第10回 回路とシステム
軽井沢ワークショップ
The 10th Karuizawa Workshop on
Circuits and Systems, April 21-22, 1997

Avoiding Weight-illgrowth: Cascade Correlation Algorithm with Local Regularization

Qun XU
Graduate School of Nat. Sci. & Tech.,
Kanazawa University
2-40-20, Kodatsuno, Kanazawa 920, JAPAN
1160xu@ec.t.kanazawa-u.ac.jp

Kenji NAKAYAMA
Dep. of Elec. and Comp. Eng.,
Faculty of Eng., Kanazawa University
2-40-20, Kodatsuno, Kanazawa 920, JAPAN
nakayama@ec.t.kanazawa-u.ac.jp

Abstract

This paper investigates some possible problems of Cascade Correlation algorithm, one of which is the zigzag output mapping caused by weight-illgrowth of the adding hidden unit. Without doubt, it could lead to deteriorate the generalization, especially for regression problems. To solve this problem, we combine Cascade Correlation algorithm with regularization theory. In addition, some new regularization terms are proposed in light of special cascade structure. Simulation has shown that regularization indeed smooth the zigzag output, so that the generialization is improved, especially for functional approximation.

1. Introduction

Although feedforward multilayer Neural Network model has been popularized for a wide variety of classification (pattern recognition) and regression (functional approximation) applications since it was proposed by using the famous supervised learning algorithm: Backpropagation Algorithm[1], there exists one key issue: how to configure a neural network in order to achieve the approximatly optimal performances (efficiency, generalisation, representation ability, scaling, implmentation etc.). It is known that these performances are greatly related to the neural network topology. For instance, "undertraining" will occur if the size of topology is too small to learn all training data, whereas "overtraining" will deteriorate the generalization if the size is so large to learn plenty of noise. If only adopting static learning algorithm (e.g., BP algorithm or its variant QP algorithm [2]), one must determine the optimal Neural Network toplogy prior to training process by the method of trial-and-error. Needless to speak, it isn't expected because it seems like handcraft art rather than scientific design.

Aiming at the above problems, dynamic Neural Networks have been attractive, which incorporate the determination of topologies into the training process, mainly in two ways. One approach, refered to as destructive (pruning) technique, is to start with a large size network (usually one has to guess) and prune it either using sensitivity calculation methods, such as OBD, which are used to prune the topology after training is complete and then retrain repeatedly to achieve the expected performance; or using regularization methods, such as weight decay and weight elimination, which add an extra term to the error function to prune the topology during training to improve the generalization ability[5].

In contrast, another approach, refered to as constructive (growing) technique, is to begin with a small size network (usually with no hidden units) and then gradually add hidden units under the control of some rules to a proper size so that the optimal performance can be achieved[8]. By comparing them, we can find the following facts:

(1) Destructive method still requires an estimate of the relatively large size, and the initial network can take a long time to train. In addition, without the proper control of pruning, unhealthy phenomenon, such as "underpruning" or "overpruning", could appear. But the process of "pruning" is suitable for various kinds of topology, and thus is more flexible.

(2) Constructive method frees one from "guessing" a large enough initial NN configuration, and requires less computations than destructive method since it uses a small network most of time. However, it must be emphasized that the resulted topology generated by constructive method usually has special architecture, such as cascade and upstart architecture, since the same

connection strategy is throughout the whole procedure of growing, so that it has limitations when applying to practical problems.' Similarly, without proper control of growing, it could lead to "overgrowth", which is easy to generate oversized topology, or "undergrowth", which is easy to generate too small network. In both cases, the generalization will be decreased.

Whereby, we can't assert rashly which is better. In fact, combining these two methods will be more appropriate. For example, it is nature to incorporate the pruning technology into the growing method in order to overcome overgrowth. Accordingly, in this paper, we will concretely analyze the weight-illgrowth action of cascade correlation algorithm, which is one of representative constructive algorithms, and use regularization to avoid this phenomenon for improving generalization.

2. Analyse of Original Cascade Correlation Algorithm

The original Cascade Correlation algorithm was proposed by Fahlman & Lebiere [14]. For the sake of clarity, we use flow-chart to describe the Cascade Correlation algorithm as Fig.(1). In Fig.(1), concrete contents for each step are listed below:

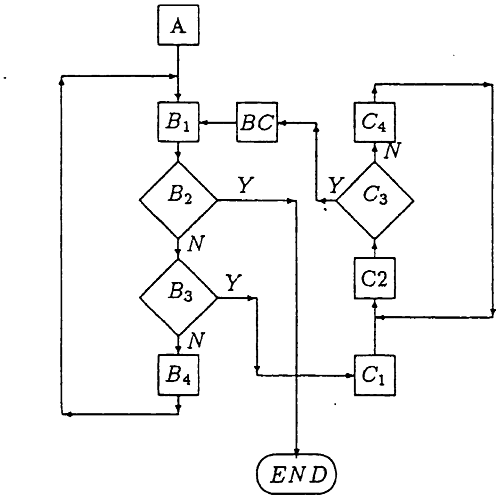

Fig. 1: The Original Cascade Correlation Algorithm

Step A: Initializing NN

1. determining the I/O units according to the practical problem.
2. input units are fully connected to output units.
3. initializing all connection weight values.

Step B1: Calculating the error over an epoch by

$$E = \frac{1}{2}\sum_{o,p}(y_{op} - t_{op})^2 \qquad (1)$$

or other error measures.

Step B2: Judging whether the performance, which depends on training error, is smaller than preset desirable value or not.

Step B3: Judging whether the overall (output units) stopping criterion, which depends on overall patient period and overall patient coefficient, is satisfied or not.

Step B4: Updating all weights connected to the output units by gradient descent, to minimize the cost function described by Equ.(1).

Step C1: Initializing candidate

1. connect all input units and previously installed hidden units to the candidate.
2. initializing the new weights.

Step C2: Calculating the correlation magnitude over an epoch by

$$C = \sum_{o}\left|\sum_{p}(y_p - \bar{y})(e_{op} - \bar{e_o})\right| \qquad (2)$$

Step C3: Judging whether the local (adding hidden unit) stopping criterion, which depends on local patient period and local patient coefficient, is satisfied or not.

Step C4: Updating the weights connected to the adding hidden unit by gradient ascent, to maximize the objective function described by Equ. (2).

Step BC: Installing the trained candidate, this is the interface between B series and C series.

1. all input connections to the candidate are frozen.
2. the candidate is connected to all output units.
3. initializing the newly connected weight values.

In fact, Cascade Correlation algorithm combines two key ideas: The training builds up cascade structure as the Tower method and the hidden unit grabs the output residual errors greedily similar to the Upstart method. From Fig.(1), we may know that the training process can be divided into two parts: one is the training of hidden unit, which maximizes the correlation between the output of hidden unit and the sum of the residual errors at the outputs and then makes the incoming connections to the new hidden unit frozen; another is the training of output units, which only trains a layer of connections relating to output units. Due to the following distinctive features: full short-cut connections, single hidden node per layer, two-stage training, freezing technology etc., it has several advantages over existing constructive algorithms: it requires no backpropagation of error signals through the connections of the network since only one layer is trained at each training stage, and therefore learns very quickly, moreover the moving target problem for the Backpropagation algorithm shoulde be reduced [14], the step-size problem and the problem of oscillating around the global minimum for the Backpropagation algorithm is avoided by dynamically adding new hidden units to the network, which means that the algorithm won't get stucked at a local minimum and oscillate around the global minimum because the residual error will cause the algorithm to add a new unit, changing the weight space[8].

Whereas, some strategies of the cascade correlation algorithm which brings up the above advantages, meanwhile, cause some possible problems as follows:

(1) In fact, maxmizing the covariance has a tendency to update weights to overcompensate errors, which we refered to as weight-illgrowth. This makes the algorithm normally generates undesirable zigzag hidden units mapping and thus results in unsmooth zigzag classification/regression surface, so that the algorithm works better for classification tasks than for regression tasks.

(2) Single-node cascaded structure results in a network that can provide very strong nonlinearities. But it can be a disadvantage if the practical problem doesn't require such strong nonlinearities or no suficient number of training examples are available to control it[12].

(3) The advantage of the freezing method is the relatively small retraining effort required when a new element is introduced. However, the method can not, in general, achieve the desired solution. The reason is that when an extra degree of freedom is introduced (e.g., by adding a new weight to the network), holding the existing network values constantly will only find the solution in an affine subset of the existing weight space. Additional degrees of freedom (extra nodes or weights) can be introduced to allow this affine subset to pass through a global minimum point (in effect, shifting some of the previously fixed coordinates). But because of the partial duplication of effort involved (i.e., two or more weights are needed to determine the value of a single dimension), such a network can not be minimal in size[4].

(4) The technique of full interconnection for each hidden unit leads to deep networks which have many layers. In addition, the fan-in of hidden units rises linearly with the size of the network. Hence, scaling ability and implementation degrade as a network grows[9].

3. Regularization and Generalization

Regularization theory was firstly proposed by Tikhonov in 1963 in the context of functional approximation. The spirit of regularization is to stabilize the solution by means of some auxiliary functional that embeds prior information, e.g., smoothness constrains on the input-output mapping, and thereby make an ill-posed problem into a well-posed one. The principle of regularization may be stated below:
Find the function $F(x)$ that minimizes the cost function, defined by

$$\Phi(F) = \Phi_s(F) + \lambda\Phi_c(F) \tag{3}$$

where $\Phi_s(F)$ is the standard error term (e.g., the mean squared error), $\Phi_c(F)$ is the regularizing term, and λ is the regularization parameter.

It is well known that the generalized BP algorithm uses regularization term (called by complexity penalty term) to improve generalization and the RBF Neural Network is also based on regularization theory. Some papers furtherly examing the relation between generalization and regularization in Neural Network have been in literature. J. Moody (1992) presented an analysis of how the expected test set error (generalization) relates to the expected training set error for nonlinear learning systems, such as multilayer perceptrons and RBF neural network, and attained an important result which gives the following relation among generalization, training error and regularization for nonlinear learning system[11]:

$$E[\varepsilon_{test}(\lambda)]_{\xi\xi'} \approx E[\varepsilon_{train}(\lambda)]_{\xi} + 2\sigma^2_{eff}\frac{P_{eff}(\lambda)}{n}. \tag{4}$$

where, ξ, ξ' represents training set and test set respectively, n is the size of the training sample, σ^2_{eff} is the effective noise variance in the response variable, λ is a regularization parameter, and $P_{eff}(\lambda)$ is the effective number of parameters in the nonlinear model. The expectations E are taken over all possible training sets ξ and test sets ξ' respectively.

It should be noted that if given appropriate regularization term, at the right side of the above equation, generally, the first term increases with λ, while the second term decreases with λ. Furthermore, whenever $\sigma^2_{eff} > 0$ and $n < \infty$, there should exist $\lambda_{optimal} > 0$ such that the generalization is minimized.

Therefore, aiming at the first one of the problems existing in Cascade Correlatin algorithm described above, we suggest that regularization will help to settle down this problem, and whereby improve generalization if incorporating it into Cascade Correlation algorithm. But, so far, there isn't detailed research for regularizing Cascade Correlation algorithm yet. In the sequel, we will addrese this problem and give empirical study.

As we have known,' Cascade Correlation algorithm possess two-stage training: one is the training of new adding unit; another is the training of output units. Hence, there exits three possible regularization strategies: the first is adding regularization term into the first stage similar to BP algorithm, which is refered as to "overall regularization"; the second is adding regularization term into the second stage, which is refered as to "local regularization"; and the third is combining both of them. Comparing them, we suppose that the second is the most appropriate, because the zigzag mapping is mainly caused by the action of the hidden units due to using the greedy criterion and it is difficult to determine the optimal pair of λ simultaneously. Therefore, we add the regularization term into Equ.(2), so that Equ.(2) can be rewritten as:

$$C=\sum_{o}\left|\sum_{p}(y_p-\bar{y})(e_{op}-\bar{e}_o)\right|-\lambda\Phi_c(W) \quad (5)$$

4. Empirical Study

In our simulation, in addition to some known typical regularization terms, we introduce two new regularization terms in light of the special cacacde structure.

Regularization Term[1]

This form of regularization term is usually refered to as representative weight decay[Plaut et al., 1986], because it gives each connection weight w_{ij} a tendency to decay, even though it can not make the weights decay constantly, whose expression is:

$$\lambda\Phi_c(W)=\lambda\sum_{(ij)}(w_{ij}^2) \quad (6)$$

Clearly, it penalizes large weight much more than small weight and eventually tend to obtain a weight vector which possess many small components.

Regularization Term[2]

To make small weight decay more quickly to zero than large one, the following regularization term was proposed by [13].

$$\lambda\Phi_c(W)=\lambda\sum_{(ij)}\frac{w_{ij}^2}{1+w_{ij}^2} \quad (7)$$

Regularization Term[3]

Ishikawa (1989) presented regularization term with constant decay rate, which is called as weight forgetting by him. Its expression is:

$$\lambda\Phi_c(W)=\lambda\sum_{(ij)}|w_{ij}| \quad (8)$$

This form remedies the deficiency that regularization term[1] can not make unnecessary connections fade away and a skeletal network emerges[6].

In fact, noting the speciality of cascade structure, we may put forward some new regularization terms, for instances below:

Regularization Term[4]

If the practical problem requires stronger nonlinearities, we may adopt the following regularization term:

$$\lambda\Phi_c(W)=\frac{\lambda L_{ij}^2}{L_i^2}\sum_{(ij)}|w_{ij}| \quad (9)$$

Where, L_i denotes the depth of hidden unit i, while L_{ij} denotes the length of the connection from node i to node j.

This form tends to make hidden units absorb the latest information from the nearest neighbours, so that the higher order feature detection can be obtained.

Regularization Term[5]

If the practical problem discourages strong nonlinearities, in contrast to regularization term[4], the regularization term is as follow:

$$\lambda\Phi_c(W)=\frac{\lambda(L_i-L_{ij}+1)^2}{L_i^2}\sum_{(ij)}|w_{ij}| \quad (10)$$

Obviously, it supports this assumption that the latest information from the nearest neighbours has become the high order feature detection which can hardly be refined.

To test the effectiveness of our ideas, we have done a simulation on classification problem and regression problem respectively. For the sake of exactness, we have made use of the program code based on the original Casacde Correlation code on a ftp site, since it is public.

4.1. Classification Problem

In this experiment, we adopt two-spirals problem, since it has been reported that the Cascade Correlation algorithm has achieved good performance in the benchmark in which Back Propagation algorithm fails. The comparative results are shown below, The performance measures are ensemble-averaged over 10 indenpendent trials, using a sigmoidal activation function for both output units and hiddenunits, a pool of 8 candidate units, and the maximum learning iteration numbers 200 for learning both the hidden-layer weights and the output-layer weights.

Table: 1: Comparison of Percent Correct for two-spirials problem

CasCor	$\lambda_{optimal}$	H.U.s	Epochs	Corr.	M.Corr.
Original		12.9	103.8	95.0	97.42
Regu.[1]	0.000008	13.9	118.5	96.1	97.42
Regu.[2]	0.000005	13.0	110.8	95.4	97.94
Regu.[3]	0.000009	12.6	104.5	95.9	99.48
Regu.[4]	0.000010	13.3	108.4	95.9	97.94
Regu.[5]	0.000008	13.7	119.1	96.0	97.94

H.U.s: Hidden Units; Corr.: Percent Correct
M.Corr.: Maximum of Percrnt Correct among 10 Trials
Regu.: Regularization

4.2. Functional Approximation

With regard to functional approximation, when we assess which is a 'good' or a 'poor' functional approximation if we are to compare fitness, some criterion is necessary. A Adams & S Waugh (1995) suggested an intuitive concept that a function that goes near the training points and varies smoothly between them is better than one which fits these points as well or better but which varies widely from the linear approximation between them. For the sake of simplicity, the simple function $f(x) = \frac{1}{1+x^2}$ is used for the comparative simulation, which is a peak with a top but has far wide wings. We have known that this function can be approximated well with a 2:2:1backpropagation network (one of the inputs is the constant bias input) and poorly fitted with the original Cascade Correlation algorithm[10]. Eleven pairs of equally spaced values are used as the training points with x varying from -5 to +5. The interpolation (generalization) capabilities are tested by evaluating the function at 101 points over the same range and finding the root mean square error.

5. Conclusion and Discussion

We have made a comparative emperical study for classification and regression problem. From the simulation of two-spirals problem, the improvement of generalization is not significant, because sometimes the zigzag output mapping affects little the classification results. Nevertheless, it is worthwhile to notice that, using regularization term[3] reduces the hidden units required and acquires the good percent correct (99.48%). But, for the regression problem, the effect of the regularization is evident. As what we have seen from Fig.2(a), the output curvature of the original cascade-correlation algorithm has obvious zigzag phenomenon, so that the sum squared error is 0.358; hidden units are 5. But after adding regularization term, smoothness evidently emerges, resulting in sum squared error decreasement. Furtherly, as what we expected, it should be noted that regularization term[4] makes the output fit better than other methods on both wings which means strong nonlinearities [see Fig.2(e)]; regularization term[5] makes the output fit better than other ones on the peak of top which represents partially linear part [see Fig.2(f)], so that the sum squared error is lower but the hidden units are more needed.

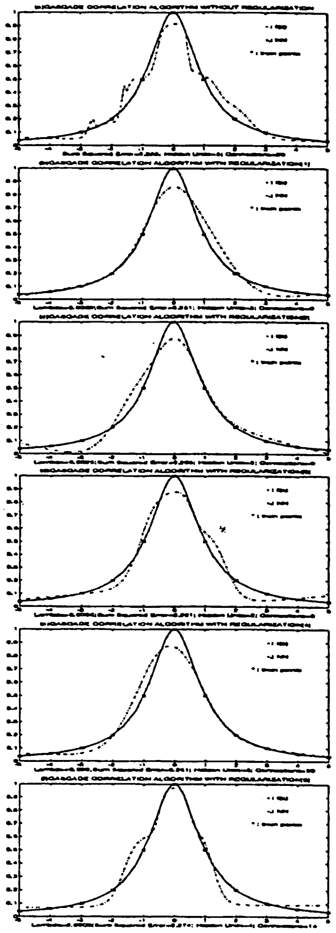

Fig. 2: Relatively Smooth Output of Cascade Correlation Algorithm with Regularization

Cascade Correlation algorithm with local regularization effectively avoid weight-illgrowth to improve the generalization, especially for regression problem. But, as we have seen from Fig.2, there are still local ill-fitted part. Therefore, it merits further research. It is a possible way to use multi-regularization terms.

References

[1] Rumelhart, D.E., Hinton, G.E., and Williams, R.J., (1986) *Learning Internal Representations by Error Propagation.* in Parallel Distributed Processing: Explorations in the Microstructures of Cognition, Rumelhart, D.E., McClelland, J.L., and the PDP Research Group, Editors, vol. 1. Cambridge, MA. MIT Press, 1986.

[2] Fashlman, S.E., (1988) *Faster-learning Variva-tions on Back-Propagation: An Empirical Study.* in Proceedings of the 1988 Connectionist Models Summer School, Morgan Kaufmann.

[3] Michael C.Mozer & Paul Smolensky (1989) *Using Relevance to Reduce Network Size Automatically.* Connection Science, Vol. 1, No. 1,1989

[4] Timur Ash (1989) *Dynamic Node Creation in Backpropagation Networks.* Connection Science, Vol. 1, No. 4,1989

[5] Reed,R. (1993) *Pruning algorithms - a survey.* IEEE Transactions on Neural Networks 4(5): 3-16.

[6] M.Ishikawa,*A structural learning algorithm with forgetting of link weights.* Technical Report TR-90-7, Electrotechnical Laboratory, Japan.

[7] M.Ishikawa, (1996) *Structural Learning with Forgetting.* Neural Networks, Vol.9, No.3, pp. 509-521,1996

[8] Natalio Simon,(1993) *Constructive Supervised Learning Algorithms for Artificial Neural Networks.* Delfty University of Technology. Neuroprose archive.

[9] Ira G. Smotroff, David H. Friedman, and Dennis Connoly,*Large Scale Networks via Self Organizing Hierarchical Networks.* 544 / SPIE Vol. 1469 Applications of Artificial Neural Networks 2 (1991)

[10] A Adams & S Waugh, *Function Evaluation and the Cascade-Correlation Architecture.* in IEEE International Conference on Neural Networks, 1995. p. 942-946

[11] John E.Moody (1992), *The effective Number of Parameters: An Analysis of Generalization and Regularization in Nonlinear Learning Systems.* NIPS4,Morgan Kaufmann Publishers,1992

[12] Sjogaard,S. (1991), *A Conceptual approach to generalization in dynamic neural networks.* Aarhus Universty, Denmark. Neuroprose archive.

[13] A.S. Weigend, D.E. Rumelhart, and B.A. Huberman, *Back-propagation, weight-elimination and time series prediction,* in Proc. 1990 Connectionist Models Summer School, PP. 105-116.

[14] Scott E.Fahlman & Christian Lebiere (1990), *The Cascade-Correlation Learning Architecture.* Advances in Neural Information Processing Systems 2, edited by D.S. Touretzky, pp.524-532, Morgan Kaufmann Publishers Inc.,CA,1990.