ANALISIS 10 (1), 59-68 (2003)

A FRAMEWORK FOR MULTI-BACKPROPAGATION

WAN HUSSAIN WAN ISHAK
FADZILAH SIRAJ
ABU TALIB OTHMAN
Faculty of Information Technology
Universiti Utara Malaysia

ABSTRACT

Backpropagation algorithm is one of the most popular learning algorithms in the Neural Network. It has been successfully implemented in many applications. However, training Neural Networks involve a large amount of data. Therefore, training the network is time consuming as each training session requires several epochs, which usually takes several seconds or even minutes. This paper proposes a multi-backpropagation approach to minimize the complexity of the network. The approach does not require an alteration of the algorithm. Instead, the large network is split into several smaller networks. An integrating network is then constructed to integrate the output from the smaller networks.

Key Words: *Neural Network, Backpropagation Network, Multi Backpropagation Network.*

ABSTRAK

Algoritma Rambatan Balik merupakan salah satu algoritma pembelajaran yang popular dalam Rangkaian Neural. Ianya telah diimplementasikan dengan jayanya dalam pelbagai aplikasi. Walau bagaimanapun, pembelajaran dalam Rangkaian Neural melibatkan jumlah data yang banyak. Oleh itu, proses latihan rangkaian memerlukan masa yang lama. Hal ini kerana setiap sesi latihan melibatkan beberapa pengulangan yang mengambil masa beberapa saat atau minit. Kertas kerja ini mencadangkan pendekatan berbilang rangkaian bagi meminimumkan kompleksiti rangkaian. Pendekatan ini tidak melibatkan pengubahsuaian terhadap algoritma. Sebaliknya, rangkaian yang besar dipecahkan kepada beberapa rangkaian yang kecil. Rangkaian

penggabung kemudiannya dibangunkan bagi menggabungkan output rangkaian tersebut.

Kata Kunci: *Rangkaian Neural, Rangkaian Rambatan Balik, Rambatan Balik Berbilang Rangkaian.*

INTRODUCTION

Backpropagation (or backprop) algorithm is one of the well-known algorithms in neural networks. Backpropagation algorithm has been popularized by Rumelhart, Hinton, and Williams in 1980s as a euphemism for generalized delta rule. Backpropagation of errors or generalized delta rule is a decent method to minimize the total squared error of the output computed by the net (Fausett, 1994). The introduction of backprop algorithm has overcome the drawbacks of previous Neural Network (NN) algorithms in 1970s, where single layer perceptrons failed to solve a simple XOR problem.

According to Fausett, the aim of backpropagation algorithm is to train the net to achieve a balance between the ability to respond correctly to the input patterns that are used for training (memorization) and the ability to give reasonable (good) responses to input that is similar, but not identical, to that used in training (generalization). Sarle (1997) describes backpropagation as follows;

- Backpropagation refers to the method for computing the gradient of the case-wise error function with respect to the weights for a feedforward network.
- Backpropagation refers to a training method that uses backpropagation to compute the gradient.
- A backpropagation network is a feedforward network trained by backpropagation.

Typically there are two methods of training, namely batch training and incremental training. In batch training, the weights are updated after processing the entire training set. On the other hand, the weights are updated after processing each case in incremental training. According to Sarle (1997) standard backpropagation usually converges (eventually) to a local minimum. For incremental training, standard backpropagation does not converge to a stationary point of the error

surface. Therefore to obtain convergence, the learning rate must be slowly reduced. In addition, study shows the existence of a relationship between gain, learning rate and weights in backpropagation networks (Thimm *et al.*, 1996). This is followed by the implications of this relationship for variations of the backpropagation algorithm.

However, a large volume of data is involved in training the network, so NN can be too complex and difficult to train. Therefore, this paper presents a multi-network approach to minimize the NN complexity. Using this approach, more data or rules could be inserted into the system without affecting its performance.

MULTI-BACKPROPAGATION FRAMEWORK

Backpropagation network is able to deal with various types of data, and model a complex decision system. Backpropagation network with hidden layer (or so called multi layer network) is able to process and model more complex problems. However, some problem domains might involve a large amount of data. Backpropagation network with four input units and two hidden units, for example, require several epochs, which create a complex model. More input units or hidden units could increase the complexity of the model and increase its computational complexity. In other words, an addition to the input unit or hidden unit could increase the model complexity and increase training time. This is because a larger network is more difficult to train. Like human learning, a complex problem requires certain period of time to establish learning.

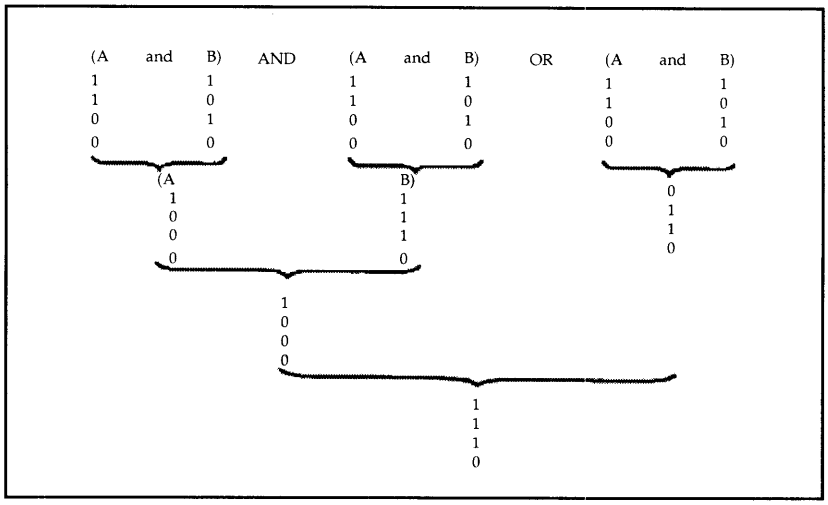
In Figure 1 we illustrate the problem of multiple logical operation **AND**, **OR** and **XOR** that is (*A* **AND** *B*) **AND** (*C* **OR** *D*) **OR** (*E* **XOR** *F*) as the theoretical framework of the multi-network approach. In NN this problem is presented as a set of inputs into the network. The relation between each input is considered to be understood by the network. As we have six inputs, the total combination would be 64. Hence, training the network to learn all 64 data sets is time consuming where for each epoch the nets have to learn 64 different patterns. Each pattern is fed into the network one at a time and its error information term is calculated.

Basically this technique combines several operations and some of the operations are repeated. For example, (*A* **AND** *B*) and (*A* **AND** *B*) **AND** (*C* **OR** *D*) are an **AND** problem. Solving both problems require

logical sets of **AND**. Manually (based on the logical tables) this problem is solved step by step as in Figure 1. The output from (*A* **AND** *B*) and (*C* **OR** *D*) is **AND**'ed together to obtain its output. Thereafter, its output is **OR**'ed with the output of (*E* **XOR** *F*). The OR and XOR could be solved using the logical set of OR and XOR.

Hence (*A* **AND** *B*) **AND** (*C* **OR** *D*) **OR** (*E* **XOR** *F*) can be divided into three logical networks, that are **AND**, **OR** and **XOR** networks. These three networks will produce a knowledge of **AND**, **OR** and **XOR** logical operation (which also represents its logical table). Each network has four sets of data that are [1,1,*t*], [1,0,*t*], [0,1,*t*], [0,0,*t*] where *t* is the targeted value. Training these networks require only several epochs. All three networks will be trained one by one and their weight or knowledge will be stored as the representation of logical operations. Knowledge from **AND** network for example can be used for both (*A* **AND** *B*) and (*A* **AND** *B*) **AND** (*C* **OR** *D*) operation.

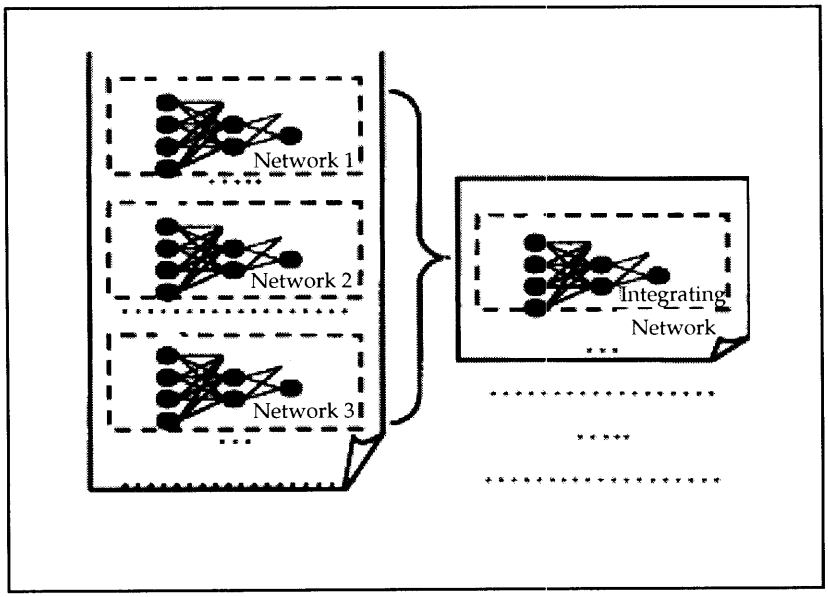
Figure 1
The Structure of (A AND B) AND
(C OR D) OR (E XOR F) Problem



This approach reduces the total number of data used in the training. For example, the original data sets consist of 64 data sets, and they have been reduced to only 12 data sets. In addition, the number of variables for the network is also reduced from six variables to two variables. Reducing the variables and the data sets can reduce the network complexity.

Minimizing the complexity means reducing the complexity of each pattern by normalizing its attributes. Normalization referred to in this study is the same as that applied in relational databases where attributes are grouped into several categories to minimize the relationship between attributes. This technique could reduce the redundancy of data. Originally, the idea of multi backpropagation network is similar to the concept of bottom-up hierarchical neural network (see for example Ohno-Machado, 1996). Several specialized networks are constructed to represent a certain component of the problem and another network integrates the outputs to produce the final result (Figure 2).

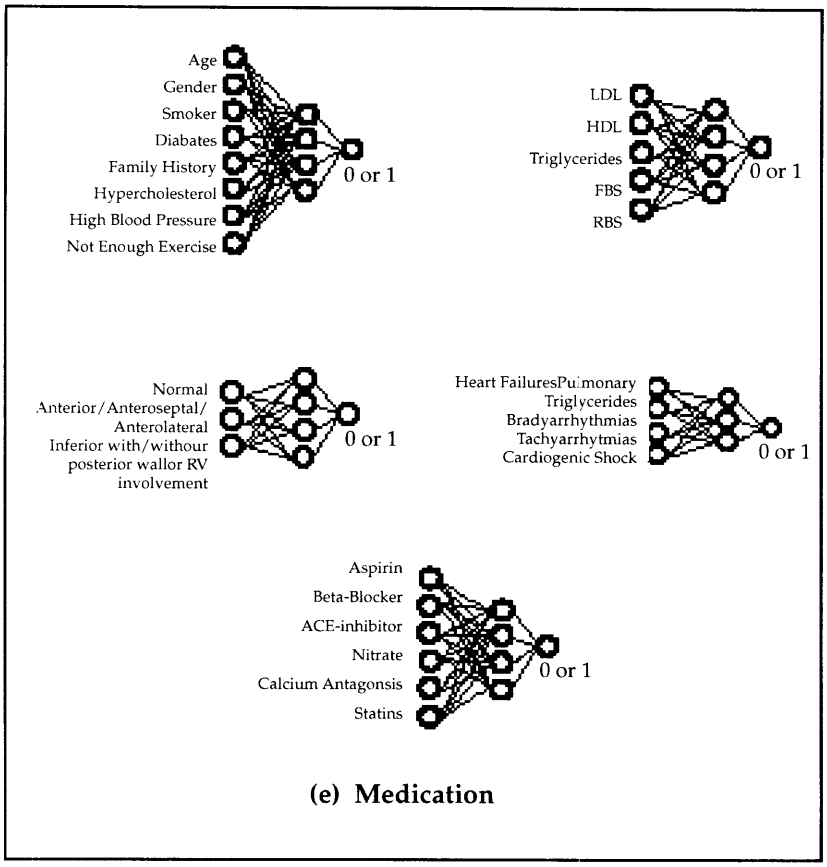
Figure 2
Framework for Multi-Backpropagation Representation



EXPERIMENT AND DISCUSSION

This study uses the Myocardial Infarction problem as the research domain. The Myocardial Infarction problem consists of 26 variables. Each variable is represented in Boolean, either true or false. In this study, the variables are divided into five different groups, they are **COMPLICATIONS**, **ECG**, **INVESTIGATION**, **MEDICATION** and **RISK FACTORS** group (Figure 3).

Figure 3
Networks by Category



Each group produces an output. For example, from the number of the risk factors or its combination, a medical practitioner could easily classify patient's risk status, i.e., whether that patient is high-risk of having Myocardial Infraction (MI) or not. This interpretation of results is either 0 or 1 based on the interpretation of the condition given. These categories are then grouped into one network to produce its final result (see Figure 4).

Compared to the smaller networks, the original network a has a large connection link between input, hidden and output layer (Figure 5). The network needs to be trained for more epochs so that the network can learn all the patterns. The larger network certainly requires more time and epochs to learn.

Figure 4
Integrating Networks

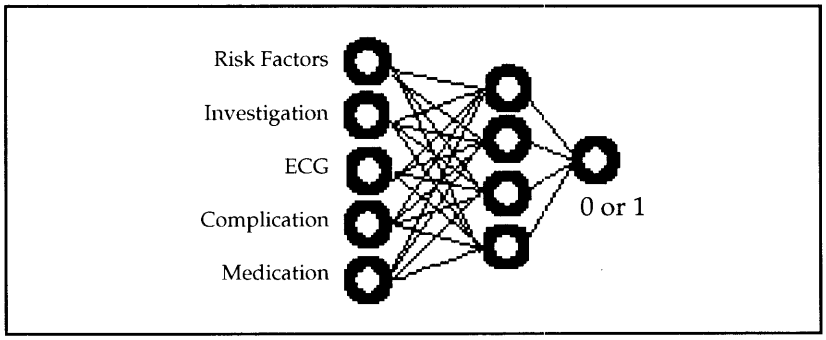


Figure 5
Predicting the Presence of Myocardial Infarction

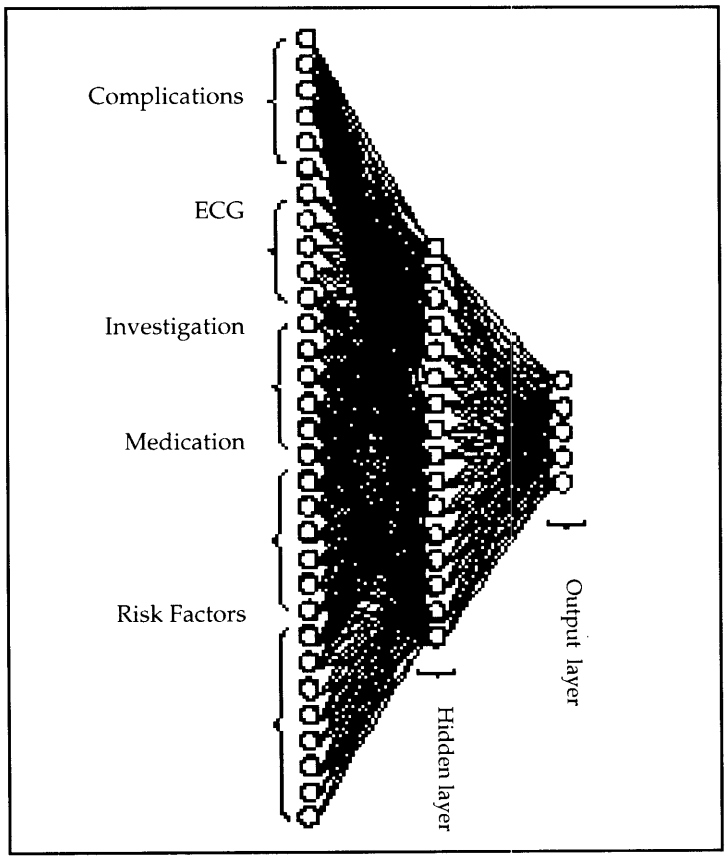


Table 1 shows the training results after training ten times. On average, the time taken for each training session to complete the learning tasks is 115,421 milliseconds with 40 epochs.

Table 1
Results for 7,466 Data (Set C) After Training 10 Times

Training	Epochs	Time (Ms)	Results	MSE
1	40	129240	100	0.004892199
2	40	106610	100	0.004892199
3	40	114900	100	0.004892199
4	40	108690	100	0.004892199
5	40	114740	100	0.004892199
6	40	115240	100	0.004892199
7	40	123420	100	0.004892199
8	40	122930	100	0.004892199
9	40	109580	100	0.004892199
10	40	108860	100	0.004892199
Average	**40**	**115,421**	**100**	**0.004892199**

Hence, it is estimated (using equation 1) that if 7,466 data takes 115,421 milliseconds to train then 67,108,864 of data takes approximately 1,037,472,836 milliseconds to complete the learning.

$$\text{Total time} = \frac{\text{Time Taken}}{\text{No. of Data Trained}} * \text{Total No. of Data} \quad (1)$$

The total epoch taken by the whole population to complete the learning is estimated as 359,544 epochs (using equation 2).

$$\text{Total epoch} = \frac{\text{No. of Epoch Taken}}{\text{No. of Data Trained}} * \text{Total No. of Data} \quad (2)$$

Table 2 shows the results after training using the multi-network approach. Each network trained for 10 times to take the time and epoch average. The results show that each network takes on average 175.833 milliseconds to complete the learning and on average 7.66667 epochs for each network to learn. In total, the networks take 1055 milliseconds and 46 epochs to generalize completely.

Table 2
Results Average

Network	Time (Ms)	Epochs	Results
Risk Factor	281	2	100
Medication	197	5	100
Investigation	32	1	100
ECG	440	34	100
Complication	83	3	100
Integrating network	22	1	100
Average	**175.833**	**7.66667**	**100**
Total	**1055**	**46**	**600**

CONCLUSION

The original data sets are too large and too complex to learn. Training the network using the single network approach does not cover all data sets. Even though the network generalizes approximately in two 115,421 milliseconds and with 40 epochs, it only represents a small portion of the whole data sets. Other data is not involved in training and the network may not recognize some of these patterns. In addition, the single network approach is estimated to take approximately 1,037,472,836 milliseconds and 359,544 epochs to generalize.

In the multi-network approach, the large network is divided into several smaller networks. Each network is trained separately. Another network called the integrated network was constructed to compile smaller networks into one network. Although many networks had to be constructed and trained separately, the multi-network approach has reduced the complexity of the network with large data sets and overcome the limitation of a single network approach. This is because the networks represent all possible combinations of data and train them respectively. In other words, in the multi-network approach, all data sets are used in training. The knowledge produced by the network can be applied for all possible data sets. The experiment reveals that the multi-network approach takes an average 175.833 milliseconds to complete the learning. In total, 1055 milliseconds and 46 epochs were taken for all networks to generalize.

ACKNOWLEDGEMENT

The authors would like to thank the Ministry of Health for allowing the access and use of Myocardial Infraction data from General Hospital Alor Setar, Kedah, Malaysia.

REFERENCES

Fausett, L. (1994). *Fundamentals of Neural Network: Architectures, Algorithms and Applications.* Englewood Cliffs: Prentice Hall.

Sarle, W.S. (1997). Neural network FAQ, part 2 of 7: learning. *Periodic posting to the Usenet newsgroup comp.ai.neural-nets.* Retrieved March 28, 2002 from World Wide Web: ftp://ftp.sas.com/pub/neural/FAQ.html.

Thimm, G., Moerland, P. & Fiesler, E. (1996). The interchangebility of learning rate and gain in backpropagation neural networks. *Neural Computation*, 8(2).

Ohno-Machado, L. (1996). *Medical applications of artificial neural networks: connectionist model of survival.* Unpublished Ph.D. Dissertation. Stanford University.