

# **THE TUNING OF ERROR SIGNAL FOR BACK-PROPAGATION ALGORITHMS**

RENUGAH A/P RENGASAMY

A thesis submitted in partial fulfillment of the requirements for the award  
of the degree of Master of Science (Computer Science)

Faculty of Computer Science and Information System  
Universiti Teknologi Malaysia

OCTOBER, 2008

*In memory of my father, M.Rengasamy*

*To my mother, S.Sarasvathy,  
for being my source of inspiration...*

*And my only sibling, R.Gobiraj  
for being my critic...*

*Thanks for being there & no words could  
describe your support & scarifications...*

*I Love you...*

## ACKNOWLEDGEMENT

I would like to take this opportunity to express my gratitude to each and everyone who have support me to make this project a success.

First of all, I would like to express my deep gratitude to my lecturer cum supervisor, Associate Professor Dr. Siti Mariyam bt Shamsuddin for her invaluable guidance and encouragement throughout my studies and towards my completion of this study. Her timeless patience, assistance and concern in understanding my responsibilities have been a key to make this study a success. Without her, I would not be able to complete in time as I have been facing a tortuous path to the completion of this study.

Besides that, I also would like to thank my panel of evaluators, Associate Professor Dr. Ali bin Selamat and Associate Professor Dr. Manan for their helpful suggestions.

And I would to express my appreciation for those who were involved directly or indirectly in supporting me towards the completion of this study. Thank you all for always encouraging and believing me. I really owe them where my words alone are not worthy for what they have done for me.

May you all are showered with HIS blessings and grace. Thank you.

## ABSTRACT

Despite of Back-propagation (BP) algorithm existence for almost four decades, it is still widely used in many fields to solve range of real world problems. However, it suffers from slow convergence and tends to trap in local minima. So, an improved two-term error function is proposed to overcome the existing problems. This new algorithm is proven to be a better algorithm. The main purpose of this study is to evaluate the efficiency of improved two-term error function by applying three different values of  $\beta$  parameter in the activation function. The improved two-term error with different error signal ( $\delta$ ) will replace the conventional error signal in standard BP. These both algorithms will be tested on three universal datasets; Iris, Balloon and Cancer by comparing the accuracy and the convergence speed. The ultimate outcome of the study would be handful information to get a better justification on these both Back-propagation algorithms usages in real application.

## ABSTRAK

Walaupun algoritma rambatan-balik (BP) sudah wujud hampir empat dekad, namun ianya masih lagi digunakan untuk menyelesaikan pelbagai masalah industri. Namun begitu, kelemahan utama algoritma ini adalah kadar penumpuan yang lambat dan mudah terperangkap di dalam minima setempat. Maka, satu kaedah iaitu pembaikan fungsi rambatan-balik dua terma dengan pembaikan ralat fungsi yang baru telah disyorkan untuk mengatasi masalah ini. Kaedah ini sememangnya dapat mengatasi kelemahan algoritma yang asal. Tujuan utama kajian ini adalah untuk menguji kadar penumpuan yang boleh dicapai oleh fungsi rambatan-balik dua terma dengan menggunakan tiga nilai  $\beta$  yang berbeza terhadap fungsi keaktifan. Kaedah pembaikan ini akan menggunakan penanda ralat yang berbeza ( $\delta$ ) daripada kaedah yang biasa yang biasa digunakan dalam BP. Kedua-dua algoritma ini akan diuji dengan menggunakan tiga jenis data universal iaitu Iris, Balloon dan Cancer bagi melihat kadar penumpuan terhadap ralat yang dicapai. Diharapkan hasil akhir projek ini dapat membantu dalam membuat justifikasi yang lebih tepat dalam penggunaan kedua-dua algoritma di dalam aplikasi sebenar.

## TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	<b>TITLE PAGE</b>	<b>i</b>
	<b>DECLARATION</b>	<b>ii</b>
	<b>DEDICATION</b>	<b>iii</b>
	<b>ACKNOWLEDGEMENT</b>	<b>iv</b>
	<b>ABSTRACT</b>	<b>v</b>
	<b>ABSTRAK</b>	<b>vi</b>
	<b>TABLE OF CONTENTS</b>	<b>vii</b>
	<b>LIST OF TABLES</b>	<b>x</b>
	<b>LIST OF FIGURES</b>	<b>xii</b>
	<b>LIST OF SYMBOLS</b>	<b>xiii</b>
	<b>LIST OF ABBREVIATIONS</b>	<b>xiv</b>
<b>1</b>	<b>INTRODUCTION</b>	
	1.1 Introduction	1
	1.2 Problem Statement	4
	1.3 Project Aim	4
	1.4 Objectives	5
	1.5 Project Scope	5
	1.6 Significance of the Project	6
	1.7 Organization of the Report	6

<b>2</b>	<b>LITERATURE REVIEW</b>	
2.1	Introduction	7
2.2	Neural Network	7
2.2.1	The Neural Network History	8
2.2.2	The Neural Network Capabilities	9
2.2.3	The Neuron	11
2.2.4	The Diagram of Neuron	12
2.2.5	The Neural Network Learning	12
2.2.6	The Activation Function	14
2.2.7	The Architecture	15
2.3	Multilayer FeedForward Network	16
2.4	Backpropagation Algorithms	17
2.4.1	Mathematical Structure of Standard BP	17
2.5	Error Function	20
2.5.1	Mean Squared Error Function	21
2.5.2	Two-term Improved Error Function	25
2.6	The Logistic (Sigmoid) Activation Function	28
2.7	Related Research Work on ANN	29
2.8	Summary	30
<b>3</b>	<b>METHODOLOGY</b>	
3.1	Introduction	31
3.2	BP Training	31
3.3	The Dataset	34
3.4	Implementation of Neural Network	35
3.4.1	Define BPNN Architecture and Parameters	36
3.4.2	Formulation of Weight Adjustment	37
3.4.3	Define the Learning Rate and Momentum Term	38
3.4.4	Define the Maximum Error	38
3.5	Workflow	39
3.6	Experiment and Analysis	39
3.7	Summary	41

<b>4</b>	<b>EXPERIMENTAL RESULTS AND ANALYSIS</b>	
4.1	Introduction	42
4.2	Algorithm Implementation	43
4.3	Experimental Results	43
4.4	Result of Iris Dataset	44
4.5	Result of Balloon Dataset	48
4.6	Result of Cancer Dataset	53
4.7	Discussion	59
4.8	Summary	61
<b>5</b>	<b>CONCLUSION</b>	
5.1	Introduction	62
5.2	Summary of Works	63
5.3	Conclusion	64
5.4	Recommendation for Future Study	65
	<b>REFERENCES</b>	66

## LIST OF TABLES

TABLE NUM.	TITLE	PAGE
2.1	Summary of early ANN research	8
2.2	Common Activation Function	14
2.3	Summary of Related Research Work on ANN	29
3.1	The BP Training Steps	32
3.2	Weight Updating Strategies	33
3.3	Batch Training Algorithm	33
3.4	Summary of Attribute Values for Balloon Dataset	35
3.5	The Network Architecture of Iris, Balloon and Cancer Dataset	37
3.6	Implementation Description	39
3.7	The Parameter of Algorithm Experiment	40
4.1 (a)	Result of Standard BP for Iris Dataset	44
4.1 (b)	Result of IeBP ( $\beta = 2$ ) for Iris Dataset	44
4.1 (c)	Result of IeBP ( $\beta = 3$ ) for Iris Dataset	45
4.1 (d)	Result of IeBP ( $\beta = 4$ ) for Iris Dataset	45
4.2 (a)	Result of Standard BP for Balloon Dataset	49
4.2 (b)	Result of IeBP ( $\beta = 2$ ) for Balloon Dataset	49
4.2 (c)	Result of IeBP ( $\beta = 3$ ) for Balloon Dataset	49
4.2 (d)	Result of IeBP ( $\beta = 4$ ) for Balloon Dataset	50
4.3 (a)	Result of Standard BP for Cancer Dataset	54
4.3 (b)	Result of IeBP ( $\beta = 2$ ) for Cancer Dataset	54
4.3 (c)	Result of IeBP ( $\beta = 3$ ) for Cancer Dataset	54

4.3 (d)	Result of IeBP ( $\beta = 4$ ) for Cancer Dataset	55
4.4	Summary of Classification	59
4.5	Summary of Results	60

## LIST OF FIGURES

FIGURE NUM.	TITLE	PAGE
2.1	The Diagram of Neuron	12
2.2	Classification of NN Learning	13
2.3	Single Layer Perceptron	15
2.4	Multi Layer Perceptron	16
2.5	Architecture graph of a multilayer feedforward network	17
3.1	The Standard BP Flowchart	32
3.2	Implementation Process	39
3.3	An Overview of the Experiment Process	36
4.1	Convergence of Iris Dataset for Standard BP and IeBP ( $\beta = 2,3,4$ )	46
4.2	Classification Accuracy of Iris Dataset	47
4.3	Convergence Characteristic of V5 Trial	47
4.4	Correct Classification of V5 Trial	48
4.5	Convergence of Balloon Dataset for Standard BP and IeBP ( $\beta = 2,3,4$ )	51
4.6	Classification Accuracy of Balloon Dataset	51
4.7	Convergence Characteristic of V5 Trial	52
4.8	Correct Classification of V5 Trial	53
4.9	Convergence of Cancer Dataset for Standard BP and IeBP ( $\beta = 2,3,4$ )	56
4.10	Classification Accuracy of Cancer Dataset	56
4.11	Convergence Characteristic of V2 Trial	57
4.12	Correct Classification of V2 Trial	58
4.13	Correct Classification for Iris, Balloon and Cancer Dataset	59

## LIST OF SYMBOLS

$W_{ij}$	- Weight connected between node $i$ and $j$
$\theta_i$	- Bias of node $i$
$a_i$	- Output of node $i$
$O_j$	- Output of node $j$
$W_{ij}(t)$	- Weight from node $i$ to node $j$ at time $t$ ,
$\Delta W_{ij}$	- Weight adjustment
$\eta$	- Learning rate
$\delta_j$	- Local gradient at node $j$
$T_j$	- Target output value at node $j$
$\alpha$	- Momentum term
$a_k$	- Activation of unit $k$
$e1$	- Mean square error at first iteration
$e2$	- Mean square error at <i>itstop</i>
$E_k$	- Error at output unit $k$
$m$	- Number of input nodes
$n$	- Number of output nodes
$o_{kj}$	- Network value from output node ( $i$ ) to hidden node ( $j$ )
$t_{kj}$	- Target value from output node ( $i$ ) to hidden node ( $j$ )

**LIST OF ABBREVIATIONS**

ANN	-	Artificial neural network
BP	-	Backpropagation
BPNN	-	Backpropagation neural network
IeBP	-	Improved two-term Backpropagation
MFNN	-	Multilayer feedforward network
MLP	-	Multilayer feedforward perceptron
MSE	-	Mean square error
NN	-	Neural Network

## **CHAPTER 1**

### **INTRODUCTION**

#### **1.1 Introduction**

An Artificial Neural Network or also known as ANN is a network of many very simple processors ("units"), each possibly having a (small amount of) local memory. The units are connected by unidirectional communication channels ("connections"), which carry numeric ("weights") data. The units operate only on their local data and on the inputs they receive via the connections.

A lot of researches have been carried out regarding the theory and the application of ANN. ANN has been a dominant tool for solving various problems such as pattern classification and recognition [1], medical imaging [2], speech recognition [3][4] and control [5]. ANN can be characterized by its architectures, learning algorithms and the activation function as well [6]. ANN learns by examples through learning algorithms, which adjusts the connection weights iteratively until the desired result, typically expressed by minimization of an error function, is achieved [7]. This design motivation is what distinguishes neural networks from other mathematical techniques. The most commonly used learning algorithms for training ANN is the back-propagation algorithm (BP) [8].

The Multilayer Perceptron (MLP) with BP learning algorithm is found to be effective for solving a number of real world problems. However, it suffers from slow convergence process or long training time and tends to stuck in local minima. Many researchers [9][10][11][12][13][14][15][16] have been proposed methods for accelerating the learning of BP. These studies shows that the BP learning ability is affected by many factors like learning structure, initial weight, error function, learning parameters and also the activation function. Besides, these methods try to modify the learning model and overcome the tendency to sink into local minima by adding a random factor to the model. However, the random perturbations of the search direction and various kinds of stochastic adjustments to the current set of weights are not effective at enabling a network to escape from local minima within a reasonable number of iterations [17].

Many researches have been trying to overcome and improve the efficiency and convergence rate. In [6][18], this can be done by the selection of better cost function, dynamics variation of learning rate and momentum and also the selection of better activation function of the neurons. Another study by [11] have summarized these approaches into seven cases: the weight updating procedure, the choice of optimization criterion, the use of adaptive parameters, estimation of optimal initial conditions, reduces the size of problem, estimation of optimal ANN structure and the application of more advanced algorithms.

In order to archive a better NN learning, a more sophisticated error measure can be used. A comparative study on the impact of various error functions in multilayer feed forward has been used for classification problem by [19]. This result indicates that the error function other than mean square error (MSE) gives a better performance of the trained network. Another study by comparing the Bernoulli error measure with the popular MSE measure has been done by [20]. Significant increase in training speed has been obtained by using the Bernoulli error measure than the MSE. A modified error function in forecasting daily maximum load demand with two activation functions, sigmoid and arctangent has been done by [21]. This study shows that the improved error with sigmoid function is much faster than compared to

the improved error with arctangent function. Basically, the convergence of the NN depends on parameters such as learning rate, momentum term, slope of the activation function and many more.

However, these NN trained with gradient descent algorithms such as backpropagation often converge very slowly for a variety of reasons. Slow convergence might be due to very small learning rate, or due to incorrect architecture with too few layers of weights (or too many), or too few hidden neurons. The  $\beta$ , slope of the activation function also affects the convergence speed. If the slope is very high then certain neurons will saturate. Saturation occurs if the output is pushed towards its extremes at some points before convergence is reached, so that the derivative is too small (at those points) to make further significant weight changes, causing the network to settle in an incorrect local minimum or reach a state of network paralysis [22]. On the other hand, if the slope is very low then the neuron will behave like a linear function,  $f(x) \approx 2$ . Though the network will still learn but it will fail in making fine decision boundaries. For a given high value of slope, the network can be protected from being paralyzed by keeping the magnitude of initial weight range and learning rate small.

From all studies and researches in order to improve the BP learning, it shows that more advanced algorithms can be experimented and compared with the original standard BP in order to speed up the learning convergence. Study by [6], proves that the two-term improved BP (IeBP) of [10] could give a better result in term of convergence rate, accuracy and able to escape from local minima. However, the tuning of error signal or the slope of the sigmoid activation function has never been done experimented and tested. Thus, this study aims to evaluate the performance of NN by determining optimum values of these slope values with 2, 3 and 4.

## 1.2 Problem Statement

BP is the most widely used learning algorithm in solving several of real world problems yet it still suffers from the slow convergence rate. As stated before, the NN training process is actually time consuming. In order to minimize the execution time, we need to increase the computational speed. These could be achieved by doing some modification to the BP algorithm. Modified error function has been proposed in [10] which have proved that the execution time of the IeBP is faster than the standard BP.

The IeBP, which introduced in [10], has been tested using various datasets. Therefore, IeBP which emphasis on modification of slope ( $\beta$ ) parameter using three different values (2, 3 and 4) will be implemented to seek for better results. Besides that, the standard BP and IeBP with three different slope values will be implemented and used to compare the performance of these both algorithms. The hypothesis of the study can be stated as:

*The  $\beta$  parameter has the significant effect on the implementation of Improved two-term BP to perform better in the classification problems.*

## 1.3 Project Aim

This study aims to determine the efficiency of standard BP and IeBP with the some tuning done on the slope parameter, ( $\beta$ ) for classification problems. Learning rate and momentum term with sigmoid activation function with three layers of NN architecture is used for BP training. In addition, this study also investigates in detail the effect of the  $\beta$  parameter in the IeBP to the network performance and

convergence during training. Three classification problems; the Iris, Balloon and Cancer are used to examine whether the various sizes of classification data affect the performance between modified error function measure and the mean square error measure.

#### 1.4 Objectives

In order to accomplish the hypothesis of the study, few objectives have been identified as stated below:

1. To develop the standard BP and IeBP ( $\beta = 2,3,4$ ) algorithms.
2. To find the efficiency of these four BP algorithms.
3. To evaluate and compare the performance of the standard BP and IeBP ( $\beta = 2,3,4$ ) algorithms.

#### 1.5 Project Scope

The main focus of this study is on the improved two-term BP error function by looking into the slope parameter tuning. The scopes of this study are as follows:

1. Three sets of universal datasets (Iris, Balloon and Cancer) are used.
2. The MSE function and the improved error function [10] are used.
3. Sigmoid function is used as the activation function for the BP network.
4. The value of slope parameter used in this testing are,  $\beta=2, 3$  and 4.
5. The value of learning rate and momentum term are set to 0.1, 0.5 and 0.9.

## 1.6 Significance of the Project

The study investigates the performance of BP learning, in terms of accuracy and convergence rate for the classification problems. The utilization of error function in IeBP is tested against the same datasets as standard BP. The results of this study will contribute to effectiveness of the  $\beta$ , slope parameter in activation function that have been used for BPNN training.

## 1.7 Organization of the Project

This report consists of five chapters. The first chapter, 'Introduction' briefly elaborates the problem being tackled, with the goals and the scope of the research. The second chapter "Literature Review" explains about appropriate literature and review on NN in classification problems, artificial neural network, BP algorithm and error functions. Some of the related previous research works also discussed in this chapter. Third chapter, "Methodology", discusses about the methodology used in this project, which explains in detail the implementation of standard BP and IeBP. Next, looking into the fourth chapter, we have the "Experimental Result" which is the analysis and findings. Lastly, we have the "Conclusion" that concludes the whole work at the same time stating its contributions and suggesting future research.

## **CHAPTER 2**

### **LITERATURE REVIEW**

#### **2.1 Introduction**

Neural network (NN) represents a technology that is rooted in many disciplines such as neurosciences, mathematics, statistics, physics, computer science and engineering [23]. The strong interest in NN in the scientific community is fueled by the many successful and promising applications such as pattern and speech recognition, signal processing, control problems and financial modeling.

#### **2.2 Artificial Neural Network**

Artificial Neural Network (ANN) is a collection of mathematical models that emulate some of the properties of biological nervous system. It is modeled from the human brain and it has artificial neuron that consist of an activation function, connection strength and activation thresholds. ANN consists of a set of processing elements (also known as neurons or nodes) which are interconnected to each other. Each node has an activation function, a function of the inputs it has received. In

general, these nodes will send its activation as a signal to several other nodes. This signal would travel through weighted connections between nodes. Each of these nodes accumulates the inputs it receives, producing an output according to an internal activation function [14].

### 2.2.1 Neural Network History

Research of NN has been going on for more than 40 years. It is all started with the pioneer work of McCulloch and Pitts in 1943 [23]. Ever since then, a lot of researches have been carried out continuously in various field of interest. There were some major research accomplishments which contributed to present ANN algorithms. Table 2.1 below lists some of the early contributions [23].

**Table 2.1: Summary of early ANN research**

Author, Year	Description
McCulloch & Pitts, 1943	Presented first abstract models of neurons
Hebb, 1949	Proposed a learning rule which allow the adjustment of the synaptic weights
Rosenblatt, 1958	Developed the original concept of <i>perceptron</i>
Widrow & Hoff, 1960	The Adaline ( <i>adaptive linear element</i> ) trained by the least-mean-square (LMS) learning rule
Minsky & Papert, 1969	Pointed out the thereotical limitations of single layer NN model
Werbos, 1974	First description of BP algorithm for training MLFF perceptron
Kohonen, 1982	Presented the self-organizing feature map
Parker, 1985 LeCun, 1985	Independent discoveries of the BP training algorithm
Rumelhart, Hinton, Williams, 1986	Developed the BP learning algorithm

### 2.2.2 The Neural Network Capabilities

The use of NN offers many useful properties and capabilities [7] as below:

#### a) Nonlinearity

NN can be either linear or nonlinear. The nonlinear feature is a special kind of sense that the NN is distributed throughout the network. Nonlinearity is a highly important property, particularly if the underlying physical mechanism responsible for generation of the input signal (which is inherently nonlinear).

#### b) Input-Output Mapping

Supervised learning involves synaptic weights modification of a NN by applying a set of labeled training samples or task samples. Each example consists of a unique input signal and a corresponding desired response. The network is presented with an example picked at random from the set, and the synaptic weights of the network are modified to minimize the difference between the desired response and the actual response of the network produced by the input signal in accordance with an appropriate statistical criterion. The training of the network is repeated for many times until the network reaches a steady state; where there are no significant changes in the synaptic weights. The previously applied training examples may be reapplied during the training session but in different order. Thus the network learns from the examples by constructing an input-output mapping for the problem at hand. The input-output mapping feature is very useful in the pattern classification task.

#### c) Adaptivity

NN has the ability to adapt their synaptic weights to changes in the surrounding environment. A NN, which is trained to operate in a specific environment, can be easily retrained to deal with minor changes in the operating environmental conditions. The adaptive capability of the NN makes it a useful tool in pattern classification, signal processing and control applications.

**d) Evidential Response**

In the context of pattern classification, a NN can be designed to provide information not only about which particular pattern to select, but also the confidence in the decision made. This latter information may be used to reject ambiguous patterns, should they arise, and thereby improve the classification performance of the network.

**e) Contextual Information**

Knowledge is represented by the very structure and activation states of a NN. Every neuron in the network is potentially affected by the global activity of all other neurons in the network. Consequently, contextual information is dealt with naturally by a NN.

**f) Fault Tolerance**

A NN implemented in hardware form, has the potential to be inherently fault tolerant, or capable of robust computation, in the sense that its performance degrades gracefully under adverse operating conditions. For example, if a neuron or its connecting links are damaged, recall of a stored pattern is impaired in quality. However, due to the distributed nature of information stored in the network, the damage has to be extensive before the overall response of the network is degraded seriously. Thus, in principle, a NN exhibits a graceful degradation in performance rather than catastrophic failure. There is some empirical evidence for robust computation, but usually uncontrolled.

**g) Uniformity of Analysis and Design**

Basically, NN benefits from the universality as information processors. This feature manifests itself in different ways:

- Neurons, in one form or another, represent an ingredient common to all NN.
- This commonality makes it possible to share theories and learning algorithms in different applications of NN.
- Modular networks can be built through a seamless integration of modules.

## h) Neurobiological Analogy

The design of a NN is motivated by the analogy with the brain. Neurobiologists look to (*artificial*) NN as a research tool for the interpretation of neurobiology phenomena. On the other hand, engineers look to neurobiology for new ideas to solve problems more complexes than those based on conventional hard-wired design techniques [24][25][26][27][28][29].

### 2.2.3 The Neuron

Neuron is the very basic information processing unit of an ANN. It consists of a set of links, weights, adding function and activation function. Initially, the set of links are describing the neuron inputs with weights.

$$u = \sum_{j=1}^m W_j X_j \quad (2.1)$$

where,

$W_j$  is weight of neuron j,

$X_j$  is input of neuron j.

Then there is an adder function, which also called as linear combiner for computing the weighted sum of the inputs. In addition it has an activation function for limiting the amplitude of the neuron output.

$$y = \varphi(u + b) \quad (2.2)$$

where,

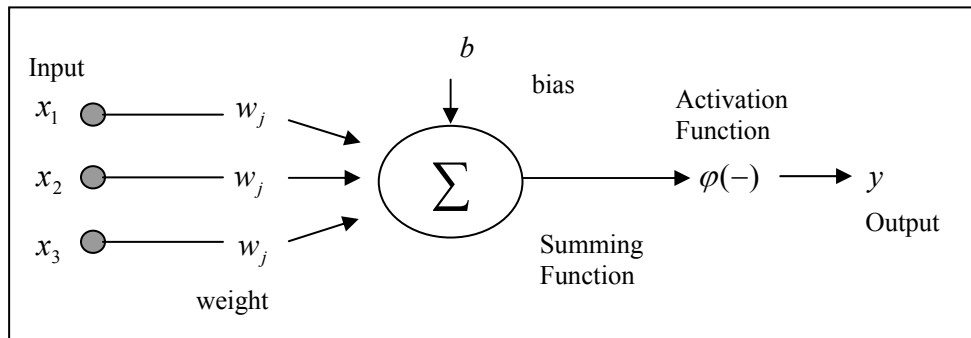
$\varphi$  is activation function,

$u$  is weighted sum,

$b$  is bias.

### 2.2.4 The Diagram of Neuron

Neuron consists of a set of inputs and generates output when some activation function applied to it. The figure 2.1 below shows the basic diagram of neuron.



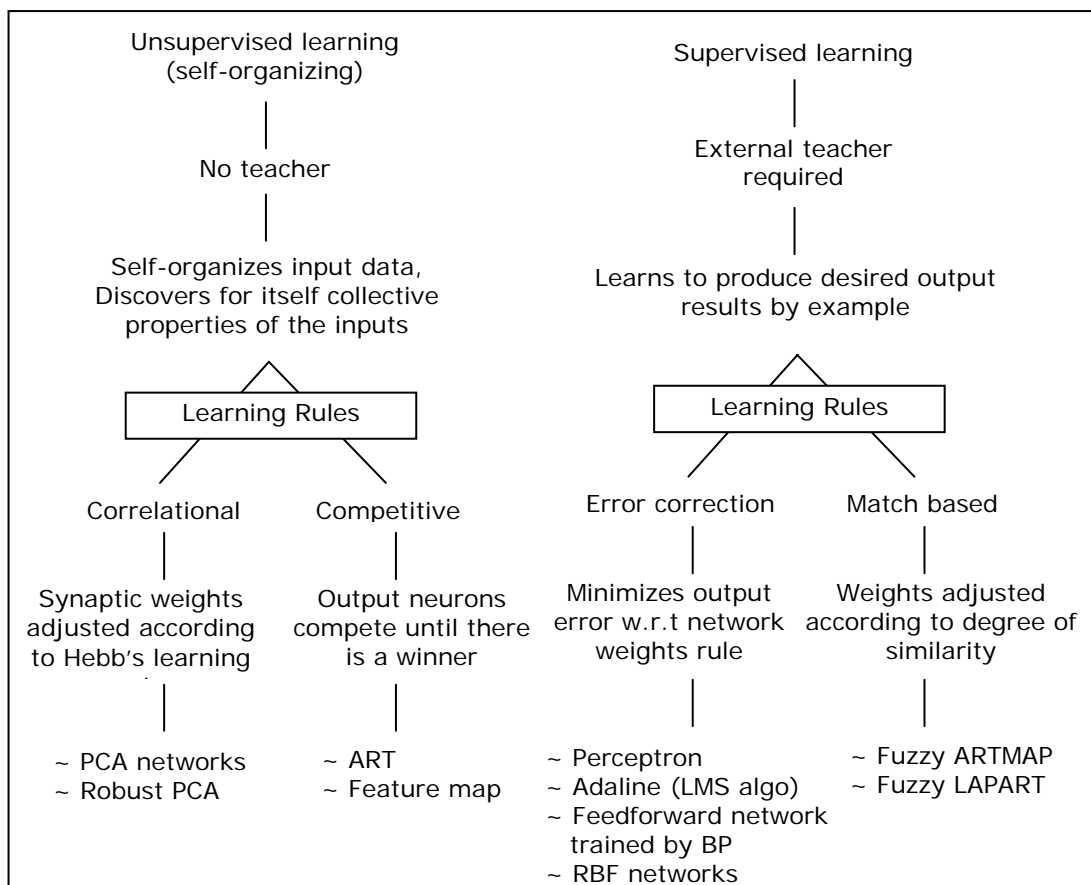
**Figure 2.1: Diagram of Neuron**

### 2.2.5 The Neural Network Learning

ANN consists of a set of processing elements, also known as neurons or nodes, which are interconnected to each other. It can be described as a directed graph in which each node has internal state, called as activation function, a function of the inputs it has received. Typically, a node sends its activation as a signal to several other nodes. This signal travels through weighted connections between nodes. Each of these nodes accumulates the input it receives, producing an output according to an internal activation function.

NN has the ability to learn from its environment and improve its performance through learning. NN can be classified as feedforward, while some other as recurrent, depending on their connectivity. A NN is a feedforward network if an arbitrary input vector is propagated forward through the network and caused an

activation vector to be produced in the output layer. Meanwhile, an ANN is a recurrent network if the output vector is propagated backward to the previous layer. Figure 2.2 below shows some examples of different types of NN and the type of learning (supervised versus unsupervised) [3][23].



**Figure 2.2: Classification of NN Learning**

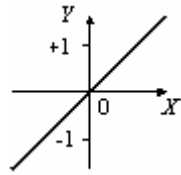
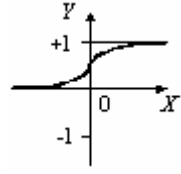
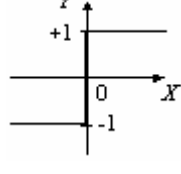
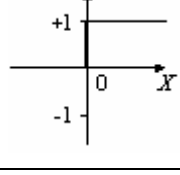
Unsupervised learning is a process of modifying weights without specifying the output of any input patterns which involves search methods. The network itself must then decide what features it will use to group the input data. Supervised learning is a process of adjusting the weights using a learning algorithm; the output for each training input is presented to the network which involves error correction methods. With the both inputs and outputs provided, it's often formulated as the minimization of an error function between the actual output and desired output.

Errors are then propagated back through network, causing the network to adjust the connection weight iteratively in order to minimize the error.

### 2.2.6 The Activation Function

Activation function is nonlinear functions which are applied after the summation of the input weight. The selection of activation function determines the neuron model [3]. There are a number of activation functions available to choose from. For an example, there is the step, sign, sigmoid, linear and so on [23]. The table shows us some of the example that is commonly used:

**Table 2.2: Common Activation Function**

<b>Linear Function</b>	$y^{linear} = x$	
<b>Sigmoid Function</b>	$y^{sigmoid} = \frac{1}{1 + e^{-x}}$	
<b>Sign Function</b>	$y^{sign} = \begin{cases} +1, & \text{if } X \geq 0 \\ -1, & \text{if } X < 0 \end{cases}$	
<b>Step Function</b>	$y^{step} = \begin{cases} 1, & \text{if } X \geq 0 \\ 0, & \text{if } X < 0 \end{cases}$	

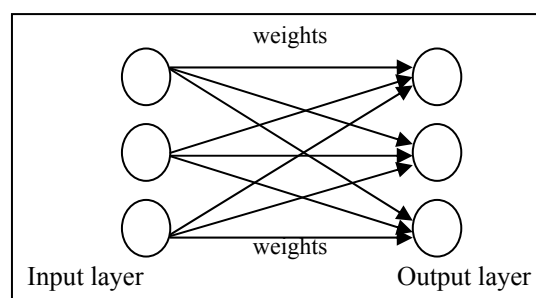
The sigmoid function has an s-shaped graph, is differentiable and is bounded range. The function is used in the multilayer neural networks ‘trained’ by the backpropagation algorithm. An example of sigmoid function is the logistic function defined by:

$$f(a) = \frac{1}{1 + \exp^{-\beta x}} \quad (2.1)$$

where  $\beta$  is the *slope parameter* of the logistic function. The slope parameter,  $\beta$  is important when  $\beta \rightarrow \infty$ , the logistic sigmoid approaches the threshold function. Meanwhile, when  $\beta \rightarrow 0$ , the function has a large nearly linear region in the middle. This study will emphasize on the impact of the  $\beta$  parameter on the convergence rate.

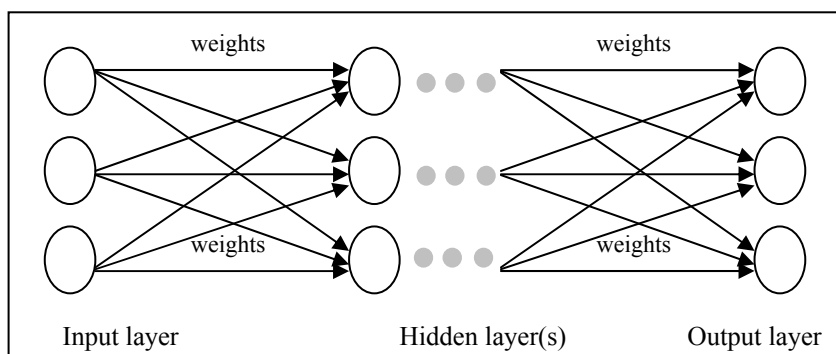
#### 2.2.4 The Architecture

Typical NN architecture can be divided into two types mainly known as single layer perceptron and multi layer perceptron [7]. Single layer perceptron consists of input and output layer which are connected by weights. It uses Least Mean Square (LMS) algorithm. The error value is formed by calculating the difference between the actual output and the targeted output for each layer of nodes. Then, the squares of those errors are minimized by updating the weights. Figure 2.3 below shows the single layer perceptron architecture.



**Figure 2.3: Single Layer Perceptron**

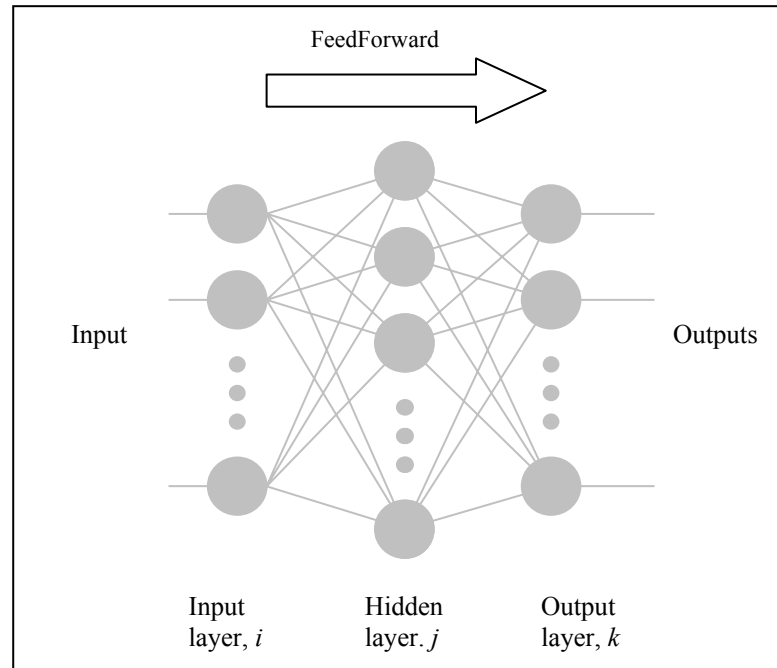
Multilayer perceptron consists of input layer, several hidden layers and an output layer. The input layers are connected by weights to hidden layer(s) and also from hidden layer(s) to the output layer. The error value is calculated using the Mean Squared Error (MSE) formula at hidden layer nodes are calculated by propagating errors backward through the network according to the number of hidden layer nodes. Figure 2.4 below shows the multilayer perceptron architecture.



**Figure 2.4: Multi Layer Perceptron**

### 2.3 Multilayer FeedForward Network

Multilayer FeedForward Network (MFNN) is one of the most popular types of NN. The architecture of MFNN is variable but in general it consists of several layers of neurons. This network model comprises network in which the connections are strictly feedforward. It employs a layer of input nodes, one or more layer of hidden nodes and a layer of output nodes. The architecture of MFNN with one hidden layer is shown in Figure 2.5 below.



**Figure 2.5: Architecture graph of a multilayer feedforward network**

## 2.4 Backpropagation Algorithms

BP algorithm first published by Rumelhart and McClelland in 1986. BP algorithm is a supervised learning method, which it is the most widely used algorithm for training MLP neural network [3]. Generally, training can be done by updating iteratively the weights, by employing the negative gradient of a MSE function. The algorithm works by measuring the output error, calculating the gradient of the error and adjusting the network weight. It is used to minimize the MSE between the actual output computed by the network and the desired output, for all possible input.

### 2.4.1 Mathematical Structure of Standard BP

The structure of BP network comprises of three layers of node, which are input layer, hidden layer. BP learning algorithm has two phases: feedforward phase and backward phase. In the feedforward phase, a training set of input pattern is presented to the network input layer. The network propagates the input pattern from the layer to layer until the output pattern is generated; the output is obtained from a summation of the weighed input of a node and maps to the network activation function. This output is calculated as follows [30]:

$$\begin{aligned} output &= f(net_i) \\ net_i &= \sum_i W_{ij} O_j + \theta_i \end{aligned} \quad (2.2)$$

where,

$W_{ij}$  is the weight connected between node  $i$  and  $j$

$\theta_i$  is the bias of node  $i$ ,

$O_j$  is the output of node  $j$ .

The most common activation function used is sigmoid function [7]. The sigmoid function signal appearing at the output node  $i$  is shown below:

$$f(net_i) = \frac{1}{(1 + e^{-net_i})} \quad (2.3)$$

In second phase, if this output pattern different from desired output, an error is calculated and then propagated backward through the network from output layer to input layer. The weights are modified to reduce the error as the error is propagated. This is the represented as below as in [6][30]:

$$W_{ij}(t+1) = W_{ij}(t) + \Delta W_{ij} \quad (2.4)$$

where,

$W_{ij}(t)$  is the weight from node  $i$  to node  $j$  at time  $t$ ,

$\Delta W_{ij}$  is the weight adjustment

The weight adjustment is computed by using the delta rule:

$$\Delta W_{ij} = \eta \delta_j x_i \quad (2.5)$$

where,

$\eta$  is learning rate ( $0 < \eta < 1$ ),

$\delta_j$  is error at node  $j$

$x_i$  is error at node  $j$

The local gradient  $\delta_j$  is calculated as below:

$$\delta_j = f'(net_j)(T_j - O_j) \quad (2.6)$$

where,

$T_j$  is the target output at node  $j$ ,

$O_j$  is the actual output at node  $j$ .

The standard BP utilizes two parameters: learning rate ( $\eta$ ) and momentum term ( $\alpha$ ). The learning rate is the crucial factoring in considering the size of weights adjustment which done in each iteration and contributes to the convergence rate [13]. It will converge faster when learning rate is large. Though, it might speedup the learning rate, the network will become unstable. Large change in weight causes the search path will fluctuate the ideal path and convergence more slowly than a direct descent. Meanwhile, the descent will progress in a very small steps which significantly will boost the total time to converge if the learning rate is too small [7][13].

According to [13][31], another way to improve the rate of convergence is by adding momentum to the weight adjustment which is done by modifying the delta rule of equation (2.4) as shown below:

$$\Delta W_{ij}(t) = \alpha \Delta W_{ij}(t-1) + \eta \delta_j x_i \quad (2.7)$$

where  $\alpha$  is a positive number ( $0 \leq \alpha \leq 1$ ) and is called as momentum term. Momentum term encourage the movement in the same direction on successive steps. It could help in smoothing out the descent path by preventing the extreme change in weight. Without momentum, the network may get stuck in local minima [13].

## 2.5 Error Function

BP algorithm is a gradient descent procedure which is used to minimize the error function by updating iteratively the connection weights. However, it suffers from slow learning speed and tends to get trapped in local minima [9][10][11][12][13]. Several number of alternative error functions have been introduced to overcome this problem.

To accelerate the learning speed of BP algorithm, [9] proposed a modified function that reduces the probability that output nodes are near the wrong extreme value of sigmoid activation function as well as prevents the overspecialization during learning. A new improved two-term error function has been introduced in [10] which resulted a better convergence outcome than the usual MSE function. This new proposed improved error function is designed specially for classification problem. Therefore, the MSE error function and the improved two-term error function will be used and to get the comparison of both implementation in term of convergence rate and accuracy for classification problem.

### 2.5.1 Mean Squared Error Function

Mean Squared Error Function or also known as MSE function is the most commonly error function used in BP learning [9]. The error is calculated using the gradient descent method. It is used to minimize the MSE between the actual output computed by the network and the desired output, for all possible input. The MSE function is defined as [9]:

$$E = \frac{1}{2} \sum_k (t_{kj} - o_{kj})^2 \quad (2.8)$$

where,

$t_{kj}$  is the target value from output node ( $k$ ) to hidden node ( $j$ )

$o_{kj}$  is the network value from output node ( $k$ ) to hidden node ( $j$ )

The derivation of the error function is one of the factors in weight updating equations. The derivative formulation of the error function to weights is as below:

The weight updating between output layer ( $k$ ) and hidden layer ( $j$ ) in gradient descent of standard BP uses the following delta rule:

$$\Delta W_{kj} = -\eta \frac{\partial E}{\partial W_{kj}} \quad (2.9)$$

where  $\eta$  is the learning rate. By chain rule, the above equation can be rewritten as:

$$\frac{\partial E}{\partial W_{kj}} = \frac{\partial E}{\partial net_k} \times \frac{\partial net_k}{\partial W_{kj}} \quad (2.10)$$

Let,

$$\frac{\partial E}{\partial net_k} = \delta_k \quad (2.11)$$

Since  $net_k = \sum_k W_{kj} O_j + \theta_j$ , thus by taking partial derivative of it, gives

$$\frac{\partial net_k}{\partial W_{kj}} = O_j \quad (2.12)$$

Substitute (2.11) and (2.19) into (2.9), it will give

$$\frac{\partial E}{\partial W_{kj}} = \delta_k \times O_j \quad (2.13)$$

We know that  $\frac{\partial E}{\partial net_k} = \delta_k$ , by chain rule

$$\delta_k = \frac{\partial E}{\partial o_k} \times \frac{\partial o_k}{\partial net_k} \quad (2.14)$$

Since the error function is given by  $E = \frac{1}{2} \sum_k (t_{kj} - o_{kj})^2$ , the partial derivative is

$$\frac{\partial E}{\partial o_k} = -(t_{kj} - o_k) \quad (2.15)$$

We know that  $o_k = \frac{1}{1 + e^{-net_k}}$ , the partial derivative becomes

$$\frac{\partial o_k}{\partial net_k} = o_k (1 - o_k) \quad (2.16)$$

Substitute (2.14) and (2.15) into (2.13), it gives

$$\delta_k = -(t_k - o_k) o_k (1 - o_k) \quad (2.17)$$

Substitute (2.16) into (2.12), it gives

$$\begin{aligned}\frac{\partial E}{\partial W_{kj}} &= \delta_k \times O_j \\ \frac{\partial E}{\partial W_{kj}} &= -(t_k - o_k) o_k (1 - o_k) \times O_j\end{aligned}\quad (2.18)$$

The adaptation of weight between output and hidden layer is now:

$$\begin{aligned}\Delta W_{kj} &= -\eta \frac{\partial E}{\partial W_{kj}} \\ \Delta W_{kj}(n) &= -\eta(-(t_k - o_k) o_k (1 - o_k) O_j) + \alpha \Delta W_{kj}(n-1) \\ \Delta W_{kj}(n) &= \eta(t_k - o_k) o_k (1 - o_k) O_j + \alpha \Delta W_{kj}(n-1)\end{aligned}$$

with the added momentum term  $\alpha$ , for a better convergence.

The weight updating between hidden layer ( $j$ ) and input layer ( $i$ ) of the two-term BP is similar to weight updating between output layer ( $k$ ) and hidden layer ( $j$ ). The delta rule as below:

$$\Delta W_{ji} = -\eta \frac{\partial E}{\partial W_{ji}} \quad (2.19)$$

By chain rule,

$$\frac{\partial E}{\partial W_{ji}} = \frac{\partial E}{\partial net_j} \times \frac{\partial net_j}{\partial W_{ji}} \quad (2.20)$$

Since we know that  $net_j = \sum_j W_{ji} O_i + \theta_j$ , thus by taking partial derivative it gives

$$\frac{\partial net_j}{\partial W_{ji}} = O_i \quad (2.21)$$

Assume

$$\frac{\partial E}{\partial net_j} = \delta_j \quad (2.22)$$

And we know that,

$$\frac{\partial E}{\partial net_j} = \frac{\partial E}{\partial o_j} \times \frac{\partial o_j}{\partial net_j} \quad (2.23)$$

Since  $o_j = \frac{1}{1 + e^{-net_j}}$ , thus by taking partial derivative of error, it gives

$$\frac{\partial E}{\partial net_j} = o_j(1 - o_j) \quad (2.24)$$

By chain rule,

$$\frac{\partial E}{\partial o_j} = \frac{\partial E}{\partial net_k} \times \frac{\partial net_k}{\partial o_j} \quad (2.25)$$

And from the previous derivation in equation (2.9) and (2.15),

$$\frac{\partial E}{\partial net_k} = \delta_k \quad (2.26)$$

and

$$\frac{\partial net_k}{\partial o_j} = W_{kj} \quad (2.27)$$

Therefore, substitute equation (2.25) and (2.26) into (2.24), it gives

$$\frac{\partial E}{\partial o_j} = \delta_k \times W_{kj} \quad (2.28)$$

Substitute (2.27) into (2.22) gives

$$\frac{\partial E}{\partial net_j} = \delta_k W_{kj} \times o_j(1 - o_j) \quad (2.29)$$

Substitute (2.28) into (2.19):

$$\begin{aligned}\frac{\partial E}{\partial W_{ji}} &= \frac{\partial E}{\partial net_j} \times \frac{\partial net_j}{\partial W_{ji}} \\ \frac{\partial E}{\partial W_{kj}} &= \delta_k W_{kj} \times o_j (1 - o_j) \times O_i\end{aligned}\quad (2.30)$$

Therefore, the adaptation of weight between hidden and input layers is now:

$$\begin{aligned}\Delta W_{ji}(n) &= -\eta \frac{\partial E}{\partial W_{ji}} \\ \Delta W_{ji}(n) &= -\eta \delta_k W_{kj} \times o_j (1 - o_j) \times O_i + \alpha \Delta W_{ji}(n-1)\end{aligned}$$

where  $\delta_k = -(t_k - o_k) o_k (1 - o_k)$ , with  $\alpha$  is the momentum rate added for a better convergence.

### 2.5.2 Two-term Improved Error Function

MSE is not the best function to use for training a neural network for classification problems [19]. Thus, this study will focus on two-term improved error function for classification problem. This new improved two-term error function shows a rapid change compared to MSE, therefore gives less iteration for convergence [10].

The proposed IeBP is produced by modifying an error function of [32] to increase the convergence rate of the BP training. This IeBP (*mm*) is defined implicitly as in [10]:

$$mm = \sum_k \rho_k$$

$$\rho_k = \frac{E_k^2}{2a_k(1-a_k^2)} \quad (2.31)$$

where,

$$E_k = t_k - a_k,$$

$E_k$  is error at output unit  $k$

$t_k$  is target value of output unit  $k$

$a_k$  is an activation of unit  $k$

As we know that the updating weight of standard BP model is using the delta rule, thus in this case, the above relation can be rewritten as

$$\frac{\partial \rho_k}{\partial w_k} = \frac{\partial \rho}{\partial Net_k} \cdot \frac{\partial Net_k}{\partial w_k} \quad (2.32)$$

Known that  $Net_k = \sum_j w_{kj} a_j + \theta_k$ , thus by taking partial derivative of it and substitutes it into (2.31) gives

$$\frac{\partial \rho_k}{\partial w_k} = \frac{\partial \rho}{\partial Net_k} \cdot a_j \quad (2.33)$$

Let's assume that  $\frac{\partial \rho}{\partial Net_k} = \delta_k$ . By using chain rule, this gives

$$\frac{\partial \rho_k}{\partial Net_k} = \frac{\partial \rho}{\partial a_k} \cdot \frac{\partial a_k}{\partial Net_k} \quad (2.34)$$

By taking partial derivatives of the activation function  $f(Net_k) = 1/(1 + e^{-Net_k})$  and simplify it by substituting in terms of  $a_k$ , gives

$$f'(Net_k) = \frac{\partial a_k}{\partial Net_k} = a_k(1 - a_k) \quad (2.35)$$

It is known that  $\rho_k = \frac{E^2}{2a(1 - a_k^2)}$ . Thus by taking partial derivatives with respect to  $a_k$ ,

gives

$$\frac{\partial \rho_k}{\partial a_k} = \frac{-[4a_k(t_k - a_k)(1 - a_k^2) + 2(t_k - a_k)^2(1 - 3a_k^2)]}{4a_k^2(1 - a_k^2)^k} \quad (2.36)$$

To simplify, equation (2.34) becomes

$$\frac{\partial \rho_k}{\partial a_k} = -\left( \frac{E + \rho(1 - 3a_k^2)}{a_k(1 - a_k^2)} \right) \quad (2.37)$$

By substituting (2.33) and (2.35) into (2.32) we have improved error signal of BP for the output layer as,

$$\frac{\partial \rho_k}{\partial Net_k} = -\frac{(E + \rho(1 - 3a_k^2))}{1 + a_k} = \delta_k \quad (2.38)$$

By substituting (2.36) into (2.31), gives

$$\frac{\partial \rho_k}{\partial w_k} = -\frac{(E + \rho(1 - 3a_k^2))}{1 + a_k} \cdot a_j \quad (2.39)$$

The adaptation of weight between output layer and hidden layer is now:

$$\begin{aligned} \Delta w_k &= -\eta \frac{\partial \rho}{\partial w_k} \\ \Delta w_k &= -\eta \left( -\frac{(E + \rho(1 - 3a_k^2))}{1 - a_k^2} \right) \cdot a_j \\ \Delta w_k &= \eta \left( \frac{(E + \rho(1 - 3a_k^2))}{1 - a_k^2} \right) \cdot a_j \end{aligned}$$

And an error signal for modified BP of the hidden layer is the same as standard BP,

$$\delta_j = \sum \delta_k w_{kj} f'(a_j) \quad (2.40)$$

## 2.6 The Logistic Activation Function

In many ANN applications, the activation functions play an important role as in the early years; they are used to fire or unfire the neuron. However, in new neural network paradigms, the activation functions are more sophisticatedly used. Nowadays, many activation functions have been used to produce a continuous value rather than discrete. One of the most popular activation functions used is the logistic activation function or more popularly referred to as the sigmoid function. This function is semi linear in characteristic, differentiable and produces a value between 0 and 1. The mathematical expression of this sigmoid function is:

$$f(a) = \frac{1}{1 + \exp^{-\beta x}} \quad (2.41)$$

where  $\beta$  controls the *slope parameter* of the sigmoid. When  $\beta$  is large, the sigmoid becomes like a threshold function and when  $\beta$  is small, the sigmoid becomes more like a straight line (linear). When  $\beta$  large, learning is much faster but a lot of information is lost, however when  $\beta$  small, learning is very slow but information is retained. Because this function is differentiable, it enables the BP algorithm to adapt the lower layers of weights in a multilayer neural network.

## 2.7 Related Research Work on ANN

The BP algorithm [3] is one of the most widely used supervised training algorithms for MFNN. As mentioned earlier, it does suffer from slow convergence rate and tends to trap in local minima. Hence, there has been a lot of research to overcome the problem by perform some modification to the conventional ANN algorithms. Table 2.3 below lists some of the related research work of ANN that has been done so far.

**Table 2.3: Summary of Related Research Work on ANN**

Research	Description
Oh., <i>et al</i> 1997  [9]	<ul style="list-style-type: none"> <li>• Proposed a modified function               <ul style="list-style-type: none"> <li>- reduces the probability that output nodes are near wrong extreme value of sigmoid activation function</li> <li>- prevents the overspecialization during learning</li> </ul> </li> </ul>
Qingwu, M., <i>et al</i>  2000  [33]	<ul style="list-style-type: none"> <li>• Uses fuzzy logic to adjust               <ul style="list-style-type: none"> <li>- network structure, initial weights, learning rate</li> </ul> </li> <li>• To guide the selection of the net topology</li> <li>• To simulate geology evaluation</li> <li>• To seek the solution for gradient descent algorithm</li> <li>• Proves can solve nonlinear problem &amp; to simulate vary complicated relationship</li> </ul>
Wang., <i>et al</i>  2004  [34]	<ul style="list-style-type: none"> <li>• Proposed a modified error function with two-terms</li> <li>• Efficiently avoid the local minima</li> <li>• Proves this modified error function improves the performance of BP algorithm</li> </ul>
Mohammed A. Otair., Walid A. Salameh  2005  [35]	<ul style="list-style-type: none"> <li>• Developed Optical BP algorithm               <ul style="list-style-type: none"> <li>- used non-linear function &amp; manage to escape from local minima with high-speed of convergence rate</li> </ul> </li> <li>• Tested against standard BP and OBP</li> <li>• OBP performs much better and faster than the BP for all training processes with different learning rate.</li> </ul>

<p>Nawi., R. S. Ransing and M. R. Ransing</p> <p>2008</p> <p>[36]</p>	<ul style="list-style-type: none"> <li>• New fast learning algorithm for NN based on Fletcher-Reeves update with adaptive gain (CGFR/AG) training algorithms</li> <li>• The conjugate gradient optimization algorithm is combined with the modified back propagation algorithm</li> <li>• The computational efficiency is enhanced by adaptively modifying initial search direction (gradient search direction)             <ol style="list-style-type: none"> <li>(1) Modification on standard backpropagation algorithm by introducing a gain variation term in the activation function,</li> <li>(2) Calculation of the gradient descent of error with respect to the weights and gains values and</li> <li>(3) the determination of a new search direction by using information calculated in step (2).</li> </ol> </li> <li>• Proves that the proposed algorithm is robust and has a potential to significantly enhance the computational efficiency of the training process</li> </ul>
---	--

From the summary above, most of the past research only focuses on the error functions of standard BP by using different methodologies. The results compared based on the performance of the algorithm used. The equation (2.41) has been implemented in the IeBP by [10]. Hence, this study will focus on tuning of the  $\beta$ , the slope parameter and its impact on the BP learning algorithms. The performance results will be compared against the Standard BP and the IeBP algorithms to give better conclusions.

## 2.8 Summary

This chapter has explained the related literature review for this study. The first part has explained regarding the history and the architecture of the ANN itself. The later part is about the related research work on the ANN. And all these partial results have contributed to carry out this project in giving a better and complete conclusion.

## **CHAPTER 3**

### **METHODOLOGY**

#### **3.1 Introduction**

This chapter explains the methodology that has been used to carry out the testing for both Standard BP and also IeBP. The discussion about the processes involved and the explanation of the dataset that has been used also included in this chapter. Besides, the techniques and the scope parameters are discussed in here. Then, the explanation on how the experiment and analysis to investigate the efficiency and the convergence rate of standard BP and IeBP also included in this chapter.

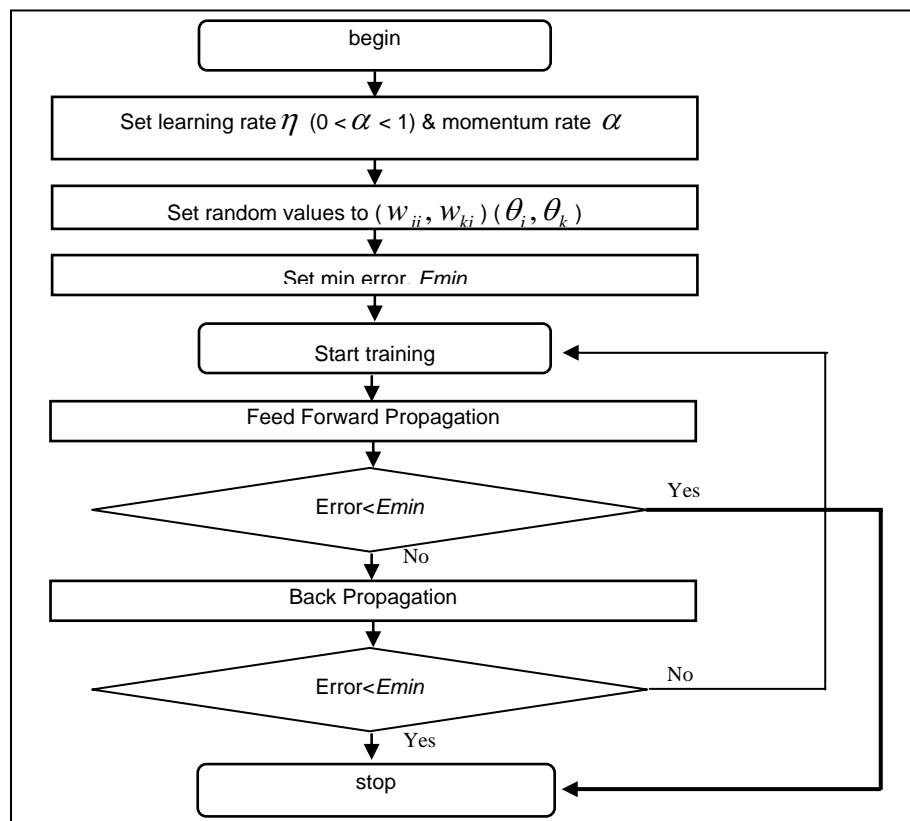
#### **3.2 BP Training**

In BPNN, the learning algorithm has two phases. First, a training input pattern is presented to the network input layer. The input pattern is propagated

through the network from left to right, and the output layer is different from the desired output, an error is calculated and then propagated backwards through the network. The weights are modified as the error is propagated. These training phases can be described with the step by step as in Table 3.1 and the flow as in Figure 3.1.

**Table 3.1: The BP Training Steps**

<b>Step 1</b>	: Initialize the weights to small random values
<b>Step 2</b>	: Randomly choose and input pattern
<b>Step 3</b>	: Propagate the signal forward through the network
<b>Step 4</b>	: Compute the prediction error
<b>Step 5</b>	: Weights adjustments by propagating the prediction error backwards
<b>Step 6</b>	: Update the weights
<b>Step 7</b>	: Repeat (step 2) for the next pattern until the error in the output layer (prediction error) is below a thresh



**Figure 3.1: The Standard BP Flow Chart**

The most important attribute which may affects the design of NN algorithm is the weight updating strategy. Generally, there are three strategies used as in table 3.2 [23]:

**Table 3.2: Weight Updating Strategies**

<b>Pattern mode</b> (on-line)	weight updating is done after the presentation of each training pattern
<b>Batch mode</b> (off-line)	weight updating is performed after the presentation of all the training patterns
<b>Block mode</b>	weight updating is performed after the presentation of a subset of the training patterns

The batch mode strategy is used on standard BP and IeBP in this study. The batch mode also known as off-line mode is chosen, as it is well suited due to weight errors, which can be calculated independently. The table 3.3 below describes the batch-training algorithm.

**Table 3.3: Batch Training Algorithm**

<p><i>1. For each pattern</i></p> <p><i>1.1 Compute the output of units in the hidden layer:</i></p> <p><i>1.2 Compute the output of units in the output:</i></p> <p><i>1.3 Compute error terms for the units in the output layer:</i></p> <p><i>1.4 Compute error terms for the units in the hidden layer:</i></p> <p><i>1.5 Compute weight changes in the output layer:</i></p> <p><i>1.6 Compute weight changes in the hidden layer:</i></p> <p><i>2. Send local weight of (output, hidden) and Receive remote weight of (output, hidden)</i></p> <p><i>3. Update weights changes in the output layer:</i></p> <p><i>4. Update weights changes in the hidden layer:</i></p> <p><i>Until</i></p> <p><i>(<math>E &lt; \text{maximum accepted error}</math>) or (num. of iterations = maximum num. of iterations)</i></p>
---

### 3.3 Dataset

Dataset is needed to represent the problem. It is used to verify the efficiency and convergence rate of both standard BP and IeBP algorithms. These both algorithms are applied to the three dataset of real world classification problem, which is taken from the universal data. The three datasets that have been used in this study are Iris, Balloon, and Cancer. These data are chosen because the samples of each dataset are already processed and cleaned. Moreover, these datasets are benchmark data with rules and conventions. The datasets are represented as the input pattern in BP learning algorithm. In addition, this study uses these same three datasets to be tested with  $\beta = 2,3,4$ . The following describes briefly the characteristic of each dataset.

#### Iris

The Iris dataset is used for classifying four-dimensional patterns, which are sepal length, sepal width, petal length, and petal width into three classes, which are iris setosa, iris versicolor and iris virginica. Iris dataset contains data for three classes with 50 instances each, where each class refers to type of iris plant. The input consists of 4 numeric attributes related to the length and the width of the sepals and petals of the iris plant. The network will have 4 input and 3 outputs to separate the pattern. The Iris dataset were split with 120 for training and 30 for testing, totaling 150 instances.

#### Balloons

This dataset is published by Michael Pazzani. There are four sets of data in Balloons dataset that represent different conditions of an experiment with the same attributes:

- a. Adult - stretch data: Inflated is true if age = adult or act = stretch
- b. Adult + stretch data: Inflated is true if age = adult and act = stretch
- c. Small - yellow data: Inflated is true if (color = yellow and size = small)
- d. Small – yellow data: Inflated is true if (color = yellow and size = small) or (age = adult and act = stretch)

Balloon dataset is classified whether it is inflated or it is not inflated according to four attributes: color, size, act and age. The values for each attribute are listed in Table 3.4 below. The Balloon dataset consists of 12 training pattern and 4 testing pattern, totaling 16 instances.

**Table 3.4: Summary of Attribute Values for Balloon Dataset**

No.	Attribute	Values
1	Color	Yellow, Purple
2	Size	Large, Small
3	Act	Stretch, Dip
4	Age	Adult, Child

### **Cancer**

The Cancer dataset requires the decision maker to correctly diagnose breast lumps as either benign or malignant based on data from automated microscopic examination of cells collected by needle aspiration. The dataset includes nine inputs and one output. The exemplars are split with 599 for training and 100 for testing, totaling 699 exemplars. All inputs are continuous variables and 65.5% of the examples are benign. The dataset was originally generated at hospitals at the University of Wisconsin Madison, by Dr. William H. Wolberg.

## **3.4 Implementation of Neural Network**

There are a few issues that have to be taken into consideration before the implementation of Neural Network. The following subtopics will discuss about the implementation of the neural network in this study.

### 3.4.1 Define BPNN Network Architecture and Parameters

BPNN network architecture involves the selecting of an appropriate number of layers and the number of nodes in each layer based on the size and type of the application and the problem involved. As stated earlier, the neural network architecture consists of three layers, which are the input layer, hidden layer and also the output layer. The required number of nodes in each layer also differs from each dataset based on the classification problem. Thus, the number of nodes in the input and output layers are determined by the input and output variables based on our dataset.

Nevertheless, the essential number of hidden nodes did not present. These hidden nodes are required for the computing difficult functions recognized as the non-separable functions. The number of nodes in the hidden layer determines the network's learning capabilities. It's been crucial of selecting the appropriate number of hidden layer for the optimal network design. The hidden layer size may affect the complexity and the required time for training but, out of all, it could influence its competence to generalize. However, there is no an appropriate standard rule or theory to determine the optimal number of hidden nodes [37]. There are many types of techniques and methods that have been used by researches for determining the right number of hidden nodes. Some of the suggested techniques are summarized as below:

1. The number of hidden nodes should be 2/3 of the size of input layer, as well as the size of output layer.
2. Pyramidal shape topology [38].
3. Kolmogorov theorem: One hidden layer and  $(2N + 1)$  hidden neurons are sufficient for  $N$  inputs [38].
4. The number of hidden nodes is selected either arbitrarily based on the trial and error approaches [39].
5. The number of hidden nodes should be  $\sqrt{m * n}$  where  $m$  is the number of input nodes and  $n$  is the number of output nodes [39].

In this study, the number of nodes in hidden layer has been chosen based on the Kolmogorov theorem. Network with lower hidden nodes are preferable as they usually have better generalization ability and less over fitting problem. The details of the network architecture for the three classification problems are shown in Table 3.5.

**Table 3.5: The Network Architecture of Iris, Balloon and Cancer Dataset**

Parameters	Iris	Balloon	Cancer
Number of layers	3	3	3
Number of input nodes	3	4	9
Number of hidden nodes	7	9	19
Number of output nodes	3	1	1
Training data	120	12	599
Testing data	30	4	100

All the nodes are associated with weight, bias, error function and activation function. In this study, the random approach is selected to initialize the weight and bias. There are two error function that have been implied in this study which are the MSE and also the improved error function by [10].

### 3.4.2 Formulation of Weight Adjustment

The main focus of this study is the activation function. The conventional sigmoid function will be applied to the standard BP. Meanwhile, the improved error function applied by applying with three different value (2, 3, 4) of the  $\beta$ , slope parameter. The derivative of the error function is one of the factors in the weight update equations [19]. Therefore the formulation of derivation of weight update equation is based on the new

improved error function in equation (3.2). The new error signal is employed only to output layer and the weight adjustment is adapted through output layer to hidden layer. The error signal of the hidden layer is the same as standard BP. All the derivations have been discussed in Chapter 2.

### **3.4.3 Define the Learning Rate and Momentum Term**

The core parameters for NN are the learning rate and momentum term, as these values will have an effect on the learning performance [40]. The earlier studies by [41] have suggested three learning rates (0.1, 0.5, and 0.9) and three momentum term also have been suggested by [42] which are (0.1, 0.5 and 0.9). As for this study, the suggested values for the learning rate,  $\eta$  and momentum term,  $\alpha$  are used for each dataset against the standard BP and IeBP with three different slope parameters,  $\beta$ .

### **3.4.4 Define the Maximum Error**

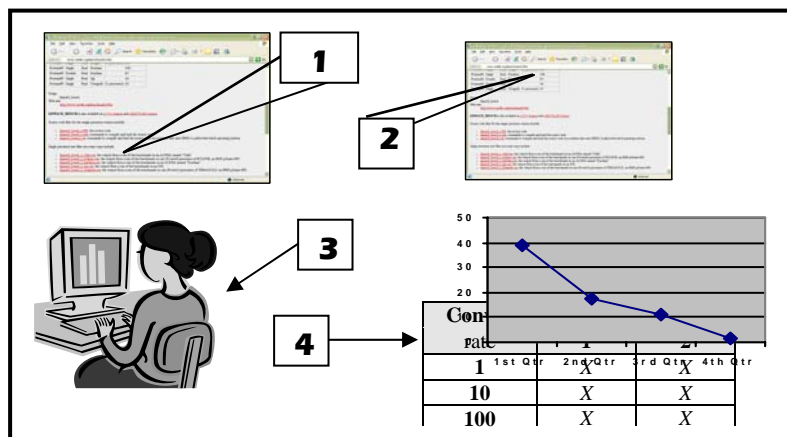
Maximum error is another parameter that should be taken into the consideration. This maximum error should be the stopping criteria for the BP training. As for this study, the maximum error is set to 0.05. Besides that, the training process of the BP is been set to a maximum of 10,000 iterations, or until the error reaches the maximum error. It is adequate for the network to train the dataset within 10,000 iterations and converge to the solution. Since the main focus of this study is the faster convergence rate, thus the minimal iteration is important.

### 3.5 Workflow

Basically, there are four core processes that are involved to carry out this study. Table 3.6 describes the task description while figure 3.2 shows the process.

**Table 3.6 : Implementation Description**

PROCESS	DESCRIPTION
1	The Std BP algorithms are tested on predefined datasets
2	The IeBP algorithms are tested on the same predefined datasets
3	The modification of $\beta$ parameter is done on IeBP
4	Output is generated based on the pre-defined parameters



**Figure 3.2: Implementation Process**

### 3.6 Experiment and Analysis

The main focus of this study is to evaluate and to seek the performance of the IeBP in classification problems. While looking at the objectives that are achieved in this study are the comparison of performance between for both standard BP and IeBP with

three different  $\beta$ , slope parameter values. Therefore, the experiment is carried out in two parts with these three main concerns as in table 3.7 below.

**Table 3.7: The Parameter of Algorithm Experiment**

Parameters	Standard BP	IeBP
<b>Activation function</b>	$f(x) = \frac{1}{1 + e^{-x}}$	$f(x) = \frac{1}{1 + e^{-\beta x}}; \beta = 2,3,4$
<b>Error Signal</b>	$\delta_k = -(t_k - o_k)o_k(1 - o_k)$	$\delta_k = \frac{(E + \rho(1 - 3a_k^2))}{1 + a_k} . a_j$
<b>Error function</b>	$E = \frac{1}{2} \sum_k (t_{kj} - o_{kj})^2$	$mm = \sum_k \rho_k$ $\rho_k = \frac{E_k^2}{2a_k(1 - a_k^2)}$

The results are compared based on several aspects such as the efficiency, convergence time and accuracy for each dataset. The analysis is done based on the result obtained from all the experiments.

### 3.7 Summary

This chapter discusses the methodology of the methods applied for the Standard BP and also the IeBP. Based on previous work and research, the IeBP proven to be better than the standard BP in term of performance. But this project is deliberate more on to the error signal and weight update method which involves the  $\beta$  parameter in the IeBP. The related methodologies involved in this project have been described above.

## **CHAPTER 4**

### **EXPERIMENTAL RESULT AND ANALYSIS**

#### **4.1 Introduction**

The chapter discusses the implementation phase of the methods for both standard BP and IeBP. The experiments have been carried out to validate the performance of the based on the Iris, Balloon and Cancer datasets. The results for each experiment is compared and analyzed.

In this study, the modified error function [10] has been applied in IeBP for every testing. A series of experiments have been carried out with three universal datasets in order to investigate the efficiency and the accuracy of both standard BP and IeBP.

The experiments are carried out which consists of nine sets of mixture for learning rate and momentum term values; represented as V1, V2, V3, V4, V5, V6, V7, V8 and V9. These same nine sets of trial were used for each dataset (Iris, Balloon and Cancer) experiments. The range of learning rate and momentum term for each experiment has been chosen between 0.1, 0.5 and 0.9 as suggested in [40][41]. Thus, the results are compared and analyzed for each datasets based on the error convergence, maximum epochs, and the classification performance. These results are analyzed based on these comparisons:

- a) Comparison of standard BP and IeBP ( $\beta = 2,3,4$ ) for each datasets

## **4.2 Algorithm Implementation**

The algorithms of the standard BP has been tested using three universal datasets. As described in Chapter 3, the modified error function [10] has been applied in IeBP program. After these have been successful, some modification has been done for the IeBP algorithm on the slope parameter using  $\beta = 2,3,4$ , so it could perform on these same datasets. Once these algorithms perform consecutively, the final results after learning and training have been captured out. Then these values have been gathered accordingly for analysis.

## **4.3 Experimental Result**

The testing of standard BP and IeBP have been carried out with three different dataset (Iris, Balloon and Cancer). The results for each experiment that have been carried out are discussed in the next section.

#### 4.4 Result of Iris Dataset

The experiment on Iris dataset refers to the training of network consists of 4 input nodes, 9 hidden nodes and 3 output nodes. The training set consists of 120 inputs and testing set is 30 inputs. The experiment will be terminated with two predefined condition; either the training error below minimum error (0.05) or within 10,000 iteration. The experimental result of Iris dataset for standard BP and IeBP ( $\beta = 2,3,4$ ) are presented in Table 4.1(a), (b), (c) and (d) respectively. Figure 4.1 presents the learning pattern produced by standard BP and IeBP ( $\beta = 2,3,4$ ).

**Table 4.1 (a): Result of Standard BP**

Group		Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)
Standard BP	V1	0.1	0.1	1.53290	10000	78.14
	V2	0.1	0.5	0.99446	10000	78.20
	V3	0.1	0.9	0.04998	4587	80.34
	V4	0.5	0.1	0.04997	8683	80.06
	V5	0.5	0.5	0.04993	4780	80.35
	V6	0.5	0.9	3.99596	10000	75.76
	V7	0.9	0.1	1.40844	10000	78.43
	V8	0.9	0.5	0.04950	3475	80.24
	V9	0.9	0.9	4.99997	10000	75.87

**Table 4.1 (b): Result of IeBP ( $\beta = 2$ )**

Group		Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)
$\beta = 2$	V1	0.1	0.1	1.00148	10000	78.24
	V2	0.1	0.5	0.19533	10000	79.86
	V3	0.1	0.9	0.04998	2922	80.39
	V4	0.5	0.1	0.51957	10000	76.04
	V5	0.5	0.5	0.04987	4223	80.87
	V6	0.5	0.9	0.03911	1974	83.23
	V7	0.9	0.1	2.00181	10000	78.20
	V8	0.9	0.5	1.50111	10000	79.09
	V9	0.9	0.9	2.50026	10000	76.16

**Table 4.1 (c): Result of IeBP ( $\beta = 3$ )**

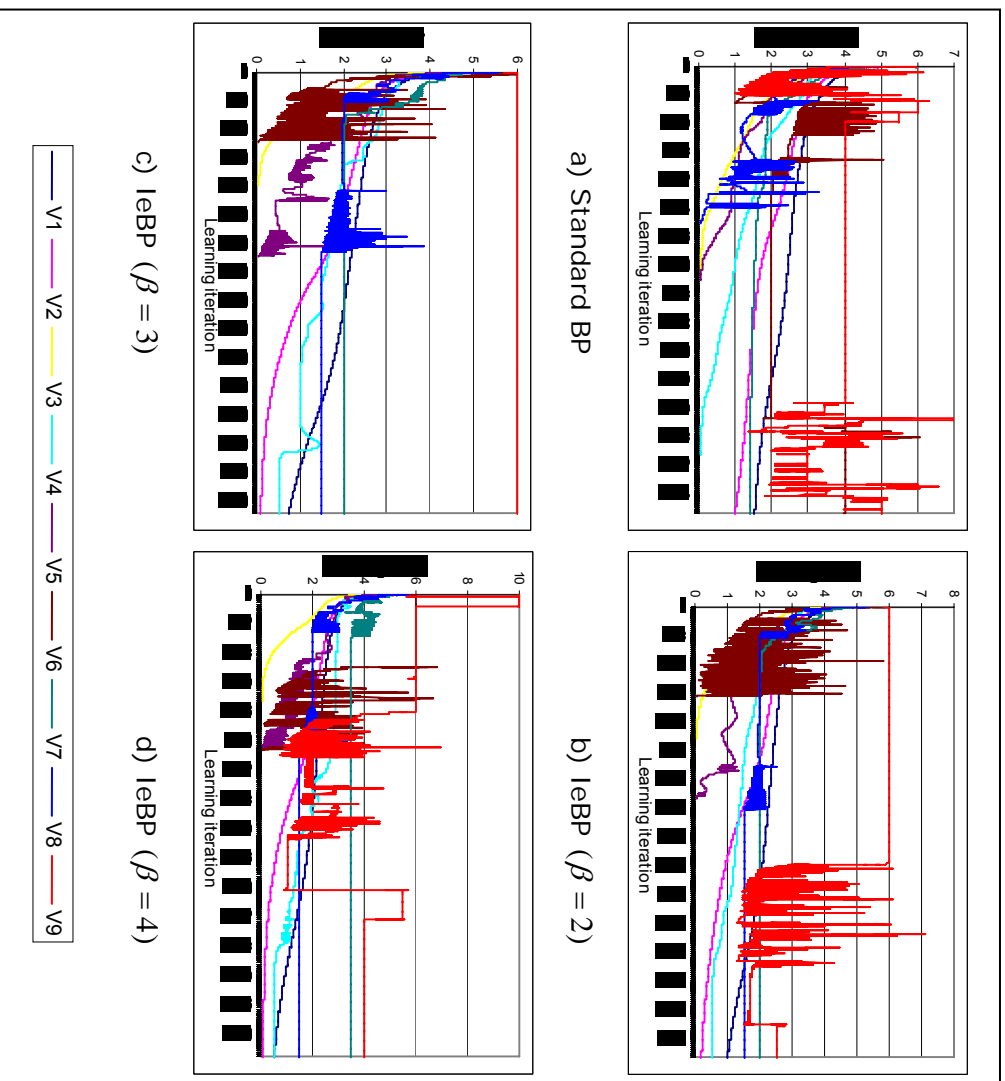
Group	Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)	
$(\beta = 3)$	V1	0.1	0.1	0.73464	10000	79.34
	V2	0.1	0.5	0.09524	10000	80.26
	V3	0.1	0.9	0.04997	2533	81.44
	V4	0.5	0.1	0.51429	10000	75.24
	V5	0.5	0.5	0.03608	4140	83.87
	V6	0.5	0.9	0.04949	1546	82.09
	V7	0.9	0.1	2.00147	10000	78.01
	V8	0.9	0.5	1.50053	10000	79.15
	V9	0.9	0.9	6.00000	10000	76.16

**Table 4.1 (d): Result of IeBP ( $\beta = 4$ )**

Group	Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)	
$(\beta = 4)$	V1	0.1	0.1	0.50548	10000	80.74
	V2	0.1	0.5	0.07214	10000	81.67
	V3	0.1	0.9	0.04996	2277	81.98
	V4	0.5	0.1	0.51245	10000	75.02
	V5	0.5	0.5	0.04122	3296	82.99
	V6	0.5	0.9	0.04795	3348	82.33
	V7	0.9	0.1	3.50072	10000	77.72
	V8	0.9	0.5	1.50066	10000	79.20
	V9	0.9	0.9	3.99999	10000	77.63

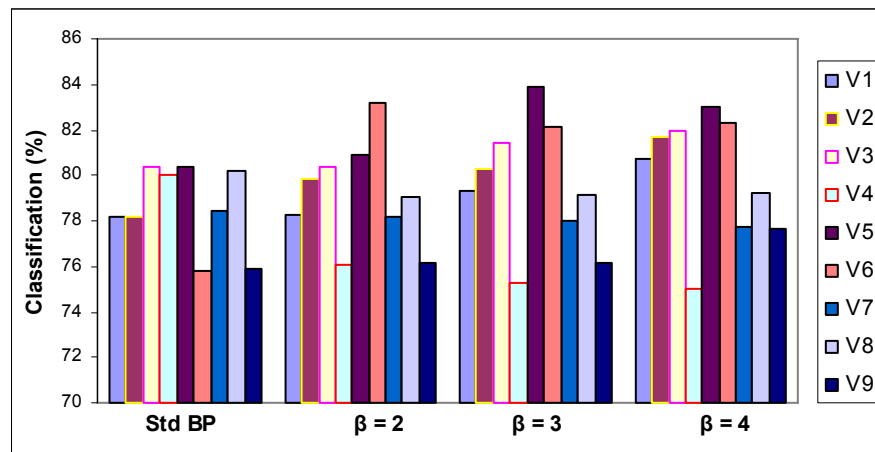
From the table 4.1 above, shows that the results for all trial for each experiment are almost similar to each other. Standard BP could give a better result at V8 trial compared to other IeBP in term of learning iterations. Figure 4.1 shows the convergence graph of learning result produced by each trial.

Figure 4.1 below shows the learning pattern of standard BP and IeBP; for each  $\beta$  parameter. The graph above shows that the learning pattern become steeper as the  $\beta$  value increased. For standard BP, the errors generated converge closely matching the maximum error and stopped at iteration 4587, 8683, 4780, and 3475 for V3 till V5 and V8 trial. The other trials could not converge to solution within 10,000 iterations.



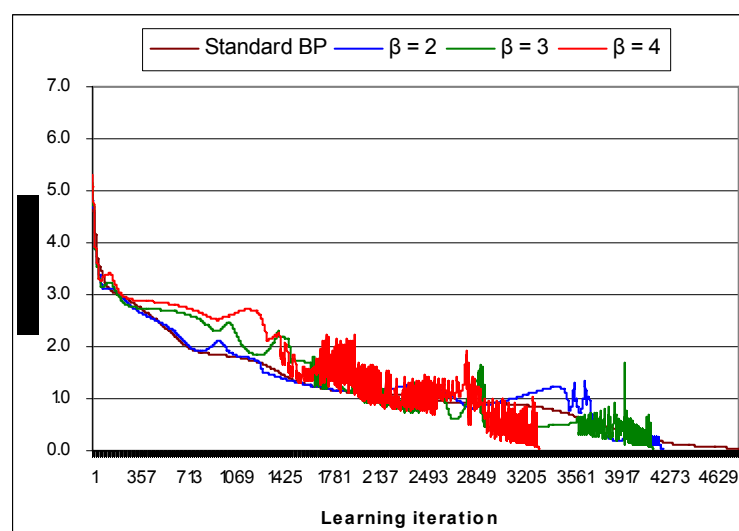
**Figure 4.1: Convergence of Iris Dataset for Standard BP and lBP ( $\beta = 2,3,4$ )**

However, for lBP ( $\beta = 2,3,4$ ), the V1, V2, V4 and V7 till V9 trials could not converge to solution within 10, 000 iterations. The lBP of ( $\beta = 2,3$ ) has the fastest converge at V6 trial whereas lBP of ( $\beta = 4$ ) converge faster at V3 trial. Meanwhile figure 4.2 below presents the classification performance for each trial, V1 till V9 formed by standard BP and lBP ( $\beta = 2,3,4$ ) respectively.



**Figure 4.2: Classification Accuracy of Iris Dataset**

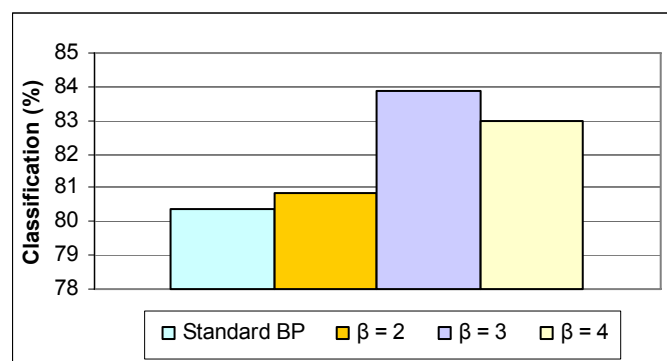
From the figure 4.2 above, the classification accuracy for IeBP are slightly better than the standard BP. The trial pair of V6 gain better accuracy in IeBP ( $\beta = 2$ ), V5 outperforms well in IeBP ( $\beta = 3,4$ ). However, the V4 trial shows decrease in classification performance for the following algorithms. The V7 trial shows a fair classification performance between these algorithms. Meanwhile, V2 and V5 trial performance are increased as the slope parameter are increased. From the analysis above, only a single result is taken which is considerably good over all trials. Thus, the result produced by trial V5 (learning rate, 0.5 and momentum term, 0.5) is chosen for achieving highest convergence rate for IeBP and depicted in Figure 4.3 below.



**Figure 4.3: Convergence Characteristic of V5 Trial**

The standard BP takes 4780 iteration to converge at 0.04993 while the IeBP take 4223, 4140 and 3296 of iterations to converge at 0.04987, 0.03608 and 0.04122 respectively. The V5 trial shows the learning speeds are become faster as the value of  $\beta$  parameter are increased even there is inconsistent in the early phase of learning speed.

In term of classification performance, the accuracy for standard BP and IeBP are significantly getting better. The accuracy of classification for standard BP achieves 80.35%, 80.87% for IeBP ( $\beta = 2$ ); 83.87% for IeBP ( $\beta = 3$ ) and decreases to 82.99% for IeBP ( $\beta = 4$ ) at V5 trial in classifying the Iris dataset as illustrated in figure 4.4 below.



**Figure 4.4: Correct Classification of V5 Trial**

#### 4.5 Result of Balloon Dataset

The network used for experimenting Balloon dataset consists of 4 input nodes, 9 hidden nodes and 1 output node. 12 data represent to the network as training set and 4 data as testing set. The experimental result of Iris dataset for standard BP and IeBP ( $\beta = 2,3,4$ ) are presented in Table 4.2(a), (b), (c) and (d)

respectively. Figure 4.3 presents the learning pattern produced by standard BP and IeBP ( $\beta = 2,3,4$ ).

**Table 4.2 (a): Result of Standard BP**

Group		Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)
Standard BP	V1	0.1	0.1	0.04994	1245	76.05
	V2	0.1	0.5	0.04999	693	76.05
	V3	0.1	0.9	0.04933	150	76.07
	V4	0.5	0.1	0.04962	259	76.04
	V5	0.5	0.5	0.04990	148	76.04
	V6	0.5	0.9	0.04774	88	76.00
	V7	0.9	0.1	0.04986	149	76.04
	V8	0.9	0.5	0.04893	88	76.01
	V9	0.9	0.9	0.03761	503	76.00

**Table 4.2 (b): Result of IeBP ( $\beta = 2$ )**

Group		Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)
$\beta = 2$	V1	0.1	0.1	0.04990	750	76.04
	V2	0.1	0.5	0.04972	419	76.03
	V3	0.1	0.9	0.04875	93	76.05
	V4	0.5	0.1	0.04916	153	76.02
	V5	0.5	0.5	0.04970	87	76.07
	V6	0.5	0.9	0.04981	363	76.01
	V7	0.9	0.1	0.04929	87	76.04
	V8	0.9	0.5	0.04714	52	76.00
	V9	0.9	0.9	0.50000	10000	76.00

**Table 4.2 (c): Result of IeBP ( $\beta = 3$ )**

Group		Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)
$(\beta = 3)$	V1	0.1	0.1	0.04983	524	76.07
	V2	0.1	0.5	0.04966	293	76.05
	V3	0.1	0.9	0.04946	71	76.05
	V4	0.5	0.1	0.04996	106	76.08
	V5	0.5	0.5	0.04830	62	76.07
	V6	0.5	0.9	0.03991	2724	76.00
	V7	0.9	0.1	0.04883	61	76.04
	V8	0.9	0.5	0.04717	38	76.05
	V9	0.9	0.9	0.50000	10000	76.00

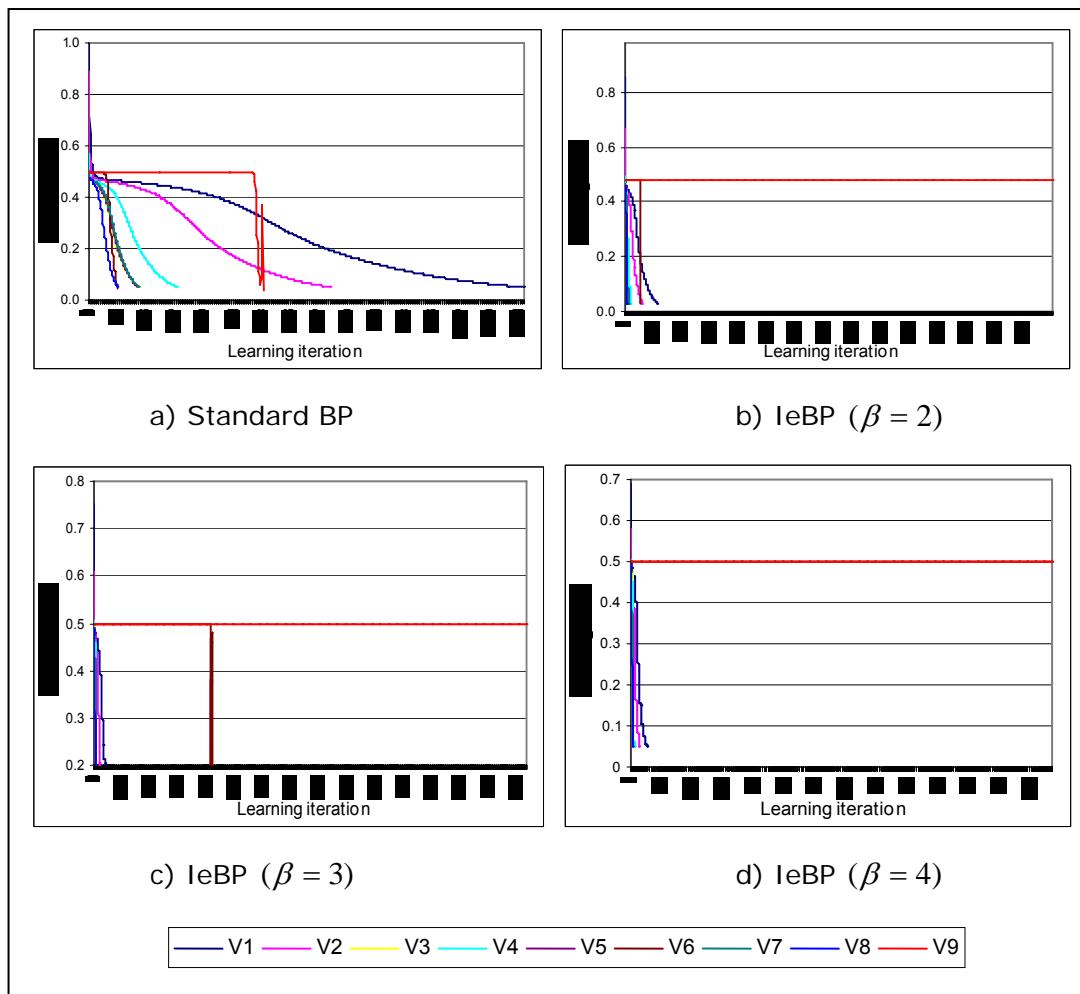
**Table 4.2 (d): Result of IeBP ( $\beta = 4$ )**

Group	Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)	
$(\beta = 4)$	V1	0.1	0.1	0.04992	391	76.04
	V2	0.1	0.5	0.04976	219	76.06
	V3	0.1	0.9	0.04811	63	76.05
	V4	0.5	0.1	0.04923	85	76.07
	V5	0.5	0.5	0.04755	48	76.09
	V6	0.5	0.9	0.50000	10000	76.01
	V7	0.9	0.1	0.04852	47	76.04
	V8	0.9	0.5	0.04757	32	76.03
	V9	0.9	0.9	0.50000	10000	76.00

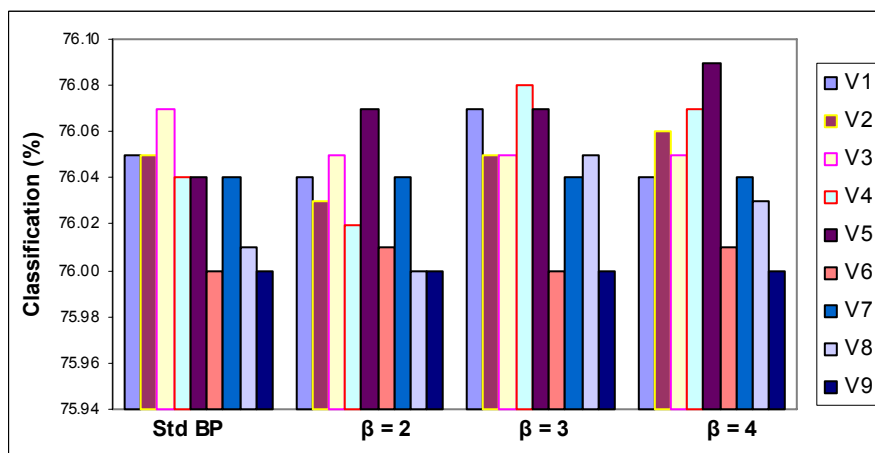
The table above shows that the errors converge nearly comparable to each other. As for standard BP, all the trial pair does converge to solution within the maximum error. The errors generated converge closely match the maximum error function specified earlier.

However, for IeBP( $\beta = 2,3$ ), the V9 trial and for IeBP( $\beta = 4$ ), the V6 and V9 trial pair could not converge to solution within 10,000 iterations. The IeBP of ( $\beta = 2,3,4$ ) has the comparable fastest converge speed at every trial except for the V9 trial which it is tend to trap in local minima. Standard BP could give a better result at V9 trial compared to other IeBP in term of learning iterations.

Figure 4.5 shows the convergence graph of learning result produced by each trial. The standard BP takes 88 iterations to converge while the IeBP takes 52, 38 and 32 of iterations respectively. This shows that the IeBP tend to converge faster than the standard BP for Balloon dataset. But however the V9 trial shows a flat convergence speed which means is likely to trap in local minima.



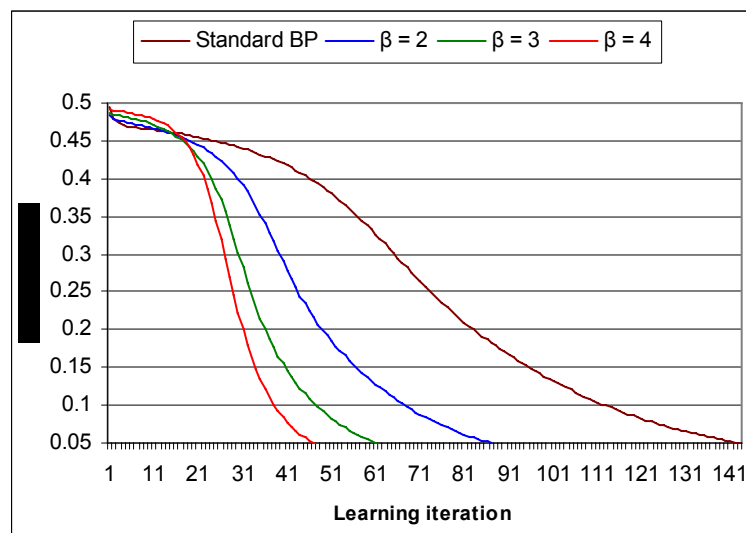
**Figure 4.5: Convergence of Balloon Dataset for Standard BP and IeBP ( $\beta = 2,3,4$ )**



**Figure 4.6: Classification Accuracy of Balloon Dataset**

The figure 4.6 above shows the classification accuracy for every trial of the standard BP and IeBP ( $\beta = 2,3,4$ ). In general, the trial pair of V7 and V9 performs fairly for every experiment. The standard BP has the highest classification rate using the V3 trial pair. The V4, V5 and V7 trial pair performs fair to each other. The V6 and V9 perform the worse for classification purpose in standard BP.

The trial pair of V5 gains better accuracy in IeBP ( $\beta = 2,4$ ) and V4 trial pair in IeBP ( $\beta = 3$ ). However, the V3 trial shows a flat performance for every experiment. Meanwhile, V2 trial performance is increased as the slope parameter is increased. From the analysis above, only a single result is taken which is considerably good over all trials. The V5 trial consists of learning rate 0.5 and momentum term 0.5 shows the learning speeds become faster as the value of  $\beta$  parameter are increased. Thus, the result produced by trial V5 is chosen for achieving highest convergence rate for IeBP and depicted in Figure 4.7 below.

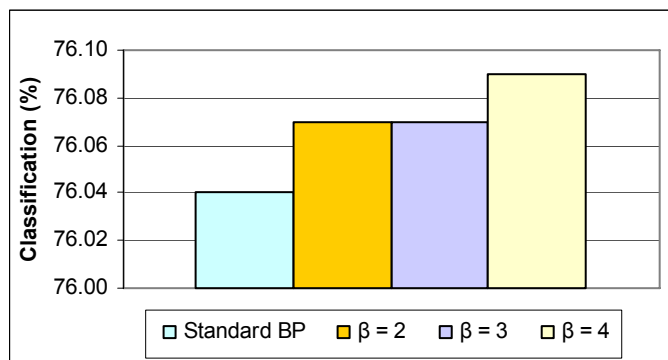


**Figure 4.7: Convergence Characteristic of V5 Trial**

The figure above shows the convergence characteristic of standard and IeBP ( $\beta = 2,3,4$ ) for Balloon dataset. It is clear that the learning speed is increased as the

slope parameter is increased for each experiment. It also illustrates that these algorithms tend to converge to the maximum error faster in much lower iterations.

Apart of that, the classification performance shows a varied correct classification percentage of standard BP and IeBP. The IeBP ( $\beta = 4$ ) obtains the highest classification performance at 76.09% while the other two IeBP of ( $\beta = 2,3$ ) obtain 76.07% respectively. On the other hand, the standard BP only manages to achieve 76.04% of correct classification for Balloon dataset classification problems as illustrated in figure 4.8.



**Figure 4.8: Correct Classification of V5 Trial**

#### 4.6 Result of Cancer Dataset

The network architecture used for Cancer dataset consist of 9 input nodes, 19 hidden nodes an output node. 599 training data have been used to learn the network and 100 data for testing. The experimental result of Cancer dataset for standard BP and IeBP ( $\beta = 2,3,4$ ) are presented in Table 4.3(a), (b), (c) and (d) respectively

while figure 4.9 presents the learning pattern produced by standard BP and IeBP ( $\beta = 2,3,4$ ).

**Table 4.3 (a): Result of Standard BP**

Group		Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)
Standard BP	V1	0.1	0.1	0.16601	10000	52.07
	V2	0.1	0.5	0.04998	8733	52.14
	V3	0.1	0.9	0.04991	1695	52.15
	V4	0.5	0.1	0.04997	2710	51.56
	V5	0.5	0.5	0.04982	1846	51.48
	V6	0.5	0.9	0.02894	679	51.04
	V7	0.9	0.1	0.04968	1800	52.17
	V8	0.9	0.5	0.04979	1492	52.43
	V9	0.9	0.9	0.04942	1230	52.46

**Table 4.3 (b): Result of IeBP ( $\beta = 2$ )**

Group		Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)
$\beta = 2$	V1	0.1	0.1	0.05338	10000	50.00
	V2	0.1	0.5	0.05000	5635	54.79
	V3	0.1	0.9	0.04855	1230	53.18
	V4	0.5	0.1	0.04995	2538	51.99
	V5	0.5	0.5	0.04992	9761	52.04
	V6	0.5	0.9	0.04639	1755	51.67
	V7	0.9	0.1	0.04351	2686	52.89
	V8	0.9	0.5	0.03605	1800	53.19
	V9	0.9	0.9	2.0000	10000	50.89

**Table 4.3 (c): Result of IeBP ( $\beta = 3$ )**

Group		Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)
$(\beta = 3)$	V1	0.1	0.1	0.04999	8956	52.24
	V2	0.1	0.5	0.04998	4814	52.79
	V3	0.1	0.9	0.04972	951	52.88
	V4	0.5	0.1	0.04986	2106	51.99
	V5	0.5	0.5	0.04665	1381	52.04
	V6	0.5	0.9	2.00000	10000	49.67
	V7	0.9	0.1	0.02375	1886	54.89
	V8	0.9	0.5	0.04582	1327	53.19
	V9	0.9	0.9	2.00000	10000	49.89

**Table 4.3 (d): Result of IeBP ( $\beta = 4$ )**

Group	Learning rate, $\eta$	Momentum term, $\alpha$	Error convergence	Learning iteration	Classification (%)	
$(\beta = 4)$	V1	0.1	0.1	0.05000	7409	52.56
	V2	0.1	0.5	0.04999	4022	52.96
	V3	0.1	0.9	0.04972	813	52.94
	V4	0.5	0.1	0.04986	1853	52.06
	V5	0.5	0.5	0.04477	1327	52.66
	V6	0.5	0.9	0.20000	10000	49.67
	V7	0.9	0.1	0.04975	1694	52.89
	V8	0.9	0.5	0.50024	10000	48.19
	V9	0.9	0.9	0.20000	10000	49.54

From the table 4.3 above, shows that the results for all trial for each experiment are almost similar to each other. Standard BP could give a better result at V9 trial compared to other IeBP in term of learning iterations but not at V1 trial. The table above shows that the errors converge nearly comparable to each other. As for standard BP, all the trial pair except V1 trial does converge to solution within the maximum error. The errors generated converge closely match the maximum error function specified earlier.

However, for IeBP( $\beta = 2,3,4$ ), the V9 trial could not converge to solution and tend to trap in local minima. The same goes for ( $\beta = 2$ ) at V1 trial, IeBP( $\beta = 3$ ) at V6, V9 and IeBP( $\beta = 4$ ) at V6, V8, V9 trial pairs. These trial pairs could not converge to solution within 10, 000 iterations. The IeBP( $\beta = 2,3,4$ ) has the fastest learning speed at 1230, 951 and 813 respectively for V3 trial.

Figure 4.10: Classification Accuracy of Cancer Dataset

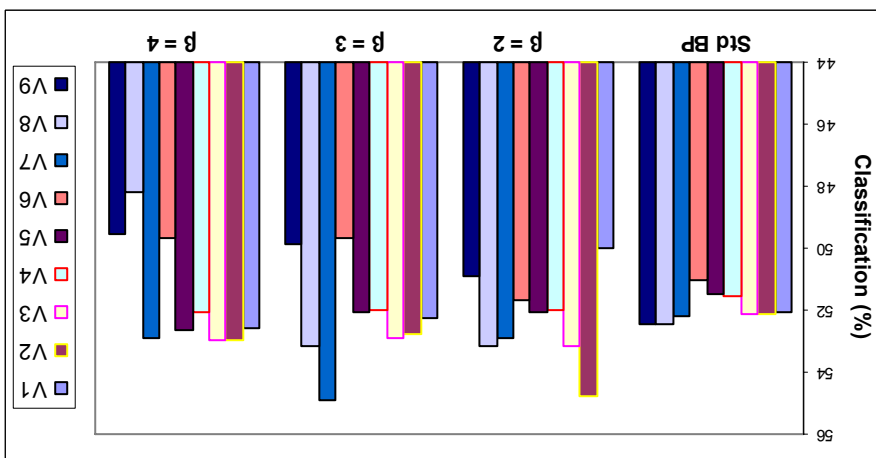
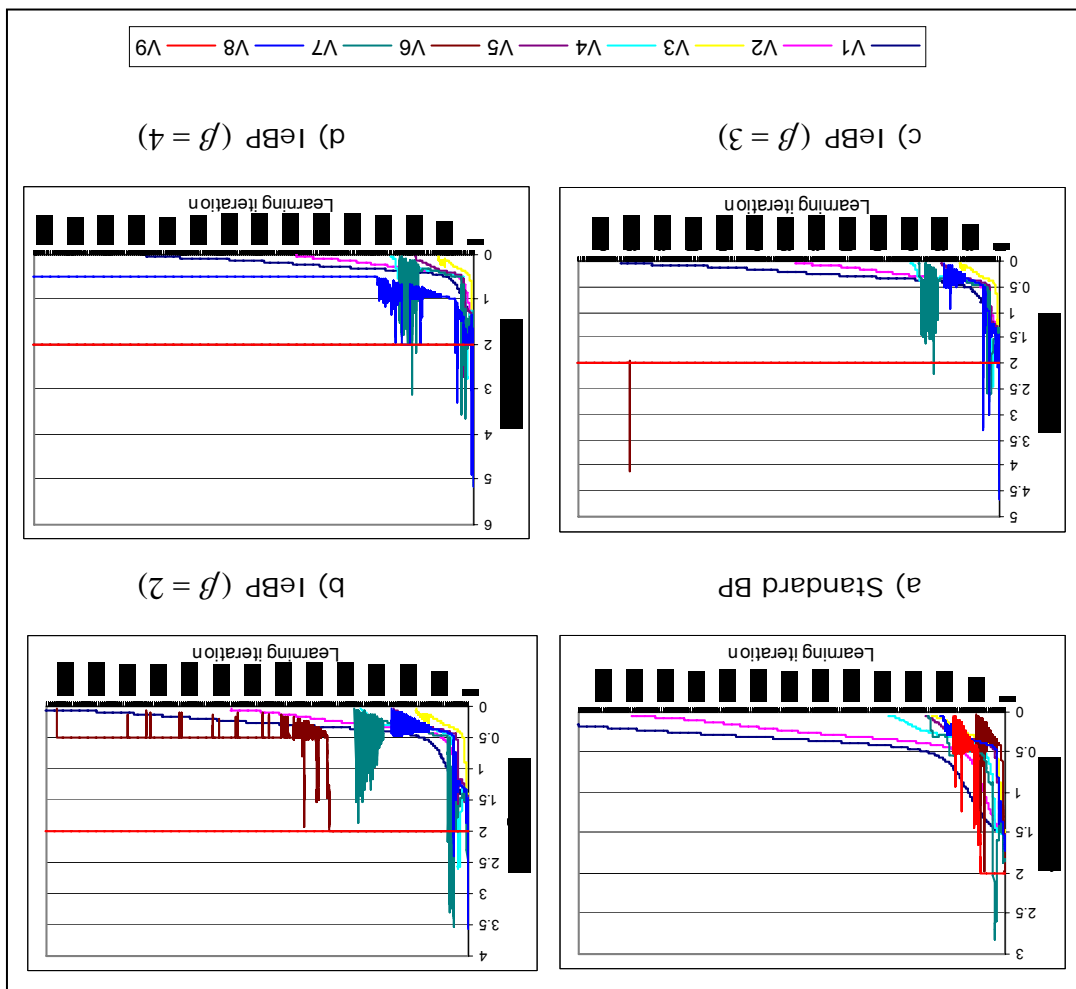
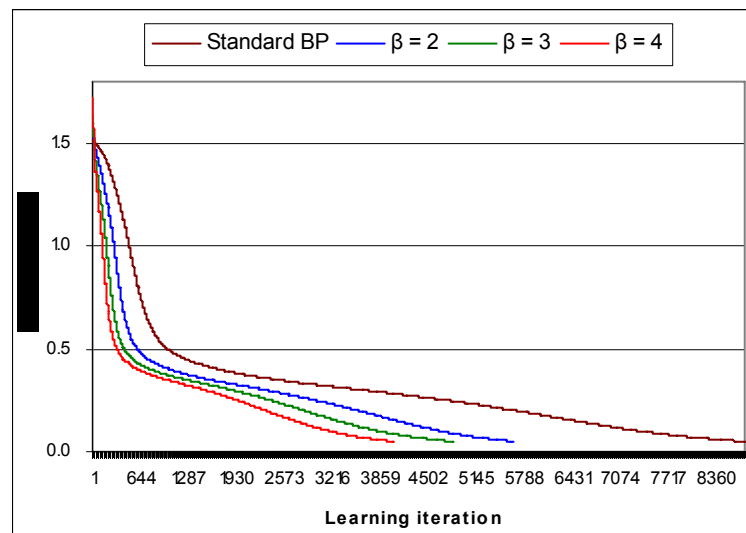


Figure 4.9: Convergence of Cancer Dataset for Standard BP and IeBP ( $\beta = 2,3,4$ )



The figure 4.10 above shows the classification accuracy for every trial of the standard BP and IeBP ( $\beta = 2,3,4$ ). In general, every trial pair performs fairly for every experiment. The standard BP has the highest rate using the V8 and V9 trial pair but V6 being worse for classification performance.

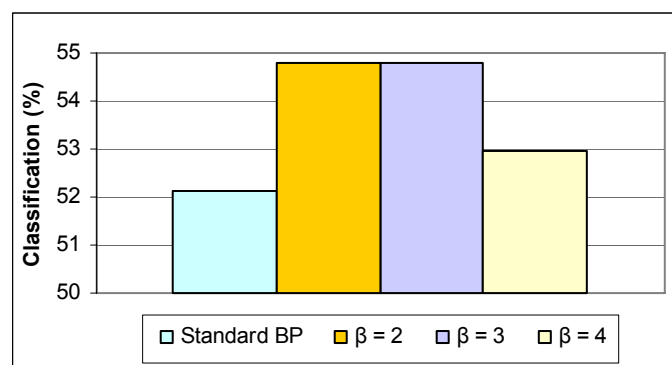
The trial pair of V2 gains better accuracy in IeBP ( $\beta = 2$ ) and V7 trial pair in IeBP ( $\beta = 3$ ). However, the V3 and V7 achieve comparable accuracy using IeBP ( $\beta = 4$ ). V3 trial pair performs comparable for each IeBP. The trial pair of V1 performs the worse in IeBP ( $\beta = 2$ ) and V6 trial pair in IeBP ( $\beta = 3$ ). The V8 trial pair gain a low classification performance in IeBP ( $\beta = 4$ ). From the analysis above, only a single result is taken which is considerably good over all trials. The V2 trial consists of learning rate 0.1 and momentum term 0.5 shows the learning speeds become faster as the value of  $\beta$  parameter are increased. Thus, the result produced by trial V2 is chosen for achieving highest convergence rate for IeBP and depicted in figure 4.11 below.



**Figure 4.11: Convergence Characteristic of V2 Trial**

The figure above shows the convergence characteristic of standard and IeBP ( $\beta = 2,3,4$ ) for Cancer dataset. It is clear that the learning speed is increased as the slope parameter is increased for each experiment. It also illustrates that these algorithms tend to converge to the maximum error faster in much lower iterations.

Apart of that, the classification performance shows a varied correct classification percentage of standard BP and IeBP. The IeBP ( $\beta = 2,3$ ) obtains the highest classification performance at 54.79% while IeBP ( $\beta = 4$ ) obtain 52.96% of classification rate respectively. On the other hand, the standard BP only manages to achieve 52.14% of correct classification for Cancer dataset classification problems as illustrated in figure 4.12 using V2 trial pair.



**Figure 4.12: Correct Classification of V2 Trial**

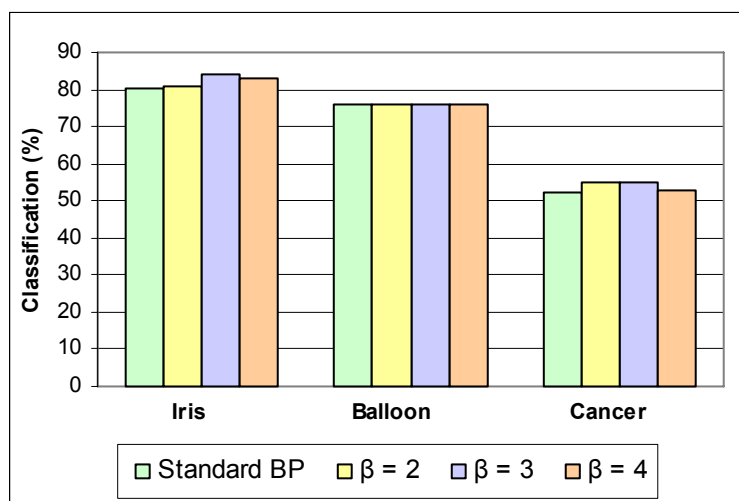
## 4.7 Discussion

Referring to the objectives, the experiments are carried out to analyze the affect of  $\beta$ , slope parameter in the improved two-term BP (IeBP) in classification problems. After conducting the experiment, the results are compared and analyzed based on the objectives. Table 4.4 below gives the summary of the classification

accuracy of each dataset compared to standard BP and IeBP ( $\beta = 2,3,4$ ) and figure 4.13 depicts the graph for each dataset.

**Table 4.4: Summary of Classification**

Dataset	Classification Accuracy (%)			
	Std BP	( $\beta = 2$ )	( $\beta = 3$ )	( $\beta = 4$ )
Iris	80.35	80.87	83.87	82.99
Balloon	76.04	76.07	76.07	76.09
Cancer	52.14	54.79	54.79	52.96



**Figure 4.13: Correct Classification for Iris, Balloon and Cancer Dataset**

For Iris dataset, it shows that the IeBP ( $\beta = 3$ ) gain a better classification accuracy than ( $\beta = 4$ ) 83.87% compare to 82.99%. As for Balloon, there is a slight difference between standard BP and IeBP. However, the IeBP ( $\beta = 4$ ) gives the correct classification at 76.09% compared to other algorithms. Meanwhile for Cancer dataset, the IeBP ( $\beta = 2,3$ ) gives the same correct classification at 54.79% compared to IeBP ( $\beta = 4$ ) at 52.96%. However, the standard BP only manages to gain 52.14% of correct classification rate. It can be concluded that a faster convergence rate algorithm does not guarantee in achieving high accuracy of the classification problems.

**Table 4.5: Summary of Results**

<b>Analysis Criteria</b>	<b>Iris</b>	<b>Balloon</b>	<b>Cancer</b>
Convergence Speed	IeBP ( $\beta = 4$ )	IeBP ( $\beta = 4$ )	IeBP ( $\beta = 4$ )
Iteration	2272	32	4022
Error Convergence	0.04996	0.04757	0.04999
Classification Performance	IeBP ( $\beta = 3$ ) 83.87%	IeBP ( $\beta = 4$ ) 76.09%	IeBP ( $\beta = 2,3$ ) 54.79%

Based on the first analysis criteria, it is proven that the IeBP ( $\beta = 4$ ) outperforms better than the others for all dataset in term of the convergence speed. The error convergence for each dataset also shows that the IeBP ( $\beta = 4$ ) could convergence to match the maximum error as defined earlier. The implementation of IeBP ( $\beta = 4$ ) performs much faster than the other IeBP as the slope parameter,  $\beta$  expected to give better results.

The speed of the error convergence is because of the implicit function of the improved two-term error function (IeBP). The implicit function is mathematically proven to produce better result than the explicit error because of the details task in the IeBP itself. In additional, training the standard BP which employed explicit mean square error (MSE) function finds a comfortable local minimum and refuses to move beyond it, and could cause instability of the internal structure of the network [10]. For that reason, the IeBP outperforms better than the standard BP.

In term of classification performance, the slope parameter,  $\beta$  do perform in a mixture pattern. As for Iris dataset, the IeBP ( $\beta = 3$ ) obtain 83.87%, for Balloon dataset; IeBP ( $\beta = 4$ ) obtain 76.09% and for Cancer dataset IeBP ( $\beta = 2,3$ ), it performs at 54.79%. However, it is not necessarily the IeBP have to produce a good classification rate. The IeBP does not work well if the dataset itself could not converge properly. This can be seen at the classification performance for Iris and Cancer dataset. This result is contradictory to the expected IeBP ( $\beta = 4$ ) tuning

would perform better as the slope parameter increased. This may be caused by some reasons and one of it is by using the unsuitable learning rate and momentum term. As for this study, the suggested value of [41][42] have been used to conduct the experiments.

#### **4.8 Summary**

This chapter explained the experimental result and performance analysis that has been carried out in this study. The implementation of the modified error function with slope parameter tuning has been done on Iris, Balloon and Cancer dataset. The analysis has been done by comparing the algorithms individually on each implementation. Also analyses and discussion has been carried out to elaborate more on the performance and realization of the algorithms based on each dataset and algorithms.

## **CHAPTER 5**

### **CONCLUSION**

#### **5.1 Introduction**

This study has been conducted to validate the performance of the Standard BP and IeBP. The previous research works and related literatures are reviewed to have a better understanding of the related research areas. Thus, this chapter will discuss about the summary of work that have been done to accomplish the objective of this study. Besides, this chapter also discusses the suggestion for future work as well as the conclusion for this study.

#### **5.2 Review**

BP algorithm has been widely recognized and be one of the effective learning method for many real world applications. This is due to fact that BP algorithm could be applied to many application fields, capable to approximate accuracy at any degree

and has some limitations such as existence of temporary instability; converge to local minima, slow rates of convergence, poor scaling properties, instability of model and many more. Therefore many modifications on standard BP have been proposed in the effort to overcome these limitations and to improve its performances.

Besides these modifications, these algorithms also could be applied to different  $\beta$  parameter systems to ensure it achieve best results of the convergence result in much shorter time. Many studies have been done by applying different method of error to the standard BP. Hence, these studies do not reveal much the real performance for both algorithms based on a specific dataset or problems.

Based on the experiments in chapter 4, the convergence rate and classification performance for the standard BP can still be improved by using other methods. Therefore, the improved two-term BP or shortly IeBP has been proposed. This improved error function is used to replace the typical MSE error function. But still, the performance of both BP algorithms can be improved a way better by taking other parameter in account. This proposed method helped in term of the convergence rate and gives better output in the classification problems.

## 5.2 Summary of Work

This study has been evolved to this far and several works have been done to make this study a success. The following summarizes the work that has been carried out for this study:

- Derived the Standard BP algorithm
- Derived the improved two-term BP for  $\beta = 2, 3, 4$

- Finalize the network structure
- Finalize the testing data (three set of universal data)
- Perform the experiments by using the dataset on
  - Standard BP
  - IeBP ( $\beta = 2, 3, 4$ )
- Capture the results and perform analysis and discussion

### 5.3 Conclusion

Based on the experiments, it can be concluded that:

- The IeBP algorithm converge significantly faster and moderate accuracy than the standard BP. This indicates that the modified error function is way efficient and the best approach that can be used to speed up the convergence and avoid local minima problem.
- The IeBP algorithm gives good classification performance compared to standard BP algorithm.
- The selection of various combinations of the control parameter affects the performance and efficiency of IeBP in solving a specific problem.
- An appropriate network architecture and selection of network parameters for the dataset will influence the convergence and the performance of network learning.

## 5.4 Recommendation for Further Study

In term of BP algorithms, there are several suggestions that could be made to improve the algorithm for future work.

- Use an optimization method to tune the various neural network parameters (e.g.: initial weight, maximum error with learning rate, momentum term) to the optimal value.
- Use and test using different pair of learning rate and momentum term.
- Opt for other activation function such as Arctangent and Logarithmic function on the BP training and investigate the performance.
- Apply other error functions than MSE such as cross-entropy and tanh to make performance comparison with the IeBP.

## REFERENCES

- [1] Y.L. Murphey and Y. Luo, Feature Extraction for a Multiple Pattern Classification Neural Network System, *Pattern Recognition Proc.*, Vol. 2, pp. 220-223, 2002.
- [2] M. Nikoonahad and D.C. Liu, Medical Ultra Sound Imaging Using Neural Networks, *Electronic Letters*, vol. 2, no. 6, pp. 18-23, 1990.
- [3] Rumelhart, D.E., and McClelland J.L., (1986) *Parallel and Distributed Processing: Explorations in the Micro Structure of the Cognition*, Vol.1. MIT Press, Cambridge, Mass
- [4] T.J. Sejnowski and C.R. Rosenberg, "Parallel Networks that Learn to Pronounce English Text," *Complex Systems*, vol. 1, pp. 145-168, 1987.
- [5] K.S. Narendra and K. Parthasarathy, "Identification and Control of Dynamical Systems Using Neural Network," *IEEE Trans. Neural Networks*, vol. 1, pp. 4-27, 1990.
- [6] Ana Salwa (2005) *Improved Two-term Backpropagation Error Function with GA-based Parameter Tuning for Classification Problem*, UTM: Master Thesis
- [7] Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*. 2<sup>nd</sup> ed. Upper Saddle River, New Jersey: Prentice Hall
- [8] Zweiri, Y.H., Whidborne, J.F., Althoefer, K. and Seneviratne, L.D. (2003). A Three-term Backpropagation Algorithm. *Neurocomputing*. 50:305-318.

- [9] Oh, S-H, (1997). Improving the Error Backpropagation Algorithm with a Modified Error Function. *IEEE Transactions on Neural Networks*. 8:799-803.
- [10] Shamsuddin, S.M., Sulaiman, M.N and Darus, M. (2001). An Improved Error Signal for Backpropagation Model for Classification Problems. *Intern. J. Computer Mathematics*. 76(1-2):297-305
- [11] Fnaiech, F., Abid, S., Ellala N. and Cheriet, M. (2002). A comparative study of fast neural networks learning algorithms. *IEEE International Conference on Systems, Man and Cybernetics 2002*, 6-9 Oct. 2002. IEEE AMC, 2002. 24-29
- [12] Jiang, M., Deng B., Gielen, G., Tang, X., Ruan, Q. and Yuan, B. (2002). A fast learning algorithm if feedforward neural networks by using novel error functions. *6<sup>th</sup> International Conference on Signal Processing 2002*. 26-30 Aug. 2002. IEEE. 2002. 1171-1174
- [13] Yu, C-C and Liu, B-D (2002). A backpropagation algorithm with adaptive learning rate and momentum coefficient. *Proc. Of the 2002 International Joint Conference on Neural Networks*. 12-17 May 2002. IEEE 2002 1218-1223
- [14] S. Bengio, Y. Bengio, J. Cloutier, Use of genetic programming for the search of a new learning rule for neural networks, *Proceedings of the First IEEE World Congress on Computational Intelligence and Evolutionary Computation*, Orlando, FL, 27–29 June, 1994, pp. 324–327.
- [15] C.B. Owen, A.M. Abunawass, Application of simulated annealing to the backpropagation model improves convergence, *Proceedings of the SPIE Conference on the Science of Artificial Neural Networks, Vol. II, 1966*, Orlando, FL, 13–16 April 1993, pp. 269–276.
- [16] A. Von Lehmen, E.G. Paek, P.F. Liao, A. Marrakchi, J.S. Patel, Factors in Uuencing learning by backpropagation, *Proceedings of the IEEE*

*International Conference on Neural Networks, Vol. I, San Diego, CA, 1988, pp. 335–341.*

- [17] C. Wang, J.C. Principe, Training neural networks with additive noise in the desired signal, *IEEE Trans. Neural Networks* 10 (6) (1999) 1511–1517.
- [18] Kamruzzaman, J. and Aziz, S.M. (2002). A note on activation function in multilayer feedforward learning. *Proc. Of International joint Conference on Neural Network 2002*. IEEE. 2002. 519-523
- [19] Falas, T. and Stafylopatis, A.G. (1999). The Impact of the Error Function Selection in Neural Network-based Classifiers. *International Joint Conference on Neural Network 3*. 10-16 July 1999. IEEE. 1999. 1799-1804.
- [20] Chow, M-Y., Goode, P., Menozzi, A., Teeter, J. and Thrower, J.P. (1994). Bernoulli Error Measure Approach to Train FeedForward Artificial Neural Network for Classification Problem. *IEEE International Conference on Neural Networks*. 27 June-2 July 1994. IEEE. 1994. 44-49
- [21] Shamsuddin, S.M., Jaaman, S.H, Majid, N. and Ismail, N. (2002). Improved Backpropagation Model for Classification on Profitability Analysis Data: A Study on Malaysian Market. *Chiang Mai J. Sci.* 29(1): 35-41.
- [22] M.P. Craven. “A Faster Learning Neural Network Classifier Using Selective Backpropagation”. *Proc. of the 4th IEEE International Conference on Electronics, Circuits and Systems* Cairo, Egypt, 1:254-258, Dec. 1997.
- [23] Fredic M. Ham and Ivica Kostanic *Principles of neurocomputing for science and engineering* McGraw Hill, Inc
- [24] Anastasio, T.J. (1993). *Modelling Vestibulo-Ocular Reflex Dynamics: From Classical Analysis to neural Networks*. F. Eeckman, Ed, Neural Systems: Analysis and Modelling. Norwell, MA: Kluwer, pp. 407-430

- [25] Sterling, P. (1990). *Retina. The Synaptic Organization of the Brain*. G.M. Shepard, ed., 3<sup>rd</sup> Edition, New York: Oxford University Press, pp.170-213
- [26] Mead, C.A. (1989). *Analog VLSI and Neural Systems*. Reading, M.A: Addison-Wesley
- [27] Mahowald, M.A. and Mead, C. (1989). *Silicon Retina*. Analog VLSI and Neural Systems (C. Mead), Chapter 15. Reading, MA: Addison-Wesley.
- [28] Boahen, K.A. and Andreou, A.G (1992). *A Contrast Sensitive Silicon Retina with Reciprocal Synapses*. Advances in Neural Information Pprocessing Systems. San Matao, CA: Morgan Kaufmann. Vol. 4, pp. 764-772
- [29] Boahen, K.A. (1996). *A Retinomorphic Vision System*. IEEE Micro. Vol. 16, num. 5, pp. 30-39.
- [30] Narayanan, S. (1997). The Generalized Sigmoid Activation Function: Competitive Supervised Learning. *Information Sciences*. 99:69-82
- [31] Negnevitsky, M. and Ringrose, M. (1999). Accelerated learning in multi-layer neural networks. *Proc. Of 6<sup>th</sup> International Conference on Neural Information Processing*. 16-20 Nov 1999. IEEE. 1999. 1167-1171
- [32] Barryl. Kalman, Sahunny Johnson and Stan C. Kwasny, (1991). An Adaptive Neural Network Parser.
- [33] Qingwu, M., Chengbin, L., Hu, C. L., Improved BP Network Algorithm Based on Fuzzy Logic and Application in Geophysics *Proceedings European Symposium on Intelligent Techniques (ESIT 2000)*, Aachen, Germany, September 2000 pp. 437- 444
- [34] Wang, X.G., Tang, Z., Tamura, H. and Ishii, M. (2004). A Modified Error Function for the Backpropagation Algorithm. *Neurocomputing*. 57: 477-484

- [35] Mohammed A. Otair., Walid A. Salameh, Speeding Up Back-Propagation Neural Networks *Proceedings of the 2005 Informing Science and IT Education Joint Conference* Arizona, USA, June 16-19 pp. 167-173
- [36] N.M. Nawi, R.S. Ransing and M.R. Ransing (2008). An Improved Conjugate Gradient Based Learning Algorithm for Back Propagation Neural Networks. *International Journal of Computational Intelligence 2008* pp. 46-55
- [37] Kim, G-H., Yoon, J.E., An, S-H., Cho, H-H. and Kang, K-I. (2004). Neural Network Model Incorporating A Genetic Algorithm in Estimating Construction Cost. *Building and Environment*. 39(11):1333-1340
- [38] Dr. Siti Mariyam Hj Shamsuddin (2004). *Lecture Note of Advanced Artificial Intelligence: Number of Hidden Neurons*. UTM. Unpublished...
- [39] Charytoniuk, W. and Chen, M.S (2000). Neural Network Design for Short-term Load Forecasting. *International Conference on Electronic Utility Deregulation and Restructuring and Power Technologies 2000*. 4-7 April 2000. City University, London, 554-561
- [40] Alpsan, D., Towsey, M., Ozdamar, O., Tsoi, A.C. and Ghista, D.N. (1995). Efficiency of Modified Backpropagation and Optimisation Methods on Real-World Medical Problem. *Neural Network*. 8(6): 945-962
- [41] Sharda, R. and R. Patil (1992). Connectionist Approach to Time Series Prediction: An Empirical Test. *Journal of Intelligent Manufacturing*, 3, pp. 317-23
- [42] Zhang, G., Patuwoo, B.D. and Hu, M.Y. (1998). Forecasting with artificial neural network: The state of the art. *International Journal of Forecasting*. 14: pp. 35-62