July 2000

Building MLP networks by construction

Ah Chung Tsoi
University of Wollongong, act@uow.edu.au

M. Hagenbuchner
University of Wollongong, markus@uow.edu.au

A. Micheli
University of Wollongong

Building MLP networks by construction

Abstract

We introduce two new models which are obtained through the modification of the well known methods MLP and cascade correlation. These two methods differ fundamentally as they employ learning techniques and produce network architectures that are not directly comparable. We extended the MLP architecture, and reduced the constructive method to obtain very comparable network architectures. The greatest benefit of these new models is that we can obtain an MLP-structured network through a constructive method based on the cascade correlation algorithm, and that we can train a cascade correlation structured network using the standard MLP learning technique. Additionally, we show that cascade correlation is a universal approximator, a fact that has not yet been discussed in literature.

Keywords

correlation methods, function approximation, learning (artificial intelligence), multilayer perceptrons, neural net architecture

Disciplines

Physical Sciences and Mathematics

Publication Details

This paper originally appeared as: Tsoi, AC, Hagenbuchner, M & Micheli, A, Building MLP networks by construction, IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, 24-27 July 2000, vol 4, 549-554. Copyright IEEE 2000.

Building MLP Networks by Construction

Ah Chung Tsoi, Markus Hagenbuchner, Alessio Micheli

University of Wollongong
Wollongong, NSW 2522
Australia

Email: {ah_chung_tsoi, markus}@uow.edu.au
Email: micheli@di.unipi.it

Abstract: In this paper, we introduce two new models which we obtain through the modification of the well known methods MLP and cascade correlation [1]. These two methods differ fundamentally as they employ learning techniques and produce network architectures that are not directly comparable. We *extended* the MLP architecture, and *reduced* the constructive method to obtain very comparable network architectures. The greatest benefit of these new models is, that we can obtain an MLP-structured network through a constructive method based on the cascade correlation algorithm, and that we can train a cascade correlation structured network using the standard MLP learning technique. Additionally, we will show that cascade correlation is a universal approximator, a fact that has not yet been discussed in literature.

1 Introduction

As is well known, a major difficulty in the development of multilayer perceptrons is that the number of hidden layer neurons is obtained by trial and error. There are no known practical methods which can provide an estimate of the number of hidden layer neurons. On the other hand, there exists a number of constructive methods which can be deployed to design a neural network. However, these methods result in neural networks which do not have the usual multilayer perceptron architecture, or are of a high computational complexity.

In this paper we introduce a constructive method which builds up a MLP architecture by adding hidden layer neurons as needed. As described later in this paper, this method is based on cascade correlation [1]. Cascade correlation , is known to have a low computational complexity, and has successfully been applied to a wide range of applications. In addition, we found that cascade correlation is also a universal approximator.

Based on observations made when comparing CC and MLP, we formulated an extended MLP architecture which results in the same network architecture as obtained from the CC algorithm. This extension closes the loop in that we have now a method which allows us to constructively build up a MLP structured network from "scratch", and train a CC structured network using the standard MLP learning technique.

The structure of this paper is as follows: Section 2 gives a brief overview of the MLP architecture, and introduces notations used throughout this paper. In Section 3, we briefly explain a constructive method which is known as cascade correlation [1]. Extended models are introduced in section 4. A constructive method which dynamically builds up a MLP structured network is introduced in section 4.1. A modification of the model described in section 2 produces a neural architecture comparable to the one obtained by cascade correlation. This is given in section 4.2. In section 5 we compare the performances of the model described in section 4.2 on a benchmark problem. Finally in Section 6, we summarize the work presented in this paper and draw some conclusions.

0-7695-0619-4/00 $10.00 © 2000 IEEE

2 Multilayer Perceptrons

The architecture of a classic multilayer perceptron (MLP) is given as follows:

$$\mathbf{y} = F_p(C\mathbf{x}) \tag{1}$$
$$\mathbf{x} = F_n(B\mathbf{u}) \tag{2}$$

where $\mathbf{u}$ is a m dimensional input vector, $\mathbf{x}$ is a n dimensional vector denoting the outputs of the hidden units, and $\mathbf{y}$ is the p dimensional output vector. B and C are matrices of appropriate dimensions. $F_n(\cdot)$ is an n dimensional vector:

$$F_n(\cdot) = \begin{bmatrix} f(\cdot) \\ f(\cdot) \\ \vdots \\ f(\cdot) \end{bmatrix} \tag{3}$$

$f(\cdot)$ is a nonlinear function. This can be a sigmoidal function, or a hyperbolic tangent function. $F_p(\cdot)$ is a p dimensional vector, defined similarly to $F_n(\cdot)$.

We will refer to this architecture as the classic MLP architecture. Note that we have abused the notation here slightly in that we include the bias in the model. Each hidden layer neuron and each output neuron is assumed to include a bias. This is commonly handled by assuming an extra node which has a constant output.

It has been found by [2], that it is more beneficial to include a direct feedthrough of the input to the output. Hence we use a more extended architecture as follows:

$$\mathbf{y} = F_p(C\mathbf{x} + D\mathbf{u}) \tag{4}$$
$$\mathbf{x} = F_n(B\mathbf{u}) \tag{5}$$

We will refer to this as the MLP architecture. The total number of parameters of this architecture is $n \times m + n \times p + m \times p$.

3 Cascade Correlation

The cascade correlation neural network architecture, as introduced by Fahlman [1], shown in Figure 1, can be described by the following model:

$$\mathbf{y} = F_p(C\mathbf{x} + D\mathbf{u}) \tag{6}$$
$$\mathbf{x} = F_n(A\mathbf{x} + B\mathbf{u}) \tag{7}$$

where $\mathbf{u}$ is a m dimensional input, $\mathbf{x}$ is an n dimensional vector denoting the hidden layer neuron activations, and $\mathbf{y}$ is a p dimensional output. Boldface variables denote vector quantities. A, B, C, and D are respectively $n \times n$, $n \times m$, $p \times n$ and $p \times m$ matrices.

Note that A has a special structure: it is a lower triangular matrix with zero diagonal elements. With the lower triangular structure, it indicates that successive elements of $\mathbf{x}$ are influenced by the predecessor values of $\mathbf{x}$. In other words, x_i depends on x_j, $j = 1, 2, \ldots, i-1$, where x_i is the ith element in the vector $\mathbf{x}$.

It is noted that the cascade correlation architecture has $n \times m + n \times p + p \times m + \frac{1}{2}n(n-1) = p(m+n) + nm + \frac{1}{2}n(n-1)$ parameters. Thus, cascade correlation has more parameters than the MLP. Specifically, it has $\frac{1}{2}n(n-1)$ parameters more than the corresponding MLP case.

Training of a cascade correlation architecture is by construction, i.e., it builds the architecture step by step, by adding one hidden layer neurons one at a time. Since it is a multilayer neural network framework, it

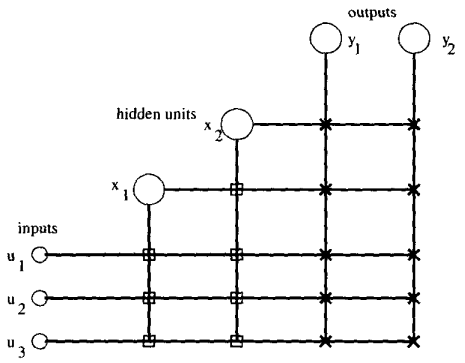

Figure 1: A diagram showing a cascade correlation neural network architecture with two hidden units, three input units, and two output units. The hidden units have nonlinear activation functions. The output units either have nonlinear activations, for classification problems; or linear activations, for regression problems. The input units all have linear activation functions. The square symbol denotes that the weights of the unit, once obtained will be frozen. The cross symbol denotes weights which are required to be trained.

has to solve the issue of how to train hidden layer neurons without information concerning the hidden layer neuron behaviours. In this instance, Fahlman used two ideas: (1) use a number of candidate units [1], and (2) the correlation between the output of the hidden layer neuron and the error, as obtained from a **previous** iteration. The correlation measure is defined as follows:

$$E_c = \sum_{j=1}^{p} \left| \sum_{i=1}^{N} (x(i) - \overline{x(i)})(e_j(i) - \overline{e_j(i)}) \right| \tag{8}$$

where $x(i)$ is the output of x in response to the ith input, and $e_j(i)$ is the jth error in response to the ith training input in the **previous** iteration. $\overline{x}$ denotes the average value of x over the training data set. This correlation measure gives a criterion in the choice of the candidate units. The candidate unit with the highest correlation will be selected to be the hidden layer neuron. The other candidate units are discarded. Section 4.1 gives a more detailed description of the learning algorithm.

4 Extended classes of architectures

From a comparison between MLP, and the CC neural network architecture, it becomes clear that we can formulate other neural network architectures within these classes. The only difference between the cascade correlation neural network architecture and the MLP is the $\frac{1}{2}n(n-1)$ weights from the preceding hidden units to the succeeding units in the case of the cascade correlation architecture. Based on this observation, it is possible to formulate two novel architectures which we call *reduced CC*, and *extended MLP*.

4.1 Reduced cascade correlation

A reduced CC neural network architecture is obtained by disallowing the connection from the previous hidden layer units to the succeeding hidden layer units. Thus, we have the following architecture:

$$\mathbf{y} = F_p(C\mathbf{x} + D\mathbf{u}) \tag{9}$$

$$\mathbf{x} = F_n(B\mathbf{u}) \tag{10}$$

[1] The candidate units differ from one another in the initial conditions only. Otherwise they are identical.

This is an equivalent to the multilayer perceptron architecture, except that the training algorithm is a step by step training algorithm, instead of being a batch training algorithm, as in the case of the MLP situation.

Given a training set $T = \{\mathbf{u}(i), \mathbf{t}(i); i = 1, 2, \ldots, N\}$, the training algorithm can be described as follows:

Step 1 We form a direct input output model as follows:

$$\mathbf{y} = F_p(D_0\mathbf{u}) \tag{11}$$

where D_0 is a $p \times m$ matrix, which can be obtained by minimizing the error criterion. Store the error sequence $\mathbf{e}(i) = \mathbf{t}(i) - \mathbf{y}(i)$ for use in Step 2.

Step 2 We use candidate units, each candidate unit is modeled as follows:

$$x = f(\mathbf{b}_1\mathbf{u}) \tag{12}$$

where $\mathbf{b}_1$ is a m vector. The unknown parameter $\mathbf{b}_1$ can be trained by maximizing the correlation measure (eq. 8) between the previous error sequence $\mathbf{e}(i)$, and the outputs of the candidate unit.

The candidate unit with the largest correlation measure will be selected as the hidden unit. The corresponding weight $\mathbf{b}_1$ is frozen. Let us denote the output of the hidden unit by x_1. Then, we can use this hidden unit together with the inputs to form the following input output model:

$$\mathbf{y} = F_p\left(D_1\mathbf{u} + \mathbf{c}_1 x_1\right) \tag{13}$$

where $\mathbf{c}_1$ is a p vector. The unknowns D_1 and $\mathbf{c}_1$ can be determined by minimizing the error criterion.

Step 3 The way ahead is clear as we can determine the next hidden unit using the correlation measure, and so on.

4.2 Extended multilayer perceptron architecture

On the other hand, it is possible to extend the multilayer perceptron to one which allows the preceding hidden layer neurons to influence the succeeding ones, i.e.,

$$\mathbf{y} = F_p\left(C\mathbf{x} + D\mathbf{u}\right) \tag{14}$$

$$\mathbf{x} = F_n\left(A\mathbf{x} + B\mathbf{u}\right) \tag{15}$$

The $n \times n$ matrix A has a very special structure, viz., it is lower diagonal, with zero elements in the diagonal. Thus even though the form of the architecture appears the same as a recurrent neural network, in actual it is different.

The training algorithm of this architecture can be determined by minimizing the error criterion. Note that the updating equations for matrices B, C and D are the same as those obtained in a MLP. As for the updating equation for matrix A, we need to proceed as follows: consider an element a_{ij} in the matrix A. We can differentiate the error criterion with respect to this parameter as follows:

$$\frac{\partial J}{\partial a_{ij}} = -\sum_{\ell=1}^{N} \delta^T(i) \frac{\partial \mathbf{x}}{\partial a_{ij}} \tag{16}$$

where $\delta(i) = C^T \Lambda(\mathbf{y})\mathbf{e}(i)$, and J is the error criterion. A popular error criterion is the least square error defined as:

$$J = \frac{1}{2}\sum_{i=1}^{N} tr((\mathbf{t}(i) - \mathbf{y}(i))(\mathbf{t}(i) - \mathbf{y}(i))^T) \tag{17}$$

$tr(\cdot)$ denotes the trace operator. Now, $\frac{\partial \mathbf{x}}{\partial a_{ij}}$ can be obtained by formally differentiating $\mathbf{x}$ with respect to a_{ij} as follows:

$$\frac{\partial \mathbf{x}}{\partial a_{ij}} = \Lambda(\mathbf{x})\left(A\frac{\partial \mathbf{x}}{\partial a_{ij}} + Q_{ij}\mathbf{x}\right) \tag{18}$$

where Q_{ij} is a $n \times n$ matrix with all elements 0, except an element of 1 in the i th row and j th column position. Eq(18) can be solved as follows:

$$\frac{\partial \mathbf{x}}{\partial a_{ij}} = (I - \Lambda(\mathbf{x})A)^{-1} Q_{ij}\mathbf{x} \tag{19}$$

The matrix inversion of $I - \Lambda(\mathbf{x})A$ is particular simple, as A is lower triangular with 0 in the diagonal elements. Thus, $I - \Lambda(\mathbf{x})A$ is a lower triangular matrix with all diagonal elements equal to 1. Thus, Eq (19) can be solved for the values of $\frac{\partial \mathbf{x}}{\partial a_{ij}}$. Once this is obtained, the value of $\frac{\partial J}{\partial a_{ij}}$ can be obtained, and hence the elements of A can be updated.

Note that we can determine the unknown matrix A in this simple manner is due to the fact that we only have feedforward links in the hidden layer neurons, i.e., x_j depends on x_i, $i = 1, 2, \ldots, j-1$. If it is full recurrent neural network, i.e., when A has no special structure, then, while the solution is still possible, as there exist many algorithms for inverting $I - \Lambda(\mathbf{x})A$, the nice insight into the operation of the architecture is lost.

Thus, it can be observed from this that the cascade correlation architecture occupies a place somewhere in between the fully recurrent neural network (with no delays in the interconnections among the hidden layer neurons), and the MLP. It is hence an interesting architecture to explore, even though we know that the MLP architecture by itself is already a universal approximator.

5 Comparison

We compare the behaviour of the MLP architecture with the extended model by considering data set `monk1` from the well known MONK's benchmark problem. This data set is supposed to be easy learnable. We used the following architecture: There were 6 input units which were fully connected to 7 hidden layer neurons, which itself were fully connected to a single output unit. The extended MLP architecture had additionally forward connections between hidden layer neurons as described in section 4.2. The networks were trained using standard back-propagation, momentum, or RProp learning. Table 1 displays the results. It shows the number of iterations required to obtain 100% classification rate on the test set. As it can be seen, the extended model is able to learn the given task faster than MLP when using the standard gradient descent method, but performs somewhat slower when updated by the RProp rule.

	std. backprop	momentum	RProp
MLP	490	464	98
Extended MLP	344	303	164

Table 1: Number of iterations required to learn the monks problem

6 Summary

This paper has introduced a methodology that allows the cascade correlation algorithm to be used to constructively build a MLP from "scratch". In other words, one does not need to know the size of the hidden layer at all. One only needs to use the cascade correlation and it will automatically find the MLP architecture. This observation is an important one, in that so far, everyone takes it for granted that it is not possible to infer automatically a MLP structure. One either finds a cascade correlation architecture which is not exactly the same as a MLP architecture, or a MLP architecture but one does not know how many hidden layer neuron to use.

By the very fact that it is possible to obtain a MLP architecture using cascade correlation, it is observed that the reduced cascade correlation architecture is a universal approximator. This again is an interesting observation. Whether the cascade correlation architecture is a universal approximator or not has not been discussed in the literature. However, from the observations made here, it is readily recognized that the reduced cascade correlation neural network is a universal approximator. Thus, by inference, the cascade correlation neural network architecture is a universal approximator. This may be one reason why it has been successful in practical applications.

It is also interesting to note that it is not possible to use the cascade correlation algorithm to obtain the classic MLP architecture, at least in its present form without modifications. Basically, in the classic MLP architecture, there is no direct feed through from the input to the output, i.e., matrix $D = 0$. As observed previously, the availability of the error from the immediate preceding step is crucial in the training of the hidden layer neurons. Without this, we cannot formulate the correlation measure. Secondly it appears that the influence of the input directly on the output is crucial to the cascade correlation training algorithm.

Fortunately, there is some theoretical justification in this formulation, i.e., with a direct feed through from the input to the output. It was shown [2] that the MLP architecture which includes a direct feed through from the input to the output performs better than the one without this direct feed through. Hence, one can safely make use of the extended MLP architecture.

The extended MLP architecture is also an interesting one, in that it is half way between a fully recurrent neural network, and the classic MLP. This architecture was not reported before, at least not in the form discussed here.

This also raises an interesting question: What are the functions of these links as represented by the matrix A? From a universal approximator point of view, these links are superfluous, i.e., they are not required to ensure a universal approximator property. But, their presence may facilitate easier learning, as they allow a more nonlinear mapping between the input and output, without creating an additional layer, as common in practical application of MLP. The idea of using a lower triangular matrix A with zero diagonal elements has not been suggested before. Hence, this may be the next architecture which one may attempt without having to use a two hidden layer MLP architecture.

References

[1] S. E. Fahlman. The cascade-correlation learning architecture. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 524–532, San Mateo, CA, 1990. Morgan Kaufmann Publishers.

[2] E. Sontag. Neural networks for control. In H.L. Trentelman and J.C. Willems, editors, *In Essay on Control: Perspectives in the Theory and its Applications*, pages pp. 339–380. Ed. Birkhauser, Boston, 1993.