A New Incremental Learning Technique[1]

Nick Dunkin, John Shawe-Taylor
Department of Computer Science
Royal Holloway College
University of London
U.K.
{nickd, john}@dcs.rhbnc.ac.uk

Pascal Koiran
LIP
ENS-Lyon
46 Allée d'Italie
Lyon
France
koiran@lip.ens-lyon.fr

NeuroCOLT Technical Report Series

NC-TR-97-014

February 1997[2]

Produced as part of the ESPRIT Working Group
in Neural and Computational Learning,
NeuroCOLT 8556

NeuroCOLT Coordinating Partner

Royal Holloway
University of London

Department of Computer Science
Egham, Surrey TW20 0EX, England

For more information contact John Shawe-Taylor at the above address
or email neurocolt@dcs.rhbnc.ac.uk

[1]Also appeared in Neural Nets, WIRN Vietri-96, Proceedings of the 8th Italian Workshop on Neural Nets. Springer, 1997. pp. 112–118.
[2]Received February 28, 1997

Abstract

We present a new type of constructive algorithm for incremental learning. The algorithm overcomes many of the problems associated with standard back propagation such as speed and optimum network size. We investigate the ability of the network to learn and test the resulting generalisation of the network.

keywords : *Incremental learning, neural networks, back propagation*

1 Introduction

The back propagation method of training neural networks, although popular, has many disadvantages. The most significant of these is the time that the back propagation algorithm takes to train a network; it is often necessary to run the algorithm for many hundreds of epochs before a minimum is reached.

The other major disadvantage of the back propagation algorithm has to do with the size of the network. The back propagation algorithm slows down considerably as the size of the network increases so an over estimation of the initial network size can lead to drastic increases in learning time.

There are also the problems of *over* and *under* fitting. If a network does not have enough connections then the network will not be able to learn the desired function, if however the network possesses too many connections then the architecture may appear to learn the desired function but the resulting generalisation properties of the neural network will be poor. Such a network has *over-fitted* the examples and is acting more like a look-up table.

In response to these learning problems researchers have begun to investigate constructive network architectures [2, 3, 5, 1]. Constructive neural network algorithms build a network as part of the learning process, the aim being to speed up the learning process and let the size of the network be determined from the data. Unfortunately, many constructive algorithms have a tendency to add too many nodes and end up over-fitting the training data.

Such is the case with the popular cascade correlation architecture which adds nodes incrementally as new singular layers. Once a unit has been trained its weight connections are fixed, so if this unit has learned a *feature* that turns out to be incorrect through later training it must be compensated for by later units. This can lead to over-fitting and poor generalisation properties.

In this paper we describe a new incremental learning algorithm that constructs a single sigmoidal *hidden* layer during learning in such a way that *over-fitting* is avoided, resulting in good generalisation properties and short learning times.

2 A New Incremental Algorithm

The new incremental algorithm uses two principles which combine to produce quick training times and good generalisation properties.

Firstly, at each stage only one node in the network is being trained and hence this can be done very efficiently. Secondly, the algorithm bounds the contribution of each added node which limits the expressibility of the network and thus gives desirable generalisation properties.

The initial network contains one unit in the hidden layer with input size dictated by the training data. The internal units have a sigmoidal activation function with the output node producing a simple linear function of its inputs. Every node in the internal layer has a threshold value associated with it.

Training begins with the single internal node and progresses until the reduction of the total squared error of the network is less than some constant β (usually 0.0001) over ω iterations, where ω is a constant (usually 500).

When the network has reached this *boredom threshold*, training stops and the connection weight values of the network are frozen. At this time a new unit is added to the internal layer of the network. Before training can recommence it is necessary to factor down the contribution of the previously trained network. All the previously trained connections in the output layer are factored by $(1 - \bar{\alpha})$ where $\bar{\alpha}$ has a value of $1/n$. The contribution of the new node in the network is multiplied by a value of $\bar{\alpha}$. At any time during training only one sigmoidal node is being adapted. In the next section we show how, at each stage of the incremental process, the function learned by the single sigmoid node will work towards a general solution.

2.1 Theoretical Result

The following theorem was first presented by Koiran [4] and shows that an algorithm exists for training a network with n hidden nodes incrementally, that is in terms of the functionality, as good as training a fixed network with n hidden nodes. The incremental training will be of course, much easier

Theorem 1.0. *Let G be a subset of a Hilbert space H, with $||g|| \leq B$ for each $g \epsilon G$.*
Let $m = inf_{F \epsilon co(G)}||F - f||$, where $f \epsilon H$ and $co(G)$ is the convex hull of G.
Then for every $c > \sqrt{B^2 + m^2}$,

$$||f_n - f|| \leq m + \frac{c}{\sqrt{n}},$$

where $f_n, n = 1, 2, ...$ is any sequence chosen to satisfy

$$||f_1 - f||^2 \leq inf_{g \epsilon G}||g - f||^2 + \epsilon_1,$$

and, for $n \geq 2$,

$$||f_n - f||^2 \leq inf_{g \epsilon G}||\alpha_n f_{n-1} + \bar{\alpha}_n g - f||^2 + \epsilon_n,$$

where $\alpha_n = (n-1)/n$, $\bar{\alpha}_n = 1 - \alpha_n = 1/n$, and $\epsilon_n \leq (c^2 - B^2 - m^2)/n^2$.

Theorem 1.0 shows that a solution to some problem can be derived incrementally. Let us take G as the set of possible functions realisable by a single sigmoidal unit, whose output is multiplied by a constant value, at most B,

$$G = \{A\sigma(< w, x >)|A < B\}$$

where $\sigma(x)$ is the sigmoid function and $< w, x >$ denotes the inner products of vectors w and x.

The function f_n is the convex combination of n functions from G; the state of the frozen network with n internal nodes,

$$f_n = (g + (n-1)f_{n-1})/n,$$

where g is some function from G. This means that at each addition stage of the algorithm the training of the new node is equivalent to finding g_0 given f_{n-1} such that,

$$||\alpha f_{n-1} + \bar{\alpha} g_0 - f||^2 < \ inf_{g \epsilon G}||\alpha f_{n-1} + \bar{\alpha} g - f||^2 + \epsilon_n,$$

where $\alpha + \bar{\alpha} = 1, \bar{\alpha} = 1/n$. f is the target and $\epsilon_n = 1/n^2$

The function g_0 is determined by the weight vector w and output multiplier a. These are the values we must compute in order to find g_0. The weight vector is found by performing back propagation on the one node network. The output multiplier is found once this weight vector has been determined.

Training continues, using standard back propagation, until the desired error level is achieved. Back propagation is often disregarded due to its slow training times but our algorithm is only ever training one internal node at a time so back propagation is more than satisfactory as a training algorithm.

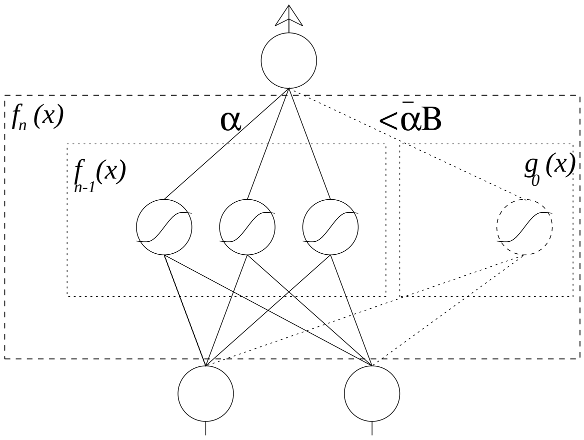

Figure 1 The network $f_{n-1}(x)$ with the addition of a new node ($g_0(x)$) giving the new network $f_n(x)$

3 Testing and Evaluation

In this section we describe the testing of the new algorithm and compare it with another popular incremental algorithm; Fahlman's cascade correlation [2].

3.1 Function Approximation

The algorithm was tested on function approximation problems, with both noisy and clean data.

3.2 Function One

The first function was a simple three dimensional function of the form,

$$f(x,y,z) = (x * \cos(y-z)) + \frac{x^2}{2} * \sin(2(y-z))$$

The complete training set consisted of 125 sets of example presentations. This training set was divided into 10 separate training sets which contained 20, 30, 40, 50, 60, 70, 80, 90, 100 and 125 training examples. These training sets were used to train ten networks. Each network converged to an acceptably low error.
Each of the ten networks were then presented with a test set that contained 25 previously unseen examples. The results of these tests are presented in the next section.

3.3 Function Two

The second function to be approximated was a two dimensional function,

$$f(x,y) = (exp(cos(4 * (x+y)))/2)$$

Eleven test sets were generated at random, from a normal distribution between 0 and 1 on the x and the y axis. The number of examples in each test set were 10, 20, 30, 40,50, 60, 70, 80, 90, 100 and 125. Eleven networks were trained with the test sets and then each network was tested on a test set of 25 previously unseen examples drawn at random from the a normal distribution in the same way as before.

This second experiment was then repeated with the addition of random noise on the input data. The randomly introduced noise was in the range $(-0.05 - +0.05)$.

3.4 Comparison with Cascade Correlation

One of the most popular constructive algorithms is the Cascade Correlation architecture described by Fahlman and Labiere. The function approximation problems were tested on a cascade correlation network and the results are presented in the next section.

It can be seen from the graph that the new incremental algorithm generalises well on the unseen test set. The plot flattens out to a constant low error of approximately 0.04 TSE(Total Squared Error). The plot for the cascade correlation architecture shows erratic behaviour, sometimes performing well on the unseen examples and other times performing badly. The conjecture is that although the cascade correlation architecture learns the training sets within an allowable error, the resulting generalisation properties of the trained network are unsatisfactory.

Cascade correlation performs badly on the training set with additional noise, displaying poor error rates on the test set and non uniform behaviour as the number of examples in the original training set increases. There is also an indication that over-fitting occurs in noisy data. As the number of noisy examples increases towards a 125 the performance of cascade correlation deteriorates, indicating that over-fitting may be occurring.

4 Results of Experimental Work

4.1 Comparative Results

For each experiment a graph was plotted to show how cascade correlation and the new incremental algorithm performed on the previously unseen test set with an increasing number of examples in the training set.

The performance of cascade correlation in the first experiment is particularly poor with out new algorithm producing a much more uniform behaviour pattern.

In the second experiment, on the two dimensional data, cascade correlation begins to perform more satisfactorily, however there are some problems apparent at the lower end of the training sets.

The training set with additional noise provides cascade correlation with some problems. There is indication of over-fitting at the far right of the graph and throughout the test sets the performance on the unseen data is unpredictable. The trace produced by the new algorithm is very similar to the trace produced for the clean data, indicating that generalisation properties of this new algorithm are good and that the addition of noise has not resulted in over-fitting of the training data.

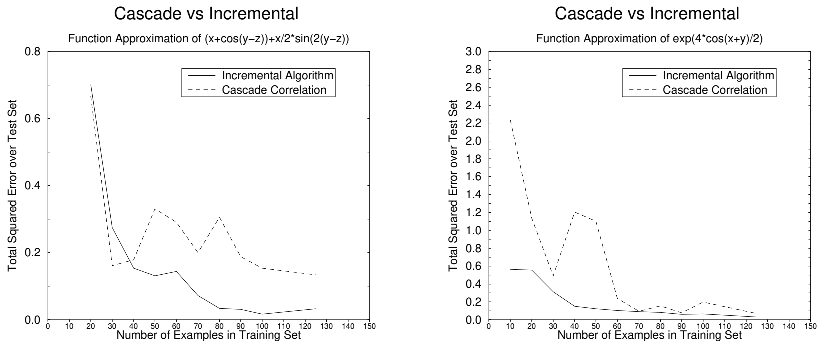

Figure 2 Results for Function One and Function Two

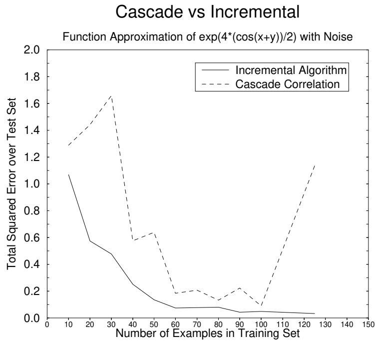

Figure 3 Function Two - with additional noise

4.2 The Approximated Manifolds

It is interesting to look at the manifold described by the network generated in the second function experiment. The following approximation, from the incremental algorithm, was with 60 training points and the addition of six internal nodes.

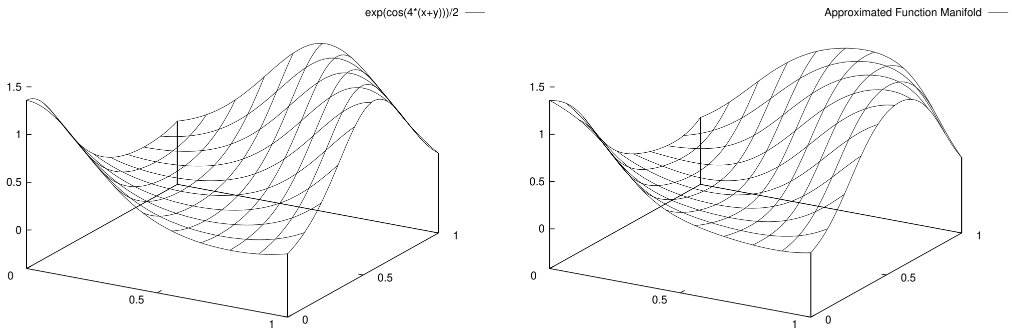

Figure 4 Target and approximated functions

5 Discussion and Further Work

In this paper we have shown, both theoretically and experimentally that this new algorithm performs well and results in a network whose generalisation properties are comparable with that of a neural network with a fixed *a priori* size.

Further comparisons need to be done, first with cascade correlation and then with other incremental algorithms.

References

[1] D.DeMers & G.Cottrel (1992) Non-Linear Dimensionality Reduction, University of California 1992

[2] Scott Fahlman & Christian Lebiere (1991). The Cascade-Correlation Learning Architecture, CMU-CS-90-100, Technical Report

[3] Christian Jutten & Rachida Chentouf(1995), A New Scheme for Incremental Learning, Neural Processing Letters

[4] Pascal Koiran (1994) Efficient Learning of Continuous Neural Networks, COLT 94 Techniques in Neural Learning, NC-TR-95-036, Technical Report

[5] Tin-Yau Kwok & Dit Yan Yeung(1995), Constructive Feedforward Neural Networks for Regression Problems : A Survey, HKUST-CS95-43, Technical Report