IMPROVED SPIKEPROP ALGORITHM FOR NEURAL NETWORK LEARNING

FALAH.Y.H.AHMED

A thesis submitted in fulfilment of the requirements for the award of the degree of Doctor of Philosophy (Computer Science)

Faculty of Computing
Universiti Teknologi Malaysia

MAY 2013
To my beloved father and mother.
ACKNOWLEDGEMENT

In the Name of Allah, Most Gracious, Most Merciful

All praise and thanks are due to Allah, and peace and blessings be upon his messenger, Mohammed (peace be upon him).

Alhamdulillah, it is with Allah S.W.T will that I get to finish this thesis in the time given. Here, I would like to express my heartfelt gratitude to my supervisor Professor Dr. Siti Mariyam Shamsuddin and without her guidance and advice this study would not have been possible. She has been incredibly wise, helpful, understanding, and generous throughout the process. She has truly been a mentor and I owe here my deepest thanks. Also I would like to express my heartfelt to my co-supervisor Assoc. Prof. Dr. Siti Zaiton Mohd Hashim for her support and guidance during my study. And I would like to thanks to my External supervisor Prof. Dr. Nikola Kasabov for his support and guidance during my study. And finally I would like to thanks Dr. Haza Nuzly Bin Abdull Hamed for advices and guidance.

I have made many friends during my time in UTM and I thank them for their support and encouragement. Their views and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space.

A lot of information useful to the work was found via the World-Wide Web; I thank those who made their materials available by means of this medium and those who kindly answered back to my roll-calls of help sent over the World-Wide Web.
ABSTRACT

Spiking Neural Network (SNN) utilizes individual spikes in time domain to communicate and to perform computation in a manner like what the real neurons actually do. SNN had remained unexplored for many years because it was considered too complex and too difficult to analyze. Since Sander Bothe introduced SpikeProp as a supervised learning model for SNN in 2002, many problems which were not clearly known regarding the characteristics of SNN have now been understood. Despite the success of Bohte in his pioneering work on SpikeProp, his algorithm is dictated by fixed time convergence in the iterative process to get optimum initial weights and the lengthy procedure in implementing the sequence of complete learning for classification purposes. Therefore, this thesis proposes an improvement to Bohte’s algorithm by introducing acceleration factors of Particle Swarm Optimization (PSO) denoted as Model 1; SpikeProp using \( \mu \) Angle driven Learning rate dependency as Model 2; SpikeProp using Radius Initial Weight as Model 3a, and SpikeProp using Differential Evolution (DE) Weights Initialization as Model 3b. The hybridization of Model 1 and Model 2 gives Model 4, and finally Model 5 is obtained from the hybridization of Model 1, Model 3a and Model 3b. With these new methods, it was observed that the errors can be reduced accordingly. Training and classification properties of the new proposed methods were investigated using datasets from Machine Learning Benchmark Repository. Performance results of the proposed Models (for which graphs of time errors with iterative timings, table of number of iterations required to reduce time error measurement to saturation level and bar charts of accuracy at saturation time error for all the datasets have been plotted and drawn up) were compared with one another and with the performance results of Standard SpikeProp and Backpropagation (BP). Results indicated that the performances of Model 4, Model 5 and Model 1 are better than Model 2, Model 3a and Model 3b. The findings also reveal that all the proposed models perform better than Standard SpikeProp and BP for all datasets used.
ABSTRAK

Rangkaian Saraf Pepaku (SNN) menggunakan dedenyut tunggal dalam domain masa untuk mewujudkan komunikasi dan penghitungan seperti yang dilakukan oleh saraf tabii. SNN tidak diterokai dengan mendalam disebabkan oleh mekanisme pelaksanaannya yang amat sukar untuk dianalisa. Sejak diperkenalkan oleh Sander Bothe pada tahun 2002 sebagai model pembelajaran terpandu, banyak masalah yang tidak jelas pada masa lampau berkenaan dengan ciri-ciri SNN sudah boleh diperjelaskan pada masa kini. Walaupun SpikeProp sudah berjaya digunakan secara meluas sorotan kejayaan Bothe terhadap algoritma Spikeprop, algoritma ini masih lagi mempunyai masalah dalam konteks tetapan masa penumpuan dalam proses jujukan bagi mendapatkan pemberat awalan yang optimal dan prosedur yang lama untuk melaksanakan pembelajaran lengkap bagi tujuan pengelasan. Oleh yang demikian, tesis ini mencadangkan perbaikan terhadap algoritma Bohte dengan memperkenalkan faktor pecutan dalam SpikeProp menggunakan PSO dan dilabel sebagai Model 1; SpikeProp menggunakan Kadar Pembelajaran Sudut Terpandu Bersandarkan $\mu$, Model 2; SpikeProp menggunakan Pemberat Awalan Jejari, Model 3a; dan SpikeProp menggunakan Pemberat Awalan Jejari Evolusi Pembahagi, Model 3b. Penhibridian Model 1 dan Model 2 memberi Model 4 dan akhirnya Model 5 dicadangkan hasil penhibridan Model 1, Model 3a dan Model 3b. Berdasarkan model baru ini, didapati bahawa ralat boleh disusutkan dengan baik dan pantas. Sifat pengelasan dan latihan bagi kaedah cadangan telah dikaji menggunakan set data daripada Storan Piawaian Pembelajaran Mesin. Prestasi keputusan bagi kaedah cadangan (iaitu dengan graf ralat masa terhadap masa lelaran, jadual bilangan lelaran yang diperlukan untuk menyusutkan pengukuran ralat masa kepada tahap tepu dan carta bar ketepatan pada ralat masa tepu bagi semua set data telah diplot) di bandingkan dengan prestasi SpikeProp and Rambatan Balik piawai. Hasil kajian menunjukkan bahawa prestasi Model 4, Model 5 dan Model 1 lebih baik berbanding Model 2, Model 3a dan Model 3b. Hasil dapatan juga mendapati bahawa prestasi ke semua model cadangan ini adalah lebih baik berbanding dengan prestasi SpikeProp and Rambatan Balik piawai bagi semua set data.
### TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DECLARATION</td>
<td>ii</td>
</tr>
<tr>
<td></td>
<td>DEDICATION</td>
<td>iii</td>
</tr>
<tr>
<td></td>
<td>ACKNOWLEDGMENT</td>
<td>iv</td>
</tr>
<tr>
<td></td>
<td>ABSTRACT</td>
<td>v</td>
</tr>
<tr>
<td></td>
<td>ABSTRAK</td>
<td>vi</td>
</tr>
<tr>
<td></td>
<td>TABLE OF CONTENTS</td>
<td>vii</td>
</tr>
<tr>
<td></td>
<td>LIST OF TABLES</td>
<td>xii</td>
</tr>
<tr>
<td></td>
<td>LIST OF FIGURES</td>
<td>xiv</td>
</tr>
<tr>
<td>1</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Overview</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>Problem Background</td>
<td>2</td>
</tr>
<tr>
<td>1.2.1</td>
<td>Spiking Neural Networks (SNNs)</td>
<td>5</td>
</tr>
<tr>
<td>1.2.2</td>
<td>Learning in Spike Neural Networks (SpikeProp)</td>
<td>6</td>
</tr>
<tr>
<td>1.3</td>
<td>Problem Statement</td>
<td>8</td>
</tr>
<tr>
<td>1.4</td>
<td>Research Objectives</td>
<td>9</td>
</tr>
<tr>
<td>1.5</td>
<td>Scope of the Study</td>
<td>11</td>
</tr>
<tr>
<td>1.6</td>
<td>Thesis Organization</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>LITERATURE REVIEW</td>
<td>14</td>
</tr>
<tr>
<td>2.1</td>
<td>Introduction</td>
<td>14</td>
</tr>
<tr>
<td>2.2</td>
<td>Artificial Neural Networks (ANNs)</td>
<td>15</td>
</tr>
<tr>
<td>2.3</td>
<td>Computational Process in ANNs</td>
<td>19</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>3.4</td>
<td>Enhancement SpikeProp Architecture by PSO (Model 1-PSOSpikeProp)</td>
<td>69</td>
</tr>
<tr>
<td>3.5</td>
<td>The Proposed Learning Rate $\mu$ for SpikeProp (Model 2)</td>
<td>74</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Proposed $\mu$ Angle Driven Dependency Learning Rate (Model 2)</td>
<td>76</td>
</tr>
<tr>
<td>3.6</td>
<td>Weights Initialization</td>
<td>77</td>
</tr>
<tr>
<td>3.6.1</td>
<td>Proposed Initial Weight for SpikeProp (Model 3a-Radius Initial Weight)</td>
<td>77</td>
</tr>
<tr>
<td>3.6.2</td>
<td>Proposed initial weight using Differential Evolution (Model 3b)</td>
<td>81</td>
</tr>
<tr>
<td>3.7</td>
<td>Encoding Input Datasets Variables</td>
<td>86</td>
</tr>
<tr>
<td>3.8</td>
<td>Encoding Output Datasets Variable</td>
<td>88</td>
</tr>
<tr>
<td>3.9</td>
<td>Encoding the XOR problem</td>
<td>88</td>
</tr>
<tr>
<td>3.10</td>
<td>Error Measurement Functions</td>
<td>89</td>
</tr>
<tr>
<td>3.11</td>
<td>Summary</td>
<td>90</td>
</tr>
</tbody>
</table>

4  
THE PROPOSED MODELS FOR SPIKEPROP LEARNING ENHANCEMENTS 91
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>91</td>
</tr>
<tr>
<td>4.2</td>
<td>Model 1- Enhancement Spikeprop Architectures by PSO (PSOSpikeProp)</td>
<td>94</td>
</tr>
<tr>
<td>4.3</td>
<td>Model 2- Spikeprop using Angle driven dependency Learning Rate</td>
<td>98</td>
</tr>
<tr>
<td>4.4</td>
<td>Model 3 - SpikeProp enhancement by Weights Initialization</td>
<td>99</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Model 3a –Radius Initialization Weight (RIW) Method</td>
<td>99</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Model 3b- DE Weights Initialization Weights</td>
<td>100</td>
</tr>
<tr>
<td>4.5</td>
<td>Model 4 - Hybridization of Model 1 with Model 2 for SpikeProp Learning Enhancement</td>
<td>102</td>
</tr>
<tr>
<td>4.6</td>
<td>Model 5 – Hybridization of Model 1, Model 3a and Model 3b for Spikeprop Learning Enhancements</td>
<td>104</td>
</tr>
<tr>
<td>4.7</td>
<td>Testing and Training</td>
<td>107</td>
</tr>
<tr>
<td>4.7.1</td>
<td>Hold-Out Validation</td>
<td>107</td>
</tr>
<tr>
<td>4.7.2</td>
<td>K-Fold Cross Validation</td>
<td>108</td>
</tr>
<tr>
<td>4.8</td>
<td>Classification Accuracy</td>
<td>109</td>
</tr>
</tbody>
</table>
5 RESULTS AND DISCUSSION

5.1 Introduction

5.2 Experimental Study
  5.2.1 Experimental Design

5.3 The Proposed Models
  5.3.1 Analysis of Standard SpikeProp
  5.3.2 Analysis of Standard BP
  5.3.3 Analysis of the Proposed Model 1: PSO- Spikeprop
  5.3.4 Analysis of the Proposed Model 2: SpikeProp with Angle driven dependency ($\mu$)
  5.3.5 Analysis of the Proposed Model 3a: Spikeprop based on Radius Initial Weight
  5.3.6 Analysis of the Proposed Model 3b: Spikeprop Learning based on Differential Evolution (DE) Weights Initialization
  5.3.7 Analysis of the Proposed Model 4: Hybridization of Model 1 and Model 2
  5.3.8 Analysis of the Proposed Model 5: Hybridization of Model 1, Model 3a and Model 3b

5.4 Analysis of the Proposed Models based on Error in Time (EiT)
  5.4.1 Encoding Error of the Proposed Models for Liver Datasets
  5.4.2 Encoding Error of the Proposed Models for Cancer Datasets
  5.4.3 Encoding Error of the Proposed Models for BTX datasets
  5.4.4 Encoding Analysis of the Proposed Models for Diabetes Datasets
  5.4.5 Encoding Analysis of the Proposed Models for Heart problem
  5.4.6 Encoding Analysis of the Proposed Models for Hepatitis Datasets
  5.4.7 Encoding Analysis of the Proposed Models for Iris Datasets
5.5 Error-in-Time Analysis of the Proposed Models based on the Iterations 138
5.6 Accuracy Analysis of the Proposed Models 141
5.7 Analysis of the Standard SpikeProp and the Proposed Models using K-Fold Cross-Validation 144
5.7.1 Analysis of the Model 1 using K-Fold Cross-Validation 145
5.7.2 Analysis of the Model 2 using K-Fold Cross-Validation 146
5.7.3 Analysis of the Model 3a using K-Fold Cross-Validation 147
5.7.4 Analysis of the Model 3b using K-Fold Cross Validation 148
5.7.5 Analysis of the Model 4 using K-Fold Cross-Validation 149
5.7.6 Analysis of the Model 5 using K-Fold Cross-Validation 150
5.8 Result and analysis comparison in terms of accuracy using K-Fold Cross-Validation 151
5.9 Summary 154

6 CONCLUSION 156
6.1 Introduction 156
6.2 Research Contributions 157
6.3 Future Work 159

REFERENCES 161
APPENDIX A-D 170-190
## LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Summary of Differences of EBP, SNN and SpikeProp</td>
<td>7</td>
</tr>
<tr>
<td>2.1</td>
<td>Different Forms of Synaptic Plasticity After (C. Koch., 1999). By ‘pre’ we Denote the Presynaptic Locus of the Phenomenon Induction, While ‘post’ Stands for the Postsynaptic Locus.</td>
<td>49</td>
</tr>
<tr>
<td>2.2</td>
<td>Algorithm SpikeProp</td>
<td>58</td>
</tr>
<tr>
<td>2.3</td>
<td>Related Works Associated with SpikeProp</td>
<td>61</td>
</tr>
<tr>
<td>3.1</td>
<td>PSO Parameter Used in SpikeProp</td>
<td>70</td>
</tr>
<tr>
<td>3.2</td>
<td>Spiking Time Patterns of XOR</td>
<td>88</td>
</tr>
<tr>
<td>4.1</td>
<td>Description of datasets</td>
<td>110</td>
</tr>
<tr>
<td>5.1</td>
<td>Results for Standard SpikeProp Algorithm Using Hold-Out Validation</td>
<td>118</td>
</tr>
<tr>
<td>5.2</td>
<td>Results for Standard BP Algorithm Using Hold-Out Validation</td>
<td>119</td>
</tr>
<tr>
<td>5.3</td>
<td>Results for Model 1: PSO-SpikeProp Using Hold-Out Validation</td>
<td>120</td>
</tr>
<tr>
<td>5.4</td>
<td>Results for Model 2: SpikeProp Based Angle Driven Dependency</td>
<td>121</td>
</tr>
<tr>
<td>5.5</td>
<td>Results of Model 3a: SpikeProp Based on Radius Initial Weight</td>
<td>123</td>
</tr>
<tr>
<td>5.6</td>
<td>Results of Model 3b: SpikeProp Based on DE Weights Initialization</td>
<td>124</td>
</tr>
<tr>
<td>5.7</td>
<td>Results for Model 4: Hybridizing of Model 1 and Model 2</td>
<td>125</td>
</tr>
<tr>
<td>5.8</td>
<td>Model 5: Hybridization of Model 1, Model 3a and Model 3b</td>
<td>126</td>
</tr>
<tr>
<td>5.9</td>
<td>Analysis of Error-in-Time</td>
<td>140</td>
</tr>
<tr>
<td>5.10</td>
<td>Number of iterations</td>
<td>140</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>5.11</td>
<td>Training Accuracy Using Hold-Out Validation</td>
<td>142</td>
</tr>
<tr>
<td>5.12</td>
<td>Testing Accuracy using Hold-Out Validation</td>
<td>143</td>
</tr>
<tr>
<td>5.13</td>
<td>Results for Standard SpikeProp for 10-Fold Cross-Validation</td>
<td>145</td>
</tr>
<tr>
<td>5.14</td>
<td>Results for Model 1 for 10-Fold Cross-Validation</td>
<td>146</td>
</tr>
<tr>
<td>5.15</td>
<td>Results for Model 2 for 10- Fold Cross-Validation</td>
<td>147</td>
</tr>
<tr>
<td>5.16</td>
<td>Results of Model 3a for 10- Fold Cross-Validation</td>
<td>148</td>
</tr>
<tr>
<td>5.17</td>
<td>Results of Model 3b for 10- Fold Cross-Validation</td>
<td>149</td>
</tr>
<tr>
<td>5.18</td>
<td>Results of the Model 4 for 10- Fold Cross-Validation</td>
<td>150</td>
</tr>
<tr>
<td>5.19</td>
<td>Results of the Model 5 for 10- Fold Cross-Validation</td>
<td>151</td>
</tr>
<tr>
<td>5.20</td>
<td>Training Accuracy for 10- Fold Cross-Validation</td>
<td>152</td>
</tr>
<tr>
<td>5.21</td>
<td>Testing Accuracy for 10- Fold Cross-Validation</td>
<td>153</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Artificial Neural Networks</td>
<td>16</td>
</tr>
<tr>
<td>2.2</td>
<td>Back propagation neural Network composed of three layers</td>
<td>22</td>
</tr>
<tr>
<td>2.3</td>
<td>Sigmoid function Activation function</td>
<td>23</td>
</tr>
<tr>
<td>2.4</td>
<td>The Forward Pass Backward pass of error back propagation network</td>
<td>28</td>
</tr>
<tr>
<td>2.5</td>
<td>PSO flowchart</td>
<td>34</td>
</tr>
<tr>
<td>2.6</td>
<td>Particle movement in 2D space problem</td>
<td>36</td>
</tr>
<tr>
<td>2.7</td>
<td>A diagram of neural network learning using PSO (Van den Bergh, 2001)</td>
<td>37</td>
</tr>
<tr>
<td>2.8</td>
<td>Main stages of the DE algorithm</td>
<td>38</td>
</tr>
<tr>
<td>2.9</td>
<td>spiking neural network (Sander Bohte, 2003)</td>
<td>40</td>
</tr>
<tr>
<td>2.10</td>
<td>Integrate-and-fire neuron. (Nikola Kasabov,2012)</td>
<td>42</td>
</tr>
<tr>
<td>2.11</td>
<td>Typical behaviour of a leaky integrate-and-fire neuron (rise and decay terms).</td>
<td>44</td>
</tr>
<tr>
<td>2.12</td>
<td>A) Feed Forward Spiking Neural Network B) Connection Consisting of Multiple Delayed synaptic Terminals.</td>
<td>51</td>
</tr>
<tr>
<td>2.13</td>
<td>Connections in a Feed Forward Spiking Neural Network: A) Single Synaptic Terminal: The Delayed Pre-Synaptic Potential is Weighted by the Synaptic Efficacy to Obtain the Post-Synaptic Potential. B) Two Multi-Synapse Connections. Weighted Input is Summed at the Target Neuron</td>
<td>52</td>
</tr>
<tr>
<td>2.14</td>
<td>Relationship Between $\delta x_j$ and $\delta t_j$ for an Space Around $t_j$</td>
<td>55</td>
</tr>
<tr>
<td>3.1</td>
<td>Operational Framework</td>
<td>67</td>
</tr>
<tr>
<td>3.2</td>
<td>Model 1-PSOSpikeprop Learning Process</td>
<td>73</td>
</tr>
<tr>
<td>3.3</td>
<td>Flowchart for the Differential Evolution created Initialization weights (Model 3b)</td>
<td>85</td>
</tr>
</tbody>
</table>
3.4 Population Encoding Method. Redrawn from 87
4.1 Operational Framework for Models 93
4.2 PSOSpikeProp Procedure 97
4.3 Hidden and Output Weights Matrix in Radius Initialization Weight (RIW) 100
4.4 DE Weights Initialization procedure 101
4.5 Flowchart of Model 4 104
4.6 Flowchart of Model 5 106
4.7 K-fold cross validation 109
5.1 Comparison Analysis of the Proposed Models for Liver Dataset 128
5.2 Comparison Analysis of the Proposed Models for Cancer Dataset 130
5.3 Comparison Analysis of the Proposed Models for BTX Dataset 132
5.4 Comparison Analysis of the Proposed Models for Diabetes Dataset 133
5.5 Comparison Analysis of the Proposed Models for Heart Dataset 135
5.6 Comparison Analysis of the Proposed Models for Hepatitis Dataset 136
5.7 Comparison Analysis of the Proposed Models for Iris Dataset 138
5.8 Comparison of the Proposed Methods and SpikeProp in Terms of Accuracy of Training 142
5.9 Comparison of the proposed Methods and in Terms of Accuracy of Testing 143
5.10 Accuracy of the Proposed Methods and SpikeProp for Training in K-Fold Cross-Validation 153
5.11 Accuracy of the Proposed Methods and SpikeProp for Testing in K-Fold Cross-Validation 154
CHAPTER 1

INTRODUCTION

1.1 Overview

Classification is one of the most frequently encountered processing tasks for a decision support system. Classification is necessary when a scattering of data needs to be sorted into the predefined groups or classes based on certain criteria and attributes. Classification problems are encountered in areas such as business, science, industry and medicine. They include bankruptcy prediction, credit scoring, medical diagnosis, quality control, handwritten character recognition, and speech recognition.

Traditional statistical classification procedures such as discriminant analysis are built on the Bayesian decision theory (Duda and Hart, 1973; Qasem and Shamsuddin, 2010) In these procedures, an underlying probability model must be assumed in order to calculate the posterior probability upon which the classification decision is made. One major limitation of the statistical models is that they work well only when the underlying assumptions are satisfied. The effectiveness of these methods depends to a large extent on the various assumptions or conditions under which the models are developed. Users must have a good knowledge of both data properties and model capabilities before the models can be successfully applied. Neural networks have emerged as an important tool for classification because they do not depend much on prior knowledge of the statistics of the data.
Several neurobiologists believe that the working process of a brain is similar to that of a huge parallel computer which has at its disposal about 10 billion simple processors, where each needs a few milliseconds to react according to input values. There are similarities between the function of parallel processing methods and conventional Artificial Neural Network (ANN) methods in many real-time applications.

For this reason ANN has been conceived with a structure which resembles closely to the human brain (DKlerfors, 2004). Because of this structure, computation property of ANN has been designated many names which include parallel distributed processing, neuron-computing, natural intelligent systems and machine learning algorithms.

1.2 Background of the Problem

The earliest models and schemes of ANN (the first generation ANNs) came nearly fifty years ago. In the earlier models which are theoretically incredibly straightforward, the primary generation of ANN consists of McCulloch-Pitts threshold neurons (Y.-J. Jin et al., 2013; Maass, 1997). A strand of neurons sends a binary signal (higher value signal) only if the summation of the weighted signals which it receives rises above a certain threshold level. As the output from these neurons are digital, it is convenient to treat interconnected neurons to behave as a function with a Boolean output type which can be modeled mathematically as multi-layer perceptron (MLP) and Hopfield Nets (Rebizant, 2011).

Signal responses of second generation ANN are not calculated using threshold or step activation type functions. Instead, they are calculated using continuous activation type functions which are formulated so that responses are analog for the two cases: input and output. Generally, the activation functions take the form of
the sigmoid. Sigmoidal Neural Nets, (these include Adaline Perceptrons and Radial Basis Function) have been considered under the class of second generation ANNs

Back propagation (BP) and recurrent neural networks have been applied to many applications. They are the most famous instances of neural networks consisting of neurons which make up the second generation of ANN. Obviously they outperform the first generation of ANN which are merely simple networks of binary neural units, each exhibiting either an active or an inactive, firing or non-firing states.

ANN can perform digital computations to get universal functions approximation which can also be calculated using fewer neurons in a network of the first generation whereby the corresponding circuits are resolved with Boolean threshold gates (Y.-J. Jin, et al., 2013; Maass et al., 1991).

Usually, the learning process for updating the weights of ANN interconnectors will require some form of optimization. One useful optimization technique uses PSO (Eberhart and Kennedy, 1995), which is inspired by how the biological swarm of animals works to achieve a desirable objective for the group. Since its introduction, PSO has been widely used to solve many real world problems. (X. Jin, 2011; Kennedy and Eberhart, 1997) also introduced the binary version of PSO. There have been a lot of developments and improvements in this area (Khanesar et al., 2007) (Srinivasan and Shanmugalakshmi., 2012; Yuan et al., 2009). PSO has been applied to derive universal function approximations for any analog function with random updating of weights.

For the first generations of ANN, the neurons are restricted to binary inputs and outputs. These binary impulses have a definite width, phase and time and these types of signals could be considered driven by stabilized frequencies of the neuron. However, recently it has been discovered that neurons communicate by firing short electrical pulses which operate in a mode referred to as rate coding. Higher output signal is brought about by firing at higher rate. Communication is carried out using
real spikes only at the two exclusive instants known as spiking time or no spiking time. With a communication window of this type, the value of output of a neuron can be computed and the response of the network to the values of input is known or identified only after all the neurons have fired. Each neuron can be modeled since it has a basic firing-rate and a continuously constant activation function. This model which is referred to as Spiking Neural Networks (SNN) is considered to constitute the third generation of ANNs (Gründing and Sporea, 2011).

For all the three generations of neural networks, the output signals can be continuously altered by variation in synaptic weights (synaptic plasticity). Synaptic plasticity is the basis for learning in all ANN. As long as there is non-variant activation function, accurate classification based on a certain vector input values can be implemented with the help of a BP learning algorithm like gradient-descent (Y.-J. Jin, et al., 2013; Kempter and Hemmen, 1999).

Spiking Neural Network (SNN) utilises individual spikes in time domain to communicate and to perform computation in a manner like what the real neurons actually do (Belatreche and Paul, 2012; Ferster and Spruston, 1995). This method of sending and receiving individual pulses is called pulse coding where information which is transmitted is carried by the pulse rate. Hence, this type of coding permits multiplexing of data (Gerstner et al., 1999).

For instance, analysis of visual input in humans requires less than 100ms for facial recognition. Yet, facial recognition was performed by Thorpe & Delorme (S. Thorpe et al., 2001) by using SNN with a minimum of 10 synaptic steps on the retina at the temporal lobe, allowing nearly 10ms for the neurons to process. Processing time is short but it is sufficient to permit an averaging procedure which is required by pulse coding (Gerstner, et al., 1999; Kasabov, 2012b; S. Thorpe, et al., 2001). In fact, pulse coding technique is preferred when speed of computation is the issue (S. Thorpe, et al., 2001).
1.2.1 Spiking Neural Networks (SNNs)

Neural networks which perform artificial information processing are built using processing units composed of linear or non-linear processing elements. A sigmoid function is widely used (Bishop, 2000; Haykin, 1998; Kasabov, 2012a). SNN had remained unexplored for many years because it was considered too complex and too difficult to analyze. Apart from that:

1) Biological cortical neurons have long time constants. Inhibition speed can be of the order of several milliseconds while excitation speed can reach several hundreds of milliseconds. This dynamics can considerably constrain applications that need fine temporal processing (Gewaltig, 2000; Kasabov, 2009).

2) Little is known about how information is encoded in time for SNNs. Although it is known that neurons receive and emit spikes, whether neurons encode information using spike rate or precise spike time is still unclear (Thorpe and Gaistrais, 1998). For those supporting the theory of spike rate coding, it is reasonable to approximate the average number of spikes in a neuron with continuous values and consequently process them with traditional processing units (sigmoid, for instance). Therefore, it is not necessary to perform simulations with spikes, as the computation with continuous values is simpler to implement and evaluate (Kasabov, 2010).

An important landmark study by Maass (Maass, 2001) has shown that SNN can be used as universal approximations of continuous functions. Maass proposed a three-layer SNN (consisting of the input layer, the generalization layer and the selection layer) to perform unsupervised pattern analysis. Mishra (Mishra et al., 2006) applied spiking neural networks to several benchmark datasets (which include internet traffic data, EEG data, XOR problems, 3-bit parity problems, iris dataset) and performed function approximation and supervised pattern recognition (Gruning and Sporea, 2011).
One of the ongoing issues in SNN research is how the networks can be trained. Much research has been done on biologically inspired local learning rules (Gerstner and Kistler, 2002a, 2002b; Kasabov, 2010), but these rules can only carry out supervised learning for which the networks cannot be trained to perform a given task. Classical neural network research became famous because of the error-backpropagation learning rule. Due to this, a neural network can be trained to solve a problem which is specified by a representative set of examples. Spiking neural networks use a learning rule called SpikeProp which operates on networks of spiking neurons and use exact spike time temporal coding (Bohte et al., 2002). This means that the exact spike time of input and output spikes encode the input and output values.

1.2.2 Learning in Spike Neural Networks (SpikeProp)

Supervised learning (SpikeProp) in Spiking Neural Networks (SNNs) is usually performed by a gradient descent method which explicitly evaluates the gradient of an error function on the back-propagation algorithm (ˇS´ima, 2009; Bohte, et al., 2002; Grˇuning and Sporea, 2011). Using this algorithm, SNN learns the desired firing times of the output neurons by adapting the weight parameters in the Spike Response Model (Gerstner and Kistler, 2002b). Several experiments have been carried out on SpikeProp to clarify several burning issues such as the role of the parameters for initialization and the significance of negative weights (Moore, 2002). SpikeProp can be further enhanced with additional learning rules for synaptic delays, thresholds, and time constants (Schrauwen and Campenhout, 2007) which will normally result in faster convergence and smaller network sizes for given learning tasks. An essential speedup was achieved by approximating the firing time function using the logistic sigmoid (Berkovec, 2005). Implementation of SpikeProp algorithm on recurrent network architectures has shown promising results (Tiˇno and Mills, 2005).
SpikeProp does not usually allow more than one spike per neuron which makes it suitable only for ‘time-to-first-spike’ coding scheme (S. Thorpe, et al., 2001). Its adaptation mechanism fails for the weights of neurons that do not emit spikes. These difficulties are due to the fact that spike creation or its removal due to weight updates is very discontinuous. ASNA-Prop has been proposed (Schrauwen and Campenhout., 2007) to solve this problem by emulating the feed forward networks of spiking neurons with the discrete-time analog sigmoid networks with local feedback, which is then used for deriving the gradient learning rule (Grüning and Sporea, 2011). It is possible to estimate the gradient by measuring the fluctuations in the error function in response to the dynamic neuron parameter perturbation (Fiete and Seung, 2006). Table 1 summarizes the differences among the EBP, SNN and SpikeProp algorithms.

**Table 1.1 Summary of Differences of EBP, SNN and SpikeProp**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EBP</strong></td>
<td>They are second generation ANNs which do not use step- or threshold functions to compute their output signals. Their activation functions are continuous, making them suitable for analog in- and output operations. They are more powerful than their first generation predecessors. Software simulations associated with them are easy to implement (Y.-J. Jin, et al., 2013; Zweiri et al., 2003).</td>
</tr>
<tr>
<td><strong>SNN</strong></td>
<td>They are third generation ANNs. Functionally, they operate like real biological systems because the neurons facilitate the use of individual spikes. In the past, SNNs were considered too complex and difficult to analyze. Recently, many researchers have become aware of their potential to operate in a manner more powerful than the first generation and the second generation ANNs. (Kasabov, 2010; Negnevitsky, 2002).</td>
</tr>
<tr>
<td><strong>Spikeprop</strong></td>
<td>The firing time of SNNs can be shifted by adapting the incoming synaptic weights, the threshold and the membrane time constant. Bohte et al. used this idea for developing an error-backpropagation learning method based on the exact timing of spikes called SpikeProp, (Bohte, et al., 2002; Grüning and Sporea, 2011).</td>
</tr>
</tbody>
</table>
1.3 Problem statement

Performance of SNNs is dictated by its architecture algorithm. It has been important for this research work to develop a learning algorithm for the SNN so that it is able to classify data. Biologically inspired SNN is normally capable of implementing supervised learning (ˇS´ima, 2009). However, supervised learning rule is implementable if it operates in conjunction with backpropagation (ˇS´ima, 2009). This learning rule is called SpikeProp which utilizes spike time temporal coding while the backpropagation is treated as a network of spiking neurons. In this case, the input and output variables perform encoding according to the correct spike time of both the output and input spikes.

For the supervised learning rule of SpikeProp, the learning rate, the momentum, the bias, the minimum error for the transfer, the activation function and the initial weights are similar to that of other ANNs. The performance of SpikeProp is determined by the learning parameters of the BP network. Hence, it is necessary for this study to focus on finding ways to enhance the performance of SpikeProp by optimizing the backpropagation initializing weights and the architectures of SNNs. Particle Swarm Optimization (PSO) is one technique which can be used for this purpose.

In the course of this thesis, the aim is to highlight important issues and provide solutions to a number of general and specific research questions concerning SNNs.

How will it be possible to enhance SpikeProp to enable them to efficiently process data?
This question is answered in Chapters 4 and 5, where this thesis has, among other things, suggested that accuracy (low error) and the use of less number of iterations (less time to optimize) can be achieved by using:

1. Multi decision techniques to get to the optimum (Model 1).
2. Use of Angle driven dependency Learning Rate (Model 2).
3. Radial segmentation of the weights space to aim directly on to the optimum position (Model 3a).
4. Use of chromosomes (agents) to identify optimum positions quickly (Model 3b).

What will be the optimum back propagation initializing weights and what will be the best architectures of SNNs for classification tasks?

This question is also answered in Chapters 4 and 5. This thesis stipulated that the best architecture of SNN for classification tasks is that which can guarantee so that SNN has components which get the initial weights as close as possible to the optimum positions and also has components which can search for the optimum quickly (Model 4 and Model 5).

1.4 Objectives of the Research

The goal of this study is to propose an optimum back propagation configuration for spiking neural network (using SpikeProp as the supervised learning rule) according to the following considerations:

1. Adopt new initial weights methods.
2. Propose appropriate momentum factors which are needed to speed up convergence and estimate corresponding adaptive learning average or rate.

3. Suggest methods, for example, by using PSO rules to enhance architectures of SpikeProp.

The objectives of this research are focused on.

1. To enhancing Spikeprop’s architecture by:

   I  The implementation of PSO for learning optimization (Model 1).
   II  Proposing the parameters for $\mu$ Angle Driven Dependency learning rate in SpikeProp (Model 2)
   III Determining Radius Initial Weights (RIW) for SpikeProp to accelerate learning (Model 3a).
   IV  The development of DE for SpikeProp weights initialization (Model 3b).

2. To propose the hybridization for better SpikeProp performance by:

   I  PSO Spikeprop with learning rate $\mu$ Angle Driven Dependency (Model 4).
   II  PSO SpikeProp with RIW and DEA weights initialization (Model 5).

3. To evaluate and compare the proposed method with the conventional approaches SpikeProp and BackPropagation standard.
1.5 Scope of the Study

1. Using C++ program to develop several SNN Models, each having its own learning characteristic. It has been established that in order for SNN to perform classification accurately and quickly, two important issues have to be resolved. The first issue is how to get the initial weights as close to the optimum as possible. The second is how to get to the optimum with minimum error. For this reason many different novel BP SNN algorithms (SpikeProp) were tested. The development and testing of Models 1, 2, 3a, 3b, 4 and 5 are described in Chapters 4 and 5.

2. Data sets from breast cancer, Pima Indian diabetes, heart, BTX, hepatitis, liver and Iris were used for testing the different SNN models to ascertain and identify their classification properties. These datasets are known for their attributes, classes, samples, inputs and outputs (as mentioned in Table 5.1). For each of these datasets both Hold-Out Validation and K-Fold Cross Validation (training and testing procedure) were used to find-out which of the two has an advantage. XOR datasets is a test pattern to inspect the firing times of SNN. Therefore, XOR will bring about the worst case scenario in any model that is tested.

3. Matlab 8.0 was used to look at the behavior of the output generated by supervised BP (standard ANN BP). The behavior of the output generated by standard SpikeProp was studied using C++. The results from the Matlab 8.0 program (standard ANN BP) as well as the results from the C++ program (standard SpikeProp) were then compared to the models which have been developed.
1.6 Thesis Organization

Lately, SNN has captured the imagination of researchers and scientists because SNN operates in a manner similar to biological neurons which communicate with the help of electrical pulses. The advantage of SNN over standard ANN is the fact that in SNN there is no attenuation of the communicated signals, the energy used is small and SNN allows "refresh" in the system so that no residual signal remains which can interfere with the function of classification.

Chapter 2, Literature Review, give a review on error back propagation (EBP), Spiking Neural Networks and SpikeProp algorithm and the impact of error functions and learning parameters on BP and SpikeProp. The workings of PSO and DE are explained in some length because these two systems are the basis of important modification made to SNN architecture (SpikeProp algorithm) to improve performance (training and classification ANNs, SNN network design. SNN and SpikeProp are detailed in this chapter. Broad overview about the basic concepts and traditional techniques of Artificial Neural Network (ANN), Spiking Neural Network and SpikeProp are given. Furthermore, applicability of Spiking Neural Network in ANN learning and especially BP and SpikeProp Algorithm learning were discussed.

Chapter 3, Research Methodology, comprises of research methodology, describing the overall solving-tools and standard techniques adopted. It also displays a general picture about each phase of the work. This chapter discussed the methodology which was used in this research and describes new techniques and parameters that are required for BP learning and Spikeprop. It discusses thoroughly the requirements, framework and phases of this research. It consists of standard algorithms of SpikeProp algorithm learning.

Chapter 4, The Proposed Models For Spikeprop Learning Enhancements

This chapter proposes the design and implementation technique which is improved the performance of SpikeProp. It was observed that the errors (MSE, RMSE, MAPE and MAD) of SpikeProp can be reduced by using $\mu$ Angle driven Learning rate dependency (Model 2 and used the PSO as a means of improving
SpikeProp. In other words, combining SpikeProp and PSO (to get Model 1-PSOSpikeProp) should result in an algorithm which will have many positive values. Model 1 -PSOSpikeProp. Therefore this chapter also looks at a technique of obtaining good initial weights by using RIW (Model 3a) and by using DE (Model 3b). Just as we get good strain by cross-breeding two good genes, it may be possible to get good SpikeProp algorithm by hybridizing two good techniques. Therefore, this thesis looks at the hybridization of Model 1 and Model 2 (to get Model 4) and also the combination of Model 1, Model 3a and Model 3b (to get Model 5).

Chapter 5, Results and Discussion, correlates different techniques by comparing different results produced by the proposed Model 1, Model 2, Model 3a, Model 3b, Model 4 and Model for classification problems. Performance of the proposed methods is compared, analyzed and benchmarked with previous results. This thesis investigates ways in which the performance of SNN can be improved. Several SNN models were developed. For each model the architecture, the learning process and the testing procedure is uniquely different. This is discussed in detail in Chapters 4 and 5. SpikeProp which is an algorithm that mimics the function of SNN is a useful tool for the investigation of SNNs.

Chapter 6, Conclusion and Future Work, discusses and highlights the contributions and findings of the research work, and it provides suggestions and recommendations for future study.
REFERENCES


Nakazawa, Quirk, Chitwood, Watanabe, Yeckel, Sun, et al. (2002). Requirement for hippocampal CA3 NMDA receptors in associative memory recall. (Science 297, 211–218).
Neuron, G. a. (2007). New neuron models have been developed using the simulation tools of Genesis and Neuron.


