ESANN'1998 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 22-23-24 April 1998, D-Facto public., ISBN 2-9600049-8-1, pp. 261-266

Weight Saliency Regularisation in Augmented Networks

Peter J. Edwards * and Alan F. Murray

Dept. of Electrical Engineering
Edinburgh University, UK

Abstract. This paper introduces the concept of "optimally distributed computation" in feed-forward neural networks via regularisation of weight saliency. By constraining the relative importance of the parameters, computation can be distributed thinly and evenly throughout the network. We propose that this will have beneficial effects on fault tolerance performance and generalisation ability in augmented network architectures. These theoretical predictions are verified by simulation experiments on two problems — one artificial and the other a "real world" task. In summary, this paper presents regularisation terms for distributing neural computation optimally.

1. Introduction

Optimally distributed computation leads to fault tolerance and follows the spirit of Occam's razor even in augmented networks to yield good generalisation ability. In neural network models computation is more or less distributed in the weighted connections. However, the "distributedness" is in no way optimal and poor performance, in terms of fault tolerance, is not unusual. We attest that poor generalisation ability can arise in a similar manner. Fault tolerance is optimised if the computation is spread thinly and evenly throughout the network [6]. Generalisation ability requires that the computational complexity of a network be matched with that required by the task in hand. These are two of the key questions in neural computation that remain open, despite advances in both theoretical and empirical methods. In addition, some complex real-world tasks will require custom designed hardware. In that context, fault-tolerance and optimal utilisation of resources are key to a robust and usable solution. This paper presents regularisation terms for distributing neural computation optimally.

In this paper, along with Neal [7], we assume that regularising an augmented network, i.e. an architecture enlarged to increase network capacity, is preferable over attempting to optimise the architecture to be minimal. Using an augmented network therefore, we are interested in regularising the complexity or weight saliency, or equivalently distributing computation optimally. Distributed computation is an area of neural network research in which there has been a surprising lack of formal study.

This work is supported by EPSRC grant number GR/J44539.
*Email address : pje@ee.ed.ac.uk

Minai, however, has investigated the subject, providing a theoretical basis on which we build. He notes that a network with an optimally distributed representation is also likely to be optimally robust in that each component will have minimal and uniform relevance. In addition he states that good generalisation ability can be induced via distributive constraints and that this is an alternative to the more common structural minimality[5]. Related to these ideas are concepts underlying weight-decay, used widely as a simple regularisation scheme, and the post-training augmentation approach of Emmerson, used to enhance fault tolerance. This second, perhaps less familiar algorithm replicates hidden–output weights, scaling them by the replication factor to distribute computation over a larger network[4]. Underlying both of these algorithms is the premise that weight-saliency, or relevance in the language of Minai, can be approximated by weight-magnitude. In another paper we show how this is an inaccurate approximation and one that performs poorly in practice[3]. In this paper we compare the weight-decay technique with two regularisation terms that use a much more accurate measure of weight saliency. Using two application problems, one artificial and the other a real-world task, we show how optimally distributed computation leads to both fault tolerance and good generalisation ability in augmented networks.

2. Smoothing the Solution Locus

Distributed computation can be achieved by regularising weight-saliency. Since the error Hessian measures weight saliency, its regularisation will achieve distributed computation by smoothing the curvature of the solution locus. In addition by viewing the smoothness of the solution locus as a route to distributed computation, we can introduce smoothness constraints that are amenable to implementation, to distribute the computation optimally. In this section, therefore, we build on our previous work with weight-noise [6] and show how it can be viewed as a smoothness constraint, or regularisation term. We then propose two deterministic approaches to smoothing, one equivalent to using weight-noise under ideal conditions and the other a more statistically feasible solution.

In our previous work with weight-noise we showed that a distortion in the weights leads to the network consistently finding a different solution. The learning algorithm is normally governed by a guiding cost, or error function, however in this case the weight-noise mediates the learning, in effect including additional penalty terms. By modelling the noise using a Taylor expansion and averaging in time we have shown, [6], that for the particular case of least squares training, the error function becomes :-

$$< \epsilon \left(\underline{w}, \underline{\xi}\right) >=< \epsilon \left(\underline{w}\right) > + \frac{\xi^2}{2} \sum_{A,K,P} \left(\frac{\partial y_{kp}}{\partial w_a}\right)^2 , \tag{1}$$

where $< \epsilon \left(\underline{w}, \underline{\xi}\right) >$ is the time averaged error for a network whose weight vector, $\underline{w}$, is perturbed by some noise vector, $\underline{\xi}$ and y_{kp} is number k of K components of the network output vector $\underline{y}$ for pattern p of the data. This equation shows that training with weight-noise in effect changes the error function to include this extra penalty term. This term has been discussed in detail in our earlier work [6] and can be thought of as a mechanism for obtaining distributed computation. It is important to note at this stage that the weight-noise analysed above could equally have been input-noise and a similar error function produced [1]. The extra term in the input-noise error function is one of a series of terms that are known as Tikhonov regularisers. Such terms are commonly incorporated directly into the learning to smooth the network function, i.e. :-

$$\tilde{\epsilon} = \epsilon + \lambda \sum_{I,K,P} \left(\frac{\partial^m y_{kp}}{\partial {x_i}^m} \right)^2 \tag{2}$$

where the nominal error function is modified by an additional term which is of order m, where x_i is number i of I components of the input vector $\underline{x}$ and λ controls the influence of the regularisation term in the overall error reduction. Comparing our noise-induced penalty term in (1) with this generalised Tikhonov regulariser it is clear that (1) is also a regularisation term (where $m = 1$) although of the weight–output relationship, rather than the input–output relationship. It is therefore a smoother of the solution locus as opposed to network function. In addition statistical theory would suggest that for a Tikhonov regulariser, $m = 2$ provides the lowest order at which any visual smoothness will be seen [9]. In this paper we consider two penalty terms for fault tolerance which are smoothness regularisers of this kind, although perhaps smoothers of the solution locus rather than network function. They are of the form :-

$$\Omega_{\mathrm{FT}} = \sum_{A,K,P} \left(\frac{\partial^m y_{kp}}{\partial {w_a}^m} \right)^2 \tag{3}$$

and are of order $m = 1$ and $m = 2$. We shall subsequently refer to these roughness penalties as M1 and M2 respectively. In the following sections two application problems, one artificial and the other a real-world task, are incorporated into simulation experiments using MLPs trained with these roughness penalties. In addition we train with weight-decay and compare the results.

3. Simulation Experiments

The experiments described here assess the regularisation constraints defined above and are based on two data sets. The first problem is an artificial, 2-class classification task in two dimensions. Samples for both classes are taken from normal distributions with means of 0.2 and 0.3, where both distributions are circular with variance 0.05 and 0.15 respectively. In these experiments 200 patterns are used as the training pattern set and a further 9800 for testing. Initially, the network architecture had an $I{:}J{:}K$ structure of 2:4:2, i.e. 4 sigmoidal hidden units and the outputs were encoded to classify the two classes as 1-out-of-N. This task requires the network to approximate the Bayesian optimal decision boundary which is a circle in this case. This is a difficult task especially for MLP networks with sigmoidal neurons (perhaps inappropriately) used here. The initial 4 hidden units were expanded to 16 to draw out the effects within augmented networks.

ESANN'1998 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 22-23-24 April 1998, D-Facto public., ISBN 2-9600049-8-1, pp. 261-266

The second test problem, link admission control, is a real task, using real data. Link admission control is a function of an ATM (Asynchronous Transfer Mode) communications network router. The task is to decide whether a particular link in the network can accept a new link. An estimate of the probability of losing a data packet is used to make a decision. This estimate is based on the current load present on the link and traffic parameters characterising the new connection. The neural solution has been developed at Uppsala University in Sweden [8]. Here 9 input fields are used to classify a link to be admissible or not based on the probability of loss. We use a 1-out-of-N encoding of the two output classes and, initially, 6 hidden units in an MLP network with sigmoidal neurons. The hidden layer was later increased to 16 units for experiments with augmented networks. The data includes 500 training vectors and an additional 4000 test vectors. For the purposes of training a simple steepest descent technique with incorporated line search was used and was applied until a minimum error was reached in a validation set of 2000 patterns for the 2-class problem and 1000 patterns for the link admission task. For the purpose of the experiments carried out for this paper the mean square error (MSE) was used as our cost function error making the (generally naive) assumption that the MSE minimum will be equivalent to the classification minimum.

4. Results

The experiments carried out here compare the two new regularisation terms derived above with weight-decay and a control case with no regularisation. In addition the results of further experiments with augmented networks show the effects of increasing network capacity combined with and without the different forms of regularisation. In the experiments 30 networks were trained for each case of a range of regularisation coefficient (λ) values. The results below show the minimum average generalisation error achieved and fault tolerance using the mean error Hessian (MEH) measure :-

$$\text{MEH} = \frac{1}{PK} \sum_{A,K,P} \frac{\partial^2 \epsilon\,(\underline{w})}{\partial w_a{}^2} \tag{4}$$

The MEH measure is the average curvature of the solution locus which is inversely proportional to fault tolerance for a realistic small perturbation fault model [2]. During these experiments the λ coefficient was optimised using the validation data.

Considering first the results of the experiments carried out with the artificial 2-class problem shown in Table 1. For the case of the number of hidden units J=4, it is clear that there is a dramatic improvement in the fault tolerance over the control case, for networks trained with the M1 and M2 regularisers as well as some improvement in generalisation ability. The weight-decay case also exhibits moderate generalisation improvement but at the expense of degradation in fault tolerance. This result indicates that weight-magnitude is not equivalent to weight-saliency and that the more accurate measure of weight-saliency implicit to the M1 and M2 regularisers is necessary for fault tolerance enhancement. In addition, the augmentation of the network prior to training (J=16) has a beneficial effect on generalisation performance especially when

Table 1: Comparison of network ensembles trained using weight decay, M1 and M2 penalties, and a control experiment with no regularisation. The task used was the artificial 2-class problem with four and sixteen hidden neurons.

	J	Control	Weight Decay	M1	M2
% Error	4	12.51	12.44	12.42	12.43
MEH	4	0.179	0.192	0.050	0.089
% Error	16	12.39	12.36	12.32	12.27
MEH	16	0.257	0.276	0.115	0.042

mediated by the regularisation constraints. This is particularly evident in the M2 case where there is an additional fault tolerance improvement. As the optimal boundary for the artificial task is circular, it is to be expected that more sigmoidal hidden units will be beneficial for this task. It is encouraging however that this extra capacity can be controlled by the regularisation schemes.

For the link admission task the results are shown in Table 2. The case of J=6 shows that there is improvement in fault tolerance and generalisation ability due to the M1 and M2 regularisers over the control case. Again the weight-decay case gives some generalisation improvement but deterioration in fault tolerance performance. In terms of the augmentation experiment for this task the extra nodes (i.e. in terms of J) do not lead to enhanced performance. Comparing the results for the control experiments in Table 2, the extra network complexity has a negative effect on both performance metrics. This is perhaps the result of some over-fitting escaping the validation process. We can conclude that six hidden units are sufficient for this task. It is encouraging to note, however, that the regularisers of weight saliency provide control over this extra network complexity and stop over-fitting to a large extent.

The results presented here show that the M1 and M2 regularisation terms can be used to achieve fault tolerance and enhanced generalisation ability. In addition weight-saliency regularisation can be used to distribute information optimally in an augmented network for potentially further fault tolerance enhancement and to optimise its functional capabilities. The need for structural, or architectural, minimality and hence the importance of choosing the number of hidden units is diminished as there is benefit from the use of larger networks which exhibit optimally distributed computation.

Table 2: Comparison of network ensembles trained using weight decay, M1 and M2 penalties, and a control experiment with no regularisation. The task used was the link admission problem with six and sixteen hidden neurons.

	J	Control	Weight Decay	M1	M2
% Error	6	3.092	3.054	3.060	3.028
MEH	6	0.0197	0.0276	0.0142	0.0165
% Error	16	3.190	3.048	3.082	3.048
MEH	16	0.0277	0.0341	0.0145	0.0180

ESANN'1998 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 22-23-24 April 1998, D-Facto public., ISBN 2-9600049-8-1, pp. 261-266

5. Conclusions

This paper has introduced optimally distributed computation. By including accurate weight saliency constraints during learning (as penalty terms), computation can be spread thinly and evenly throughout a network. In this way the average curvature of the solution locus is reduced, leading to dramatic improvements in fault tolerance for a realistic "small perturbation" fault model. In addition we have shown that such a constraint can also be used to regularise network complexity even for very large networks, resulting in enhanced generalisation ability.

These results are important if neural networks are to become truly parallel architectures in hardware implementations and to be able to solve complex real-world problems with limited data. Optimally distributed computation allows us to maximise the fault tolerance performance of a given architecture, for realistic hardware faults. In addition the complexity constraints allow us to produce parsimonious models even in augmented network architectures.

References

[1] C. Bishop. Training with noise is equivalent to Tikhonov regularization. *Neural Computation*, 7(1):108–116, 1995.

[2] P.J. Edwards and A.F. Murray. Fault-tolerance via weight-noise in analogue VLSI implementations — a case study with EPSILON. To appear in IEEE Proceedings on Circuits and Systems II : Analog and Digital Signal Processing, 1998.

[3] P.J. Edwards and A.F. Murray. Toward optimally distributed computation. *Neural Computation*, 10:997–1015, 1998.

[4] M.D. Emmerson. *Fault Tolerance and Redundancy in Neural Networks.* PhD thesis, University of Southampton, 1992.

[5] A.A. Minai. *The Robustness of Feed-Forward Neural Networks : A preliminary Investigation.* PhD thesis, University of Virginia, 1992.

[6] A.F. Murray and P.J. Edwards. Synaptic weight noise during MLP training : Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training. *IEEE Trans. Neural Networks*, 5(5):792–802, September 1994.

[7] R.M. Neal. *Bayesian learning for neural networks.* Springer–Verlag, New York, 1996.

[8] E. Nordström, O. Gällmo, L. Asplund, M. Gustafsson, and B. Eriksson. Neural networks for admission control in an ATM network. In L.F. Niklasson and M.B. Bodén, editors, *Connectionism in a Broad Perspective: Selected papers from the Swedish Conference on Connectionism — 1992*, pages 239–250. Ellis Horwood, 1992.

[9] G.A.F. Seber and C.J. Wild. *Nonlinear Regression.* Wiley, NY, 1989.