of comparison, we calculated the percentage reduction in the mean error rate, in relation to when $\gamma = 0$, equivalent to a simple backpropagation ensemble. On the breast cancer data, using $\gamma = 1$ gave a 6% reduction, but using the optimum value at $\gamma = 1.7$ gave a 21% reduction.

We have shown a significant performance improvement by reconsidering the bounds of the strength parameters. It should be noted that, even though the theoretical upper bound is known, in practise it seems error can rise rapidly long before this bound is reached. On the breast cancer data, error became uncontrollable beyond $\gamma = 1.8$, and on the heart disease data at $\gamma = 1.45$; it remains to be seen if it is possible to empirically characterise when this rapid increase will occur.

We know from figure 1 that the upper bound reduces as we add more networks; from this it is reasonable to assume that the optimal value would follow a similar trend. But why? What role does γ play?

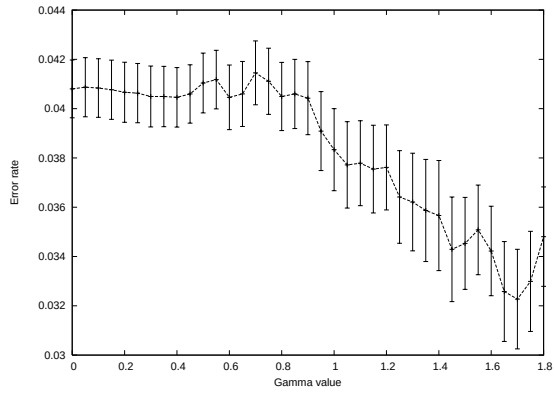

Fig. 2. Breast cancer dataset results

3 What Is the Relation of NC to Bias and Variance?

In this section we ask the question, how does NC relate to the bias-variance decomposition? Can this tell us what role the strength parameter plays?

The second term on the right handside of equation 5 is the *ensemble ambiguity*; this is maximised when training an ensemble with NC. When $w_i = \frac{1}{M}$ for all i, it can be shown:

$$-E\Big\{\sum_i w_i(f_i - \bar{f})^2\Big\} = E\Big\{\frac{1}{M}\sum_i (f_i - \bar{f})\sum_{j\neq i}(f_j - \bar{f})\Big\}$$
$$= \frac{1}{M}\sum_i \sum_{j\neq i} E\Big\{(f_i - \bar{f})(f_j - \bar{f})\Big\}$$

The expected value of the ensemble ambiguity term is an approximation to the average covariance of the ensemble members. It is an approximation because with a finite number of networks, $\bar{f} \neq E\{f\}$, and also because the sum is multiplied by $\frac{1}{M}$ instead of $\frac{1}{M(M-1)}$. We can see that when we are increasing ambiguity, we are reducing this covariance term. When training an ensemble with NC, we use the γ parameter, directly attempting to reduce covariance by over-emphasising this component. A larger γ parameter will be needed when our approximation is not very good: this is will most likely occur when we have a small number of networks, but it could also be due to noise in the training data. It is hoped that with further analysis we will be able to mathematically characterise this, and provide further guidelines for setting the strength parameter.

4 Viewing the Ensemble as a Single Estimator

In this section we briefly show how NC-Learning works on a search landscape that, using the λ parameter, can be smoothly scaled between that of a fully par-

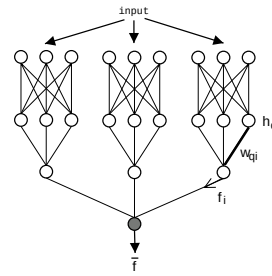

Fig. 3. A typical ensemble architecture

allel ensemble system, and a single large neural network. Regard the architecture in figure 3.

This is an ensemble of three networks, each with three hidden nodes, using a simple average combination rule for the ensemble output. We desire to update the weight, w_{qi}, marked in bold—this is one of the output layer weights for the ith network (connected to the qth hidden node). If we use NC-Learning, we need the derivative of the error ϵ_i with respect to w_{qi}. If $\lambda = 0$, we have:

$$\frac{\partial \epsilon_i}{\partial w_{qi}} = \Big[(f_i - d)\Big] \cdot \Big[f_i(1 - f_i)\Big] \cdot \Big[h_q\Big] \tag{10}$$

And if $\lambda = 1$, we have:

$$\frac{\partial \epsilon_i}{\partial w_{qi}} = \Big[(\bar{f} - d)\Big] \cdot \Big[f_i(1 - f_i)\Big] \cdot \Big[h_q\Big] \tag{11}$$

If we now consider the ensemble architecture as one large network (with fixed output layer weights), then our output node is marked in dark gray, and has a linear output function $\bar{f}$:

$$\bar{f} = a_i \tag{12}$$

and its activation function a_i:

$$a_i = \frac{1}{M} \sum_i f_i \tag{13}$$

The error of this large network on a single pattern is:

$$\epsilon = \frac{1}{2}(\bar{f} - d)^2 \tag{14}$$

Now, as before, we find the derivative of ϵ with respect to the weight w_{qi}:

$$\frac{\partial \epsilon}{\partial w_{qi}} = \frac{\partial \epsilon}{\partial \bar{f}} \frac{\partial \bar{f}}{\partial a_i} \frac{\partial a_i}{\partial f_i} \frac{\partial f_i}{\partial a_i} \frac{\partial a_i}{\partial w_{qi}} \tag{15}$$

$$\frac{\partial \epsilon}{\partial w_q} = \Big[(\bar{f} - d))\Big] \cdot \Big[1\Big] \cdot \Big[\frac{1}{M}\Big] \cdot \Big[f_i(1 - f_i)\Big] \cdot \Big[h_q\Big] \tag{16}$$

The only difference from equation (11) is the $\frac{1}{M}$. All the minima are in the same locations, but the landscape is M times shallower—the effect of which could be duplicated with a smaller learning rate in the update rule. When we change λ, we can scale smoothly between a single network with a linear output function, and a parallel ensemble system.

5 Related Work: The Ambiguity Family

In this section we briefly review some other techniques which have exploited the ambiguity decomposition in some way, either to create or combine a set of predictors. In the last few years, the ambiguity decomposition has quietly been utilised in almost every aspect of ensemble construction. Krogh and Vedelsby themselves developed an active learning scheme [4], based on the method of query by committee, selecting patterns to train on that had a large ambiguity; this showed significant improvements over passive learning in approximating a square wave function.

[8] selected feature subsets for the ensemble members to train on, using a genetic algorithm with an ambiguity-based fitness function; this showed gains over Bagging and Adaboost on several classification datasets from the UCI repository. A precursor to this work was Opitz and Shavlik's Addemup algorithm [9], which used the same fitness function to optimise the network topologies composing the ensemble. Interestingly, both these GA-based approaches also used a strength parameter, λ, to vary the emphasis on diversity. The difference between their work and NC is that NC incorporates ambiguity into the backpropagation weight updates, while Addemup trains with standard backpropagation, then selects networks with a good error diversity.

The original ambiguity paper [4] also used an estimate of ambiguity to optimise the ensemble combination weights, showing in some cases it is optimal to set a network weight to zero—essentially removing it from the ensemble. In [1] bootstrap resamples of training data are used to estimate ambiguity, in order to approximate the optimal training time; this minimises the overall ensemble generalisation error.

We can see that ambiguity has been utilised in many ways: pattern selection [4], feature selection [8], optimising the topologies [9] of networks in the ensemble, optimising the combination function [4], and also optimising training time [1]. NC fits neatly into the gap as the first technique to directly use ambiguity for network weight updates.

6 Conclusions

We analyzed an ensemble technique, Negative Correlation Learning[5], that extended from Rosen[11], and developed in the evolutionary computation literature. We show a link to the bias-variance decomposition, and hypothesise that NC succeeds by rescaling an estimate of the ensemble covariance. This formalisation of NC is a step towards placing it in a solid statistical framework. In

showing how NC uses its strength parameter to scale smoothly between an ensemble system and a single network, it serves to partially unify the concepts of training an ensemble and training a single estimator.

In addition this work highlights the need for collaboration between communities, as a technique grown in the artificial intelligence and evolutionary computation community can be of interest to the pattern recognition community. Several other works on artificial speciation[3] and multi-objective evolutionary algorithms[2] are highly relevant, and are slowly formulating a solid statistical grounding, and it is hoped future cross-disciplinary links can be fostered.

References

1. J. Carney and P. Cunningham. Tuning diversity in bagged neural network ensembles. Technical Report TCD-CS-1999-44, Trinity College Dublin, 1999.
2. Kalyanmoy Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation*, 7(3):205–230, 1999.
3. Vineet Khare and Xin Yao. Artificial speciation of neural network ensembles. In J.A.Bullinaria, editor, *Proc. of the 2002 UK Workshop on Computational Intelligence (UKCI'02)*, pages 96–103. University of Birmingham, UK, September 2002.
4. Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. *NIPS*, 7:231–238, 1995.
5. Yong Liu. *Negative Correlation Learning and Evolutionary Neural Network Ensembles*. PhD thesis, University College, The University of New South Wales, Australian Defence Force Academy, Canberra, Australia, 1998.
6. Yong Liu and Xin Yao. Negatively correlated neural networks can produce best ensembles. *Australian Journal of Intelligent Information Processing Systems*, 4(3/4):176–185, 1997.
7. Yong Liu and Xin Yao. Ensemble learning via negative correlation. *Neural Networks*, 12(10):1399–1404, 1999.
8. David Opitz. Feature selection for ensembles. In *Proceedings of 16th National Conference on Artificial Intelligence (AAAI)*, pages 379–384, 1999.
9. David W. Opitz and Jude W. Shavlik. Generating accurate and diverse members of a neural-network ensemble. *NIPS*, 8:535–541, 1996.
10. Nikunj C. Oza and Kagan Tumer. Input decimation ensembles: Decorrelation through dimensionality reduction. *LNCS*, 2096:238–247, 2001.
11. Bruce E. Rosen. Ensemble learning using decorrelated neural networks. *Connection Science - Special Issue on Combining Artificial Neural Networks: Ensemble Approaches*, 8(3 and 4):373–384, 1996.
12. N. Ueda and R. Nakano. Generalization error of ensemble estimators. In *Proceedings of International Conference on Neural Networks*, pages 90–95, 1996.