

Master Thesis
Computer Science
Thesis no: MCS-2008:22
June 2008



Measure-based Learning Algorithms

An Analysis of Back-propagated Neural Networks

Fahad Khalid

Department of
Interaction and System Design
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden

This thesis is submitted to the Department of Interaction and System Design, School of Engineering at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science (Intelligent Software Systems). The thesis is equivalent to 20 weeks of full time studies.

Contact Information:

Author:

Fahad Khalid

E-mail: fahad.khalid@gmail.com

University advisor:

Niklas Lavesson

Department of Systems and Software Engineering

Department of
Interaction and System Design
Blekinge Institute of Technology
Box 520
SE – 372 25 Ronneby
Sweden

Internet : www.bth.se/tek
Phone : +46 457 38 50 00
Fax : + 46 457 102 45

ABSTRACT

In this thesis we present a theoretical investigation of the feasibility of using a problem specific inductive bias for back-propagated neural networks. We argue that if a learning algorithm is biased towards optimizing a certain performance measure, it is plausible to assume that it will generate a higher performance score when evaluated using that particular measure. We use the term *measure function* for a multi-criteria evaluation function that can also be used as an inherent function in learning algorithms, in order to customize the bias of a learning algorithm for a specific problem. Hence, the term measure-based learning algorithms.

We discuss different characteristics of the most commonly used performance measures and establish similarities among them. The characteristics of individual measures and the established similarities are then correlated to the characteristics of the back-propagation algorithm, in order to explore the applicability of introducing a measure function to back-propagated neural networks.

Our study shows that there are certain characteristics of the error back-propagation mechanism and the inherent gradient search method that limit the set of measures that can be used for the measure function. Also, we highlight the significance of taking the representational bias of the neural network into account when developing methods for measure-based learning.

The overall analysis of the research shows that measure-based learning is a promising area of research with potential for further exploration. We suggest directions for future research that might help realize measure-based neural networks.

Keywords: Supervised learning, Inductive Bias, Artificial Neural Networks.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Niklas Lavesson for the excellent technical advice and enormous support extended throughout the course of this research. I am also grateful to him for introducing me to the exciting area of machine learning in general, and this very research topic in particular.

I would also like to thank Professor Paul Davidsson for the very interesting technical discussions we had. These discussions helped me identify the limitations of certain approaches to solving the problem, and identifying the appropriate method for achieving results.

The overall experience of studying at the Blekinge Institute of Technology has been wonderful and for that I am thankful to the faculty and staff at the institute.

Last but not the least; I am grateful to the people of Sweden in general and the people of Ronneby in particular for making my stay in Sweden the most delightful experience.

CONTENTS

ABSTRACT	1
ACKNOWLEDGEMENTS	2
1 INTRODUCTION	5
1.1 OUTLINE	5
2 BACKGROUND	7
2.1 A BRIEF INTRODUCTION TO MACHINE LEARNING	7
2.2 SUPERVISED CLASSIFICATION LEARNING	8
2.2.1 <i>Rule Induction Algorithms</i>	8
2.2.2 <i>Decision Trees</i>	8
2.2.3 <i>Linear Classifiers</i>	8
2.2.4 <i>Instance-based Learners</i>	9
2.2.5 <i>Statistical Learners</i>	9
2.3 EVALUATION.....	9
2.4 INDUCTIVE BIAS	10
2.5 CUSTOMIZATION OF THE INDUCTIVE BIAS	10
2.6 RELATED WORK	10
2.6.1 <i>Optimization of Accuracy</i>	11
2.6.2 <i>Optimization of other Measures</i>	11
2.6.3 <i>Optimization for Multiple Measures</i>	11
2.7 DIRECTION FOR THIS THESIS.....	12
3 PROBLEM DEFINITION	13
3.1 PROBLEM STATEMENT.....	13
3.2 AIM	13
3.3 OBJECTIVES	13
3.4 RESEARCH QUESTIONS	13
3.5 RESEARCH METHODOLOGY	14
3.6 CONTRIBUTIONS	14
4 THEORETICAL FOUNDATIONS – BACK-PROPAGATED NEURAL NETWORKS AND PERFORMANCE MEASURES	15
4.1 ARTIFICIAL NEURAL NETWORKS	15
4.1.1 <i>Historical Background</i>	15
4.1.2 <i>Biological Origin</i>	16
4.1.3 <i>Architecture / Structural Representation</i>	16
4.1.4 <i>The Backpropagation Algorithm for Learning</i>	17
4.2 MEASURES FOR PERFORMANCE EVALUATION	18
4.2.1 <i>Accuracy</i>	18
4.2.2 <i>Sensitivity and Specificity</i>	19
4.2.3 <i>Area under the ROC Curve</i>	19
4.2.4 <i>Mean-squared Error</i>	19
4.2.5 <i>Cross Entropy</i>	20
4.2.6 <i>Precision, Recall and F-score</i>	20
4.2.7 <i>Lift</i>	20
4.2.8 <i>Similarities in Measures</i>	21
4.2.9 <i>Qualitative Measures</i>	21
5 CHARACTERISTICS OF BACK-PROPAGATED NEURAL NETWORKS PERTAINING TO LEARNING	23
5.1 ISSUES REGARDING THE PROCEDURAL BIAS.....	23
5.1.1 <i>Limitations of the Gradient Descent Algorithm</i>	23
5.1.2 <i>Limitations of the Backpropagation Mechanism</i>	24
5.2 LIMITATIONS OF THE REPRESENTATIONAL BIAS.....	25
5.2.1 <i>Receptive Local Fields</i>	25

5.2.2	<i>Network Pruning</i>	25
5.2.3	<i>Constructive Networks</i>	25
6	DISCUSSION	27
6.1	OBJECTIVE FUNCTION REPLACEMENT.....	27
6.2	SIGNIFICANCE OF THE REPRESENTATIONAL BIAS	28
6.2.1	<i>The Bias/Variance Dilemma</i>	28
6.2.2	<i>Relationship between Representational and Procedural Bias</i>	28
6.2.3	<i>Conclusion</i>	29
6.3	USING GLOBAL SEARCH ALGORITHMS	29
7	CONCLUSIONS AND FUTURE WORK	30
7.1	FUTURE WORK.....	30
	BIBLIOGRAPHY	32

1 INTRODUCTION

A major motivation behind research in artificial intelligence has been to imitate human intelligence in computers [54]. Algorithms and methods from computer science are used to implement approximations of intelligence that would suit machine processing. One major trait of intelligence is learning, since the ability to learn makes it possible for humans to improve their performance in solving different tasks. This imitation in machines makes it possible to use computers for performing tasks of understanding and extracting meaningful information from data. This is one of the most popular application areas of machine learning, known as data mining [1].

The past decade has witnessed a “stunning progress in data mining and machine learning” [1]. A number of machine learning algorithms are now available for commercial use e.g. backpropagation based neural networks [2], support vector machines [3], C4.5 decision tree learners [4], random forests [5], meta-learning algorithms e.g. stacking [6] etc. It could be argued that most machine learning algorithms primarily comprise of a structural representation (e.g. tree, network, rules), a learner (e.g. C4.5, backpropagation, separate and conquer), and a performance element (e.g. accuracy, coverage, simplicity) which the learning algorithm optimizes.

Evaluation of machine learning algorithms is of primary importance in making real progress in data mining [1]. This is due to the fact that evaluation provides the means to compare the different algorithms, and to choose amongst them for use on a particular problem. The most common evaluation method is to estimate the predictive accuracy of the classifier under consideration. However, accuracy is not always the appropriate evaluation criterion [7] e.g. in situations where the misclassification costs among the classes are not equal. Additionally, it is argued that different evaluation measures are appropriate in different situations [8].

Most existing machine learning algorithms are implicitly biased to achieve better results against certain evaluation measures [9]. The inductive bias is partly inherited from the structural representation used; because measure function maximization is an optimization problem, and different optimization algorithms are dependent on different data structures. For example, back propagation based artificial neural networks have shown to be relatively efficient (in terms of accuracy) in comparison to decision trees [10].

A multi-criteria evaluation method known as the *measure function* [53] has been developed to be able to reveal performance aspects that cannot be found using the conventional accuracy evaluation method. Since the measure function can potentially provide a thorough insight into the inductive bias of a learning algorithm, it can be useful to develop a method that maximizes the measure function for a given learning algorithm; thereby increasing the classifier’s performance on the defined evaluation measure.

In this thesis we investigate the applicability of this concept to back-propagated neural networks. The idea is to exchange the inherent performance measure with a generic measure function (thereby altering their inductive bias); and evaluate the effectiveness of the resulting modified algorithm.

1.1 Outline

The chapters that follow are organized as follows. Chapter 2 builds the background for the research by introducing the fundamental concepts of machine learning that are specifically relevant to this thesis. An overview of related work is also presented in Chapter 2. In Chapter

3 we present a formal problem definition followed by a discussion on the research methodology used. Chapter 4 presents an introduction to the basics of artificial neural networks and performance evaluation measures. These concepts are then extended in Chapter 5 to highlight certain issues pertaining to the use of back-propagated neural networks as measure-based learning algorithms. We discuss the impact of these issues on the research in Chapter 6. The thesis concludes with Chapter 7, presenting conclusions and directions for future research.

2 BACKGROUND

This chapter starts by introducing the reader to the field of machine learning. We present an introduction to the concept of supervised classification learning, and introduce some commonly used supervised classification learning algorithms. Further, we present the concepts of classifier evaluation and inductive bias of machine learning algorithms. Finally, we correlate these concepts to establish the motivation for our research.

2.1 A Brief Introduction to Machine Learning

The field of machine learning is considered as a sub-field of artificial intelligence. It fosters methods and techniques that imitate learning in computers. The concept of learning in itself has generated philosophic discussions [11], and it is therefore hard to define learning exactly. In this thesis however, we avoid delving into any such philosophic discussions and use the following definition for our understanding [12]:

“Learning denotes changes in a system that ... enable a system to do the same task more efficiently the next time.”

For learning in computers, the above definition can be formalized as [13]:

“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks in T , as measured by P , improves with experience E .”

Learning can be either inductive or deductive. Here, we consider only the inductive machine learning methods that are applied to extract rules and patterns from large collections of data. The extraction of rules and patterns (acquisition of knowledge) helps explaining the data, and to make predictions for the future (use of the acquired knowledge).

The concept of inductive machine learning has been successfully applied in various fields [1,13,15,16]. With the growing number of credit card transactions, credit card companies face the threat of fraudulent transactions. Machine learning algorithms are used by these companies that help them identify if a transaction is legitimate or not [31]. This helps saving a lot of time and effort that would otherwise be spent on manually analyzing each and every transaction.

Identification of disease susceptibility genes in genetically transferred diseases like hypertension is a challenging area for researchers [30]. Sophisticated machine learning algorithms are employed for gene identification in these cases, since the traditional statistical methods are not adequate.

Ensuring customer loyalty is very important for telecom operators in order to sustain a steady stream of revenue. Machine learning algorithms are employed by such companies that help forecast which customers are likely to churn in the near future and which are likely to generate good revenue [55]. This helps the decision makers in launching marketing campaigns targeted towards a small set of customers, rather than adopting mass marketing.

Similarly, there are various other application areas where machine learning algorithms have been employed to help in complex decision making and/or increasing efficiency and productivity on certain tasks.

2.2 Supervised Classification Learning

A supervised classification learning algorithm can be described as an algorithm that learns to classify instances into different categories with the help of training samples [16]. The following subsections briefly introduce some of the commonly used supervised classification learning algorithms.

2.2.1 Rule Induction Algorithms

Rule induction algorithms [1] are used to generate rules for a given data set. The following expression is provided as an example of a simple rule:

$$\text{IF score} > 90 \text{ THEN grade} = \text{A} \quad (1)$$

The idea is to create rules that generalize the relationships between the various instances in the dataset. As can be observed from the example above, rules can be very simple for humans to interpret and understand.

The method used for rule extraction is called *covering* or *separate-and-conquer* [19]. A covering algorithm selects one class at a time, and tries to find all instances that belong to this class. The goal is to find rules that classify with maximum accuracy. Advanced methods based on this basic principle are used to build powerful classification algorithms like *PRISM* [19].

2.2.2 Decision Trees

Unlike rule learners, decision tree induction algorithms [1] are based on the principle of *divide and conquer*. The process begins by choosing an attribute to represent the root node of the tree. The algorithm works from top to bottom, splitting the most suitable attribute into branches at each stage; the most suitable attribute being the one that best separates the classes. The same process is applied recursively until the end.

In order to choose the most suitable attribute to split, the concepts of information gain and entropy are applied. These concepts originate primarily from the field of *information theory* [1]. The above mentioned basic method for decision tree induction is further refined to develop industry grade classification algorithms like *C4.5* [4].

2.2.3 Linear Classifiers

Linear models in classification learning have primarily been derived from the concept of *statistical linear regression* [1]. For datasets with numeric attributes, a linear combination of attributes and corresponding weights is used to express a class.

$$x = w_0 + w_1a_1 + w_2a_2 + \dots + w_na_n \quad (2)$$

The same model with a minor modification can be used for classification problems as well. The idea is to apply the above mentioned regression method for each class, but setting the output value to 1 if the instance under consideration belongs to the class and setting the output to 0 otherwise. The system of linear expressions can then be subjected to a test instance. The linear expression that produces that largest value denotes the classification.

Industry grade algorithms like *artificial neural networks* [2] and *support vector machines* [3] are examples of linear models that are used to solve hard real world problems.

2.2.4 Instance-based Learners

Instance-based learners are also known as *lazy learners* [1]. The idea is to store the data set as it is. When an unknown test instance is presented, a distance function is used to determine which instance from the training set is nearest to the test instance. The class of this nearest instance is predicted to be the class of the unknown test instance. These algorithms are known as *nearest neighbor* algorithms.

Advanced processing like the use of powerful and efficient distance functions, efficient data structures like *kd-trees* and *ball trees*, etc, leads to powerful industry grade algorithms and methods like the *k-nearest neighbor* algorithm [18] and *clustering*.

2.2.5 Statistical Learners

Statistical learners are based on *Bayes's* rule of conditional probability [1]. The rule is formally defined as:

$$\Pr[H | E] = \frac{\Pr[E | H]\Pr[H]}{\Pr[E]} \quad (3)$$

Where H is a hypothesis and E is the evidence that bears on that hypothesis. The idea here is to calculate the likelihood of an instance belonging to a particular class. The class with the highest probability is predicted to be the class for the instance. One advantage of this scheme is that the output of the classifier is a probability distribution rather than just class values. This quality can be specifically useful in certain application e.g. document classification.

The above mentioned basic principle forms the basis for a very powerful learning scheme known as *Bayesian networks* [15]. Probabilistic Bayesian networks are used in some of the world's most powerful search engines.

2.3 Evaluation

The previous section briefly described only some of the most commonly used methods in classification learning. Nevertheless, the above discussion raises an important question i.e. how do we know which method to use in a certain situation?

There are some obvious differences between the applicability of certain methods. For instance, in a situation where one of the high priority requirements for the classifier is to produce results in a human understandable format, rule learners would appear to be a better choice than multilayer perceptrons [13]. However, there are situations where the choice of which classifier to use is not that apparent. Therefore, we need a method to determine which classification algorithm to choose in a certain situation.

Evaluation methods for machine learning algorithms gain their significance from the above mentioned requirement. This is due to the fact that evaluation provides the means to compare the different algorithms, and to choose amongst them for use on a particular problem. Evaluation consists of a scientific method to statistically compare the performance of the classifiers in question on one or more performance measures [1,13]. We discuss the most widely used measures for classifier performance evaluation in Section 4.2.

2.4 Inductive Bias

At its very core, a learning algorithm searches through a space of several possible classifiers and tries to find the one that best suits the given problem. The idea is to optimize the search towards finding a classifier that would perform best on the given performance measure. The inductive bias of a learning algorithm can be defined as [22]:

“Any basis for choosing one classifier over another, other than strict consistency with the instances”

Therefore, it is the inductive bias of a learning algorithm that guides the search for the most suitable classifier. The inductive bias of a learning algorithm further consists of a *representational bias* and a *procedural bias* [22].

As discussed previously, different learning algorithms are based on different representations e.g. networks, trees, rule sets etc. The representational bias specifies this representation, and as such, defines the search space for candidate classifiers.

Once the search space has been defined, an algorithm is required that would search this space for a suitable classifier. This algorithm is specified by the procedural bias. For example, back-propagated neural networks often use the gradient descent method to search for the optimal classifier.

To summarize, it would be safe to say that inductive bias forms the heart of the learning algorithm.

2.5 Customization of the Inductive Bias

In the above sections, we have established the fact that a learning algorithm primarily uses the inductive bias to find a classifier that would perform best when evaluated against the given performance measure. However, in practice, most of the learning algorithms are inherently biased towards finding the classifier that maximizes accuracy [8]. Even though evaluating such classifiers against measures other than accuracy might still find a relatively better classifier, the search would least likely be optimal.

On the other hand, if the procedural bias of the learning algorithm is modified in such a way that it optimizes the search for a classifier that would maximize performance against the given performance measure; the search can be optimal. This concept was initially presented by Andersson et al [53].

In this thesis, we try to see, by means of theoretical investigation, whether the above stated hypothesis can be accepted. In order to limit the scope of the research, we focus our investigation on the applicability of the concept of measure-based learning to back-propagated neural networks.

2.6 Related Work

In this section we present a summary of the most prominent areas of research that are relevant to our study. We categorize the related work into three major categories based on the direction of research.

2.6.1 Optimization of Accuracy

The first category comprises of research that is directed towards improving accuracy of a particular learning scheme by introducing a modification to the original algorithm.

Holt and Semnani [23] show that the convergence of a back-propagated neural network can be improved by replacing the inherent error measure with a log-likelihood measure. However, the accuracy measure is still used for evaluation. Thus, the objective of the study is to improve the accuracy of back-propagated neural networks by introducing a new inherent measure.

In another study, Fujiki et al [24] demonstrate by means of empirical analysis that the use of Kullback-Leibler divergence measure in place of the conventional quadratic error measure improves the accuracy of a back-propagated neural network.

Similar other studies e.g. [25], demonstrate methods to further improve the accuracy of a learning algorithm.

2.6.2 Optimization of other Measures

The second category includes works focused on modifying different learning algorithms to optimize measures other than accuracy by replacing their inherent measures with the measures to be optimized.

Kukar et al [20] present a modification to the backpropagation algorithm used for training neural networks. They introduce a cost factor to the delta rule that incorporates the misclassification cost for each class. Results from their empirical study show that their measure can be very useful in situations where it is desired to bias the learning algorithms towards maximizing either sensitivity or specificity. However, an apparent limitation of the study is the assumption that misclassification costs for all classes must be known at the time of learning.

Ferri et al [26] present a method for decision tree induction that biases the inducer towards maximizing the area under the ROC curve measure. They propose a node splitting criterion that uses the value of the local area under the ROC curve to decide on splitting. This is particularly useful in situations where the costs of misclassification are not known at the time of tree induction. Results from their study indicate that decision trees induced using this approach tend to produce a better overall score for the area under the ROC curve measure.

2.6.3 Optimization for Multiple Measures

The third category focuses on the modification of a particular learning algorithm in order to enable the algorithm to optimize more than just one measure using an algorithm specific method.

Joachims [27] presents an algorithm based on support vector machines that optimizes multivariate non-linear performance measures. The study shows that this algorithm is applicable for optimization of various commonly used measures such as F1-score, Precision/Recall Breakeven Point, and area under the ROC curve.

Suzuki et al [28] present a dynamic bias selection method for prediction rule discovery. The algorithm uses multiple measures to identify the best suitable rule. However, the measures

used e.g. *predictiveness* are not widely used and are applicable only to predictive rule discovery.

2.7 Direction for this Thesis

In many real world classification problems, a learning algorithm may be required to optimize over more than one measure. This problem is to some extent solved by [27]; however, the work is specific to support vector machines, and considers only a specific class of quantitative evaluation measures.

In this thesis, we aim to investigate the applicability of a truly generic multi-criteria measure to back-propagated neural networks. Theoretical formulation for the generic measure has already been demonstrated in earlier research [29]. Here, we try to see whether this measure could actually be applied to back-propagated neural networks; and to identify the various issues that may arise in attempting to do so.

3 PROBLEM DEFINITION

In this chapter, we formally define the problem and describe the methodology used for the research.

3.1 Problem Statement

This thesis investigates the applicability of the concept of measure-based learning as applied to backpropagated artificial neural networks (BPNN).

3.2 Aim

The overall aim of this thesis is to investigate whether it is possible to modify the conventional backpropagation algorithm for neural networks to incorporate measure-based learning; and identify any issues that may arise in an attempt to do so.

3.3 Objectives

In order to achieve the aforementioned aim, the following objectives have been laid down:

- Investigate the possibility of applying the measure-based learning approach to back-propagated neural networks
- Identify the potential issues concerning measure-based learning in back-propagated neural networks
- Compare measure-based back-propagated neural networks with conventional back-propagated neural networks with respect to various performance measures
- Examine and state the primary properties of measure functions that would be suitable for use with a measure-based back-propagated neural network

3.4 Research Questions

We formulate the above mentioned objectives using the following research questions:

1. What could be the advantages and potential drawbacks of using a modified BPNN such that it uses a generic measure function instead of the implicit error rate minimization function?
2. If a traditional BPNN C_1 is compared to a modified BPNN C_2 that uses a measure function M for a particular problem, would the evaluation results be significantly different when both C_1 and C_2 are evaluated with respect to M ?
3. If a traditional BPNN C_1 is compared to a modified BPNN C_2 that uses a measure function M for a particular problem, would the evaluation results be significantly different when C_1 and C_2 are evaluated using the commonly used evaluation measures like accuracy, AUC etc?
4. What are the characteristics (if any) of a measure function that make it suitable enough for use with a generic measure function based BPNN, such that the classifier performs significantly different than a traditional BPNN?

3.5 Research Methodology

The investigative approach employed to answer the above mentioned research questions is primarily qualitative. The qualitative study is based on a broad and in-depth survey and analysis of relevant literature.

We started our research with the objective of developing a prototype BPNN employing a generic measure. Following this method, not only did we use the Weka machine learning workbench [1], the Stuttgart Neural Network Simulator¹ and the Java Object Oriented Neural Engine²; we also developed a simple yet flexible BPNN from scratch. Due to certain limitations of BPNN that bar the possibility of generalizable empirical analysis, we proceeded further with a theoretical approach. Our theoretical research uses mathematical foundations of BPNN as well different commonly used performance measures to approach the problem from different angles, in order to derive generalizable conclusions.

RQ 2 and RQ 3 are answered by analyzing some related empirical studies. RQ 1 and RQ 4 are answered by investigating and correlating the different aspects of the theory of BPNN as well as the theoretical foundations for the commonly used performance evaluation measures.

3.6 Contributions

This thesis constitutes a pioneering effort in the area of measure-based learning algorithms by investigating the applicability of the concept to BPNN. Since it is pure research, we do not envision a direct application of the results. However, the results of the research pave way for further research into the area by providing solid theoretical foundations and identifying directions that are likely to generate fruitful results.

The positive results of the investigation encourage further exploration of the concept, and identify the direction that might lead to a possible means of applying the concept to artificial neural networks.

The pitfalls and issues that we identify encourage investigations into alternate methods and algorithms for application of the concept.

¹ “Stuttgart Neural Network Simulator”. Home page URL: <http://www.ra.cs.uni-tuebingen.de/SNNS/>. Last visited: May 25, 2008.

² “Java Object Oriented Neural Engine”. Home page URL: <http://www.jooneworld.com/>. Last Visited: May 25, 2008.

4 THEORETICAL FOUNDATIONS – BACK-PROPAGATED NEURAL NETWORKS AND PERFORMANCE MEASURES

The purpose of this chapter is to introduce the reader to the fundamental concepts of back-propagated neural networks (BPNN) and some commonly used performance evaluation measures for supervised classification learning algorithms.

4.1 Artificial Neural Networks

The human brain has certain distinct characteristics that are not present in the computers that we use. Some of these characteristics are [32]:

- Massively parallel nature of processing elements
- Computational and representational distribution
- Ability to learn
- Ability to generalize
- Adaptivity
- Contextual information processing
- Fault tolerance

Due to the specific combination of the above characteristics, the human brain can simply outperform the modern day computers in certain tasks e.g. face recognition, speech recognition etc. Computers on the other hand are much faster and efficient than the human brain on certain numerical computations.

The primary difference between the human brain and digital computers is architectural. One of the fundamental differences is the massively parallel architecture of the human brain [33]. Biologically inspired neural networks, also known as artificial neural networks that are designed on the principle of parallel distributed processing have shown to be better at tasks where computers are generally known to perform poorly. Studying artificial neural networks has also helped gain better insight into the working of the human brain.

The study of artificial neural networks involves knowledge and expertise from various areas of science e.g. neurophysiology, cognitive psychology, control theory, computer science and artificial intelligence, mathematics and statistics, and so forth [34]. Developments in each of these areas contribute to the other and vice versa.

4.1.1 Historical Background

The history of the development of artificial neural network theory comprises of ups and downs in the interest in this field. The research area started out with the pioneering work on *perceptron* architecture and learning theory by McCulloch and Pitts [35]. This was followed by Rosenblatt's *perceptron convergence theorem* [36]. However, shortly after that Minsky and Papert [37] published their report on the limitations of the simple perceptron. This report had a demoralizing effect on both scientists in the area, as well as sources of funding. Thus, for almost twenty years, there was rarely any significant development in the field.

The major renewal of interest in the field can be contributed to Werbos's [38] work on *multilayer perceptron* that overcame the limitations of the simple perceptron. It was later

proved that artificial neural networks with certain configurations have the capability to represent a function of arbitrary complexity [32]. In recent years, the use of artificial neural networks has been observed in numerous fields.

4.1.2 Biological Origin

The primary component of the human nervous system is a cell called the *neuron* [34]. The basic use of this cell is information processing. A neuron is composed of a body, a tail like extending structure called the *axon*, and very fine branch like structures called *dendrites*.

The dendrites of a neuron work as receivers for impulses from other neurons. The axon is used to transmit these impulses. Two neurons are connected using very fine strands called *synapses*. Depending on the type of the synapse, an impulse may excite or inhibit the receiving neuron. The excitatory or inhibitory characteristics of a synapse may change with experience and learning. This is what fundamentally constitutes human memory.

A single neuron in the human brain may be connected to approximately 10^3 other neurons. The connectionist model of the human brain makes use of this massively parallel interconnected system of neurons to make complex decisions in a very short time.

4.1.3 Architecture / Structural Representation

In general, artificial neural networks can be viewed as weighted directed graphs. However, for the scope of this thesis we only study feed-forward networks, which can specifically be represented as acyclic weighted directed graphs. The particular type of feed-forward networks that we focus on is the *multilayer perceptron*.

A single artificial neuron is represented by a node in the graph. In a multilayer perceptron, these neurons are arranged in layers. Neurons in one layer are connected to the neurons in the preceding layer by weighted directed edges called *synapses*.

The layers in a multilayer perceptron are of three basic types. The *input layer* receives input from the environment. Typically, the number of neurons the input layer is kept equal to the number of attributes in the training instance. After the input layer are one or more *hidden layers* that act as feature detectors in pattern recognition tasks [39]. There is no specific formula for calculating the number of hidden layers required, or the number of neurons required per hidden layer. However, it has been theoretically proved that as many as two hidden layers are enough to solve tasks of any complexity [32]. The final layer is the *output layer*. For classification, the number of neurons is typically equal to the number of possible class values. For Boolean classification however a single output neuron might be enough, since an *on* state can represent one class and the *off* state can represent the other. The output neurons produce as output the classification estimate.

In order for the network to work as a classifier, it has to go through the process of learning i.e. it has to learn how to classify a certain input. This requires an appropriate learning algorithm. We discuss one of the most commonly used learning algorithms for multilayer perceptrons in the next section.

4.1.4 The Backpropagation Algorithm for Learning

The backpropagation algorithm [2] is a commonly used training algorithm for neural networks. It is also the most commonly used algorithm for supervised learning in multilayer perceptrons. As the name implies, it works by propagating error values back into the network.

The learning phase requires a training set that contains sample instances for which the classification is already known. The network is fed one instance at a time which it processes. The computation performed on each instance can be divided into two passes.

The forward pass consists of the process of receiving input at the input nodes, and taking it to the output nodes. The journey of input values to the output nodes involves transformations at each layer.

In the simplest form of the backpropagation algorithm, the input nodes do not apply any output transformations and forward the input values as they are received. Each neuron in the hidden layer receives as input its *induced local field* [34]. This is defined as the linear sum of all input values and their synaptic weights. The output of the hidden neuron is calculated by the associated *activation function*. Typically, the same activation function is used for all hidden neurons. The most commonly used activation function is the sigmoid activation function. It is defined as [13]:

$$f(x) = \frac{1}{1 + e^{-x}} \quad (4)$$

The three most important properties of this function are, 1) the output of the sigmoid function is a non-linear function of its input; 2) its asymptotic nature, and 3) that it is differentiable. The non-linearity is essential so that the network is capable of representing highly non-linear functions. The asymptotic nature of the function limits the output values between 0 and 1. The differentiable nature of the function is important to the *gradient descent* search procedure which is at the very heart of the backpropagation algorithm. We will discuss the gradient descent procedure in the next subsection.

The output of the last hidden layer is fed to the output neurons in the same way as inputs are fed to the first hidden layer. The output layer generates the output depending on which activation function is being used. This can be different from the activation function used for hidden neurons, depending on the application requirements.

The output generated by the network is compared against the target output already available from the training set, and the difference between the two outputs is calculated for each output node. This difference is then fed back into the network in the reverse order i.e. the error enters through the output nodes and travels backwards. The error difference is used to update the synaptic weights.

4.1.4.1 Backpropagation and Gradient Descent

Gradient descent is a widely known optimization algorithm for non-linear unconstrained optimization problems [40]. The error backpropagation algorithm uses this method in order to search for the most suitable classifier [41]. The criterion for suitability is based on the concept of reducing the difference between the output produced by the network, and the expected output.

The gradient descent algorithm is responsible for optimizing the network weights in such a way that the mean squared error between the network output and the target output is minimized. The search procedure can be visualized as going through an error surface, trying to find the point with the minimum possible error.

4.1.4.2 Inductive Bias in Back-propagated Neural Networks

The procedural bias of a BPNN is defined by the objective function that the gradient descent algorithm tries to optimize. As discussed in the previous section, the basic objective function is based on error rate minimization. Therefore, it can be said that the BPNN are primarily biased towards error rate minimization.

The representational bias of a BPNN is dependent on multiple factors pertaining to the structure of the network. Considering the case where the number of input and output nodes is determined by the earlier mentioned procedure; the number of hidden layers, and number of neurons per hidden layer has a major influence on the representational bias. Moreover, the structural configuration of the synaptic connections among neurons also affects the classification performance, and can hence be considered as a factor contributing to the representational bias.

4.2 Measures for Performance Evaluation

While establishing the background for this thesis, we mentioned that we would like to look into the possibility of biasing the BPNN to performing well on any given performance measure. In order for that, we first analyze the basic properties of some of the most widely used performance measures and try to develop a correlation among these measures.

The following subsections present some classifier performance evaluation measures that have been repeatedly used in practical machine learning applications, and try to establish a correlation between them.

4.2.1 Accuracy

Classification accuracy is probably the most widely used performance measure amongst the machine learning community [19]. The idea is to determine the success rate of the classifier in classifying unknown instances. More formally, the classification accuracy can be defined as:

$$\frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

Where TP is the number of instances correctly classified as positive; TN is the number of instances correctly classified as negative; FP is the number of instances incorrectly classified as positive; and FN is the number of instances incorrectly classified as negative. To summarize, the above formula gives a ratio of correctly classified instances over the total number of instances classified.

Even though accuracy appears to be intuitively appropriate for classifier performance comparison, real life datasets pose challenges that measurements of accuracy cannot overcome. For example, accuracy assumes equal misclassification costs for all classes. For certain classification problems, this assumption does not hold, and therefore, we need performance measures that consider costs of misclassification for each class.

4.2.2 Sensitivity and Specificity

One of the application areas of machine learning is medical diagnostics [20]. From the results of diagnostics tests, it is important for medics to determine the percentages of people who suffer from a certain disease, and those who do not. Sensitivity and specificity are two measures that provide this information. These can be formally defined as:

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (6)$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}} \quad (7)$$

i.e. sensitivity provides the proportion of people with disease who have a positive test result, and specificity provides the proportion of people without disease who have a negative test result. In certain cases, it might be critical for a classifier to correctly classify an ill patient as ill, but incorrectly classifying a healthy person as ill might be bearable. Then, a classifier with a higher score on sensitivity might be a better choice than a classifier with higher accuracy but lower sensitivity.

4.2.3 Area under the ROC Curve

Receiver operating characteristics (ROC) graphs were originally used in signal detection to characterize the tradeoff between hit rate (sensitivity) and false alarm rate (1 - specificity) over a noise channel [21].

A ROC graph plots sensitivity vs. specificity for all possible thresholds. The number of positives included in the sample is plotted on the vertical axis, against the number of negatives included in the sample on the horizontal axis. Both axes are represented as percentages.

The area under the ROC curve (AUC) metric is used as a summary statistic for ROC analysis [1]. The larger the area, the better the model would be. ROC analysis is used in visualizing the behavior of diagnostic systems, and for visualization in medicine.

4.2.4 Mean-squared Error

Mean-squared error (MSE) is one of the most commonly used performance measure for regression problems, and is also frequently used in classification. It is defined as [19]:

$$\text{MSE} = \frac{1}{N} \sum_{n=1}^N (O_n - T_n)^2 \quad (8)$$

Where N is total number of instances in the training set, O_n is the output produced by the classifier against the n^{th} instance, and T_n is the target output for that instance. Therefore, MSE can be regarded as a measure of the extent to which the classifier output differs from the target/expected output.

MSE has more significance to BPNN than just being a performance measure. The standard BPNN uses MSE as the objective for its gradient search algorithm [34]. That is, the gradient search seeks to minimize the MSE of the BPNN.

4.2.5 Cross Entropy

In certain situations, the classifier is required to produce likelihood estimations rather than actual values i.e. each output value corresponds to the probability that the input instance belongs to a certain class. Cross Entropy (CE) provides a measure of the maximum likelihood. CE is defined as [19]:

$$CE = -\frac{1}{N} \sum_{n=1}^N \{T_n \log(O_n) - (1 - T_n) \log(1 - O_n)\} \quad (9)$$

As can be seen from the mathematical definition, the basic principle behind CE is the same as MSE i.e. minimizing the error. However, in case of CE, the error is a term in the probabilistic setting.

4.2.6 Precision, Recall and F-score

These measures are commonly used in information retrieval applications [1] e.g. web search engines. Following are the definitions for precision and recall:

$$\text{Precision} = \frac{\text{Number of relevant documents retrieved}}{\text{Total number of documents retrieved}} \quad (10)$$

$$\text{Recall} = \frac{\text{Number of relevant documents retrieved}}{\text{Total number of relevant documents}} \quad (11)$$

F-score is a summary statistic for precision and recall. It gives the harmonic mean of precision and recall at a certain operating point. It can be defined as:

$$F - \text{score} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}} \quad (12)$$

4.2.7 Lift

Lift is a widely used measure in data mining for market analysis [1]. As an example, lift can be used to determine how good a classifier is at identifying a set of customers that is more likely to respond to a certain marketing campaign, out of a large customer database. The idea is to compare the classifier's performance with a random selection algorithm. Lift can be defined as [19]:

$$\text{LIFT} = \frac{\% \text{ of true positives above the threshold}}{\% \text{ of dataset above the threshold}} \quad (13)$$

The threshold value is set as per application requirement.

4.2.8 Similarities in Measures

In this section, we try to develop a correlation between all the above mentioned performance evaluation measures. Let us first divide them into two groups; 1) numeric error measures i.e. MSE and CE, and 2) measures based on TP, TN, FP and FN i.e. sensitivity, specificity, precision, recall, lift, AUC and F-score.

Let us take a scenario where we have several classifiers out of which we have to choose the best classifier against each performance measure. The classifiers that score best on MSE and CE would be the ones that have the least scores on these measures. If we look at equations (8) and (9), it is clear that theoretically, the least possible score on both these measures is 0. This is possible only when the classifier's output contains no errors at all on the given test set, against which the score is calculated.

Let us now consider the other measures. Sensitivity is maximized when there are no false positives, and specificity is maximized when there are no false negatives. AUC is maximized when both sensitivity and specificity are at their maximum values i.e. 1. A similar correlation can be derived for precision, recall and F-score.

It is not possible to achieve the above mentioned perfect results on most of the real life datasets because of the uncertainty associated with prediction, and the huge number of possible unseen instances. However, let us assume a purely theoretical situation where we can achieve such a perfect result. In that case, there exists a classifier that would achieve a maximum score at all the above mentioned performance measures. That would be the perfect classifier that never makes any mistakes, and always correctly classifies every instance.

We have introduced the most commonly used performance measure known as classification accuracy. For our perfect classifier, FP and FN would both be zero, which would give us 100% classification accuracy.

This provides us with a very simple correlation between all the above mentioned performance measures. For every classifier, the global (ideal) optimum is to achieve a state where it does not make any mistakes i.e. achieve 100% classification accuracy. In practice however, this is not achievable. Therefore, each measure represents a certain local optimum that bears certain characteristics of the global optimum. For applications that are measured against a certain measure, the characteristics represented by this local optimum are more suitable for this application than the other local optima.

The above is a rather simplified correlation in terms of global similarities with respect to classifier search by learning algorithms. There have also been studies that try to establish these correlations in terms of characteristics using the concept of isometrics [42, 43] as well as statistics [44]. For example Fürnkranz et al [43] prove that there are equivalences among some of the measures used for the evaluation of separate-and-conquer rule learning algorithms. They use isometrics to visualize the search heuristics in a 2-dimensional space. Similarly, Flach [42] discusses different measures in terms of 3-dimensional isometrics in order to develop a better understanding of these measures. Even though these studies provide some insight into the measures and their similarities, it is not possible to correlate them in such a way that would somehow help in defining generic measure functions.

4.2.9 Qualitative Measures

The performance measures discussed above can be categorized as quantitative measures. We would like to mention here that even though our research focuses primarily on taking the quantitative measures into account, there have been studies that have highlighted the

significance of qualitative measures. Examples of such measures are *comprehensibility* and *interestingness* [45]. One of the motivations behind the introduction of these measures is to enable evaluation of the discovered knowledge in terms of usefulness for human decision making. However, the evaluation of such measures is not a trivial task. The very fact that these measures are qualitative makes it necessary that these be converted into equivalent numerical representations, so that evaluation algorithms can be used. In this thesis, however, we focus on the quantitative measures and investigate BPNN only in this respect.

5 CHARACTERISTICS OF BACK-PROPAGATED NEURAL NETWORKS PERTAINING TO LEARNING

Now that we have established the fundamentals of the BPNN and the most commonly used performance measures, we dedicate this chapter to looking at certain properties of the BPNN that raise issues regarding their use as measure-based learning algorithms.

As discussed previously, an important property of a measure-based learning algorithm would be to be able to incorporate a generic measure as the objective function. The measure may or may not be multivariate. In the following sections, we try to see how certain properties of BPNN may prove as limitations to this concept.

5.1 Issues Regarding the Procedural Bias

The procedural bias of a BPNN is defined by the backpropagation mechanism as well as the gradient descent method used for objective function optimization. In this section, we explore some issues regarding these two primary elements of the procedural bias.

5.1.1 Limitations of the Gradient Descent Algorithm

We have mentioned in the previous chapter that the procedural bias of BPNN is defined primarily by the objective function used for the gradient descent algorithm. Therefore, the following subsections discuss certain issues concerning the gradient descent optimization procedure that must be considered with respect to measure-based learning.

5.1.1.1 Differentiable Objective Function

One of the fundamental properties of the gradient descent algorithm is that the objective function must be differentiable [40]. Even though MSE minimization is differentiable, and widely applicable, not all learning objectives are differentiable.

One example is the objective function for AUC maximization. The estimation of the AUC objective function can be formally defined in terms of the Mann-Whitney statistic as [46]:

$$\text{AUC}(\beta) = \frac{1}{PQ} \sum_{j=1}^P \sum_{k=1}^Q g(\beta \cdot (x_j^+ - x_k^-)) \quad (14)$$

In this very form, the objective function is not differentiable, and therefore it is not possible to use this function as the measure for BPNN. However, for certain functions, differentiable approximations are possible. For example, it has been shown that AUC can be defined as a differentiable approximation that can be optimized by using the gradient descent method. Nevertheless, functions might exist for which differentiable approximations have not been found.

5.1.1.2 Gradient Descent as Local Search

An inherent problem with the gradient descent method is that it is a local search algorithm [47]. This raises the issue of the inability of the algorithm to efficiently find the global

optimum. The algorithm is known to fall into local optima, and in those cases it is very hard for the algorithm to recover.

There have been several efforts to make it possible for the gradient descent algorithm to perform well on global search problems [41]. However, this is still an open area of research, especially in the case of application of gradient descent to neural network optimization.

Certain objectives may require a global search, and the above mentioned limitation of the gradient descent algorithm may pose a problem.

5.1.1.3 Multi-objective Optimization

Simplicity and specificity are examples of two objective functions that are non-commensurable i.e. it might not be possible to maximize both sensitivity and specificity at the same time. Therefore, an algorithm that attempts to do so must define a multi-objective function.

In such a case of multi-objective optimization, the objective is to achieve an optimal point where values for both the objectives are as maximized as possible i.e. the objective becomes a vector. A common practice is to combine the two objective functions into a single function so that we have a target scalar objective function instead. This allows the use of traditional methods for optimization of scalar objective functions.

However, aggregation of multiple objective functions into one function has its disadvantages. How to aggregate the multiple functions is the first question. There can be numerous possible ways of aggregating two objective functions, and it is very hard to determine which method would produce the optimal results. Also, often a priori information about the relative significance of the two objective functions is required; and even having that does not guarantee optimal results.

Gradient descent is primarily a scalar objective optimization algorithm. There are special methods designed for multi-objective optimization e.g. goal programming [48], that perform significantly better on such problems.

5.1.2 Limitations of the Backpropagation Mechanism

While error rate minimization maps directly to the mechanism of error backpropagation, the same mechanism does not naturally suit the calculation of certain other measures. For simplicity, let us consider the case of sensitivity in medical diagnosis. As defined previously, sensitivity value represents the ratio of true positive test results, to the total positives in the dataset. As we can see, in order to compute sensitivity, the algorithm must have classified all instances.

In the conventional backpropagation algorithm (also known as *sequential mode backpropagation* [34]), error values are calculated per instance, rather than per epoch. This makes it impossible to have cumulative measures like sensitivity available for error measurement.

One might consider the use of *batch mode backpropagation* [34] in such a scenario. In case of batch mode backpropagation, the error calculations and weight updates are performed only once per epoch. This makes it possible to calculate cumulative measures like sensitivity. However, if we take a closer look at the error backpropagation procedure, weight updates are calculated per output node i.e. the update per output node depends on the correct and incorrect classification outputs on that node. Therefore, in a multilayer perceptron with 3

output nodes, even if we calculate the value for sensitivity per epoch; the question arises of how to map a single scalar value over 3 output nodes so that the weight update represents this error rather than the error per output node.

The above mentioned scenario indicates that such cumulative measures can only be calculated by making modification to the backpropagation mechanism, and are not supported by the standard procedure.

5.2 Limitations of the Representational Bias

It is important to note that the structure of a neural network plays a critical role in the ability of the network to perform adequately. The number of nodes in the network, the synaptic connections and orientation of these connections are very important. Therefore, we consider it important to present some of the techniques that make use of improving the representational bias of the network. Following subsections discuss some of these techniques.

5.2.1 Receptive Local Fields

For certain applications where suitable prior information is available, problem specific specialized architectures can be built in order to improve classification performance. A commonly used method of doing so is the use of *receptive local fields* [34]. For a partially connected network with one hidden layer, each hidden neuron could have limited synaptic connections to only some of the input neurons. This way, input synapses to each hidden neuron are different, and represent some problem specific bias. A special type of neural networks that makes use of this method is known as a *convolutional network* [34].

The problem with this method is the requirement for the availability of prior information. In most of the real life problems, prior information is simply not available, in which case this method is inapplicable.

5.2.2 Network Pruning

It is important for a classifier to have a good generalization capability. In neural networks, this can be achieved by keeping the size of the neural network to a minimum [34]. This is because a small network is less likely to learn noise in the dataset and thus generalize better when presented with unseen instances.

One method of achieving so is to start learning with a rather large sized network that gives a good enough performance, and later on weaken or eliminate unnecessary synaptic weights. This procedure is generally known as network pruning. There are several pruning techniques that can be applied to neural networks; one of which is e.g. *optimal brain damage* [49].

5.2.3 Constructive Networks

The problem of limiting the size of the network can be taken from a different angle as well. In the human brain a very important part of learning of complex problems occurs during childhood [50] e.g. learning a language for the first time. The brain starts small and simple, and then improves its performance over time.

The same concept when applied to artificial neural networks is known as *network growing* [34]. Constructive neural network learning algorithms start with a relatively small network

and add neurons on the fly as per requirement, until a satisfactory solution is found. One popular category [34] of such algorithms is the algorithms that use iterative weight update.

6 DISCUSSION

We will now make an effort to evaluate the feasibility of measure-based learning and BPNN. Also, we try to identify alternative solutions to certain BPNN bottlenecks.

6.1 Objective Function Replacement

A measure-based learning algorithm has to have a replaceable objective function so that it could be biased towards any required performance measure. Therefore, the most important question that arises is whether or not BPNN are suitable enough for objective function replacement.

The extensive literature survey carried out in this thesis shows that there have been successful attempts at replacing the inherent MSE objective function with other objective functions. Cross Entropy (CE) and *Mean Cross Entropy* (MCE) are often used as objective functions in probabilistic settings, and empirical studies exist that prove these measures have shown promising results [51].

Another interesting objective function is the *Classification Figure of Merit* (CFM) [52]. While the traditional method in error backpropagation is to minimize the difference between the classifier output and the target output, the CFM objective function requires a different calculation. It seeks to maximize the difference between the values of the output node that is supposed to be in the *on* state, and all the other nodes that are supposed to be in the *off* state. The overall impact of using this objective function is that the set of misclassification tokens is markedly different from the one produced when MSE or CE are used as objective functions.

Cost-sensitive learning is also a very interesting approach for bias replacement. It has been shown [20] that introducing a cost factor (representing the misclassification cost for each instance) term to the MSE objective function enables the backpropagation algorithm to be biased towards performing better at classifying a certain class than the other classes. In a binary classification problem e.g. medical diagnostics, such a mechanism can be used to bias the algorithm towards improving the sensitivity measure. This can be useful in certain situations where it is vital to make sure sensitivity score is always high. The drawback of this particular method is that it requires misclassification costs to be known in advance. This may be possible in some applications, but for most, this is not the case. Hence, this method cannot be considered as a generic solution.

Even though all three of the above mentioned objective functions are not exactly the same, they do have some similarities that make it possible for them to be used as objective functions for the BPNN gradient search. Following are the common properties:

- All functions are differentiable.
- All functions primarily try to minimize classification errors, and thus can be rightly called as *error functions*.
- All functions comply with the standard error backpropagation mechanism. Even though CFM differs in terms of how the error is calculated in the first place, the backpropagation mechanism is the same as used by the sequential mode backpropagation procedure.

It is uncommon to see any objective functions in use with BPNN that do not comply with the above mentioned properties. We assume that this is primarily due to the reasons we have mentioned in the previous chapter.

Hence, it can be safely concluded that in order for an objective function to be useable with the conventional BPNN, it must possess the above mentioned characteristics. This can be considered as a major bottleneck in implementing BPNN as a measure-based learning algorithm. This is due to the fact that for a measure/objective function to be generic, it should not be this constrained in characteristics. Such a limitation would typically apply when we try to formulate a multi-objective measure function that combines quantitative and qualitative measures into a single function.

6.2 Significance of the Representational Bias

In the previous chapter, we presented some of the methods used for altering the representational bias of a neural network in order to improve performance. We also mentioned that certain representational characteristics contribute towards increasing the generalizability of the network. In this section, we present further arguments to establish the significance of the representational bias.

6.2.1 The Bias/Variance Dilemma

As mentioned previously, in every practical situation, every learning algorithm generates classifiers that are prone to errors i.e. all classifiers make mistakes [1]. There are basically two sources of errors; 1) error of the learning algorithm, and 2) error from the dataset.

The error rate for a particular algorithm is called its *bias*, which for a certain problem matches how adequate the learning algorithm is for that problem. The datasets used for training are finite and do not represent the entire population of instances. The expected value of this source of error is referred to as *variance* of the learning algorithm for a certain problem.

In an ideal case, we would like for both bias and variance to be minimal. However, in practice, a small value of bias causes a larger variance and vice versa. This phenomenon is known as the *bias/variance dilemma* [16].

In artificial neural networks, this problem can be overcome to a certain extent by introducing a bias that does not have harmful effect on the classification task. The introduction of such bias makes it possible to significantly reduce the variance.

A practical method for introducing such a bias is by using a *constrained network* [34]. This can be done by using the local receptive fields method introduced in the previous chapter. As we already know, the use of local receptive fields is an alteration of the representational bias of the network. This highlights the significance of the structural representation.

6.2.2 Relationship between Representational and Procedural Bias

As we know, the backpropagation algorithm is responsible for carrying errors back from the output nodes and performing gradient descent for optimal weight updates. However, the learning algorithm is dependent on the structural representation in two ways.

The first would be the dependency on the synaptic connections. The synaptic connections act as the media for propagation of errors. The presence or absence of a synaptic connection

determines whether a weight update is possible at all, and what would be the final weights on the target neuron.

Secondly, there is a dependence on the presence or absence of neurons. We have mentioned before that neurons in the hidden layer act as feature detectors. Another important role of these neurons is that they determine the partitioning of the instance space [34]. Therefore, the number of these neurons can have a significant impact on how the instance space is partitioned, and thus limit or expand the possibility for the learning algorithm to find a global optimum.

Similarly, it is fairly straight forward to see that the structure of the network is meaningless unless the learning algorithm can exploit it. If the objective function is inappropriate, or the performance of the optimization procedure is inadequate for any reason, desirable results cannot be achieved by altering the representational bias alone.

6.2.3 Conclusion

The above discussion brings an important point to surface. In order to be able to use a generic measure function, objective function replacement alone might not be the optimal solution. Instead, a better approach would probably be the one that takes into account optimization of both the learning algorithm and the network structure.

6.3 Using Global Search Algorithms

As highlighted in the previous chapter, gradient descent is primarily a local search procedure and therefore highly prone to falling into local optima. However, the advantage of using a global search algorithm instead of gradient descent is two fold.

The first advantage is that a global search algorithm has a better chance of finding the global optima. Even though gradient search suffers from the local search problem, there have been efforts to improve its performance. For example, it is very common to use the algorithm with a momentum term that helps it find its way out of local minima. Despite the local search problem, gradient descent has shown to produce excellent results for various applications. Therefore, we do not consider this advantage of global search as absolutely vital.

Depending on the choice of the global search algorithm, we can get another advantage. Genetic algorithms (GA) have been used for neural networks in several reported studies, and shown promising results [47]. The research on the use of GA for neural networks is being carried out in two major directions. One direction is where the GA is used to find the optimal topology of the network. The second direction, and the one we will consider further, is the use of GA to optimize network weights.

When used for search i.e. network weight optimization, GA not only provides a global search; the method also makes it possible to define objective functions having a relatively wide range of characteristics. This property of GA makes it possible to define objective functions that do not possess the characteristics necessary for gradient search e.g. differentiability. Moreover, since GA does not require error backpropagation; limitations identified earlier for the error backpropagation mechanism no longer apply.

To summarize, for the purpose of achieving measure-based learning with neural networks; GA appears to be a more suitable training algorithm than gradient search.

7 CONCLUSIONS AND FUTURE WORK

So far in the thesis we have presented and discussed several characteristics of BPNN and some of the most commonly used performance measures; and tried to establish correlations between these performance measures. We have analyzed these characteristics and correlations in the context of measure-based learning, which leads us to draw the following conclusions:

- The error backpropagation mechanism and the gradient descent procedure limit the set of usable objective functions. These limitations do not favor the concept of a generic measure function since the possible combinations of measures are very limited. The set of useable measures does not even include some of the most commonly used quantitative measures, while for a generic measure function it might even be required to cater for qualitative measures. Therefore, neither the standard error backpropagation mechanism nor the gradient descent method is suitable for measure-based learning.
- Our study reveals that optimization of the representational bias has a vital significance for neural networks. Therefore, a network would probably generate much better results if the measure functions for the procedural bias are used in conjunction with algorithms for optimizing the representational bias.

Even though the standard BPNN do not appeal to measure-based learning, the identification of bottlenecks provides a roadmap for carrying out further research. The correlations that we have established throughout this research indicate that the concept of measure-based learning does apply to neural networks in theory. However, at the moment we do not know how to materialize it. We need to develop methods to find the most suitable combination of network architectures and optimizing algorithms that would allow for the use of generic measure functions for learning.

7.1 Future Work

Following is a brief discussion of some research directions that we think might contribute to a great deal to the topic of measure-based learning.

- A possible research direction is to explore ways to enable the neural network to optimize certain commonly used measures. The possible set of objective functions can be very limited in this case. However, the set should include all the most commonly used quantitative measures including multivariate measures. This direction is somewhat on the lines of research carried out by Joachims [27].
- We have discussed in this thesis the concept of similarities in performance measures. Some work in exploring this area has already been done. However, although the existing research helps better understand the equivalences among the different measures, it is still not apparent how to exploit these similarities in the context of measure-based learning. We believe that there is potential for further research. It would be interesting to explore how the similarities between the measures can be exploited for the development of generic measure functions.

The significance of finding these similarities and establishing mathematical correlations among different measures lies in the ease with which it would be possible to model algorithms that could optimize multiple measures with adequate

performance. It would be much simpler to define generic functions that could accommodate different measures. This in turn makes it simpler to realize measure-based learning algorithms.

- For artificial neural networks, the optimization techniques we have mentioned so far are all numeric optimization models. The origins of neural networks are biological, but so far we do not know what algorithm is used for learning in the human brain [32]. We do not know whether the same learning algorithm applies to all types of neurons; or whether different algorithms are specialized at learning in different specialized parts of the brain.

It would be interesting to explore the state-of-the-art in computational neuroscience and try to investigate whether models of optimization from the human brain can be used to derive more efficient and effective algorithms for optimizing neural network weights.

- We have discussed that the gradient search method is primarily a local search algorithm. Therefore, it seems intuitive to suggest that global search algorithms should be explored in this context; for some algorithms might possess more suitable characteristics for the use of generic measure functions.

The use of genetic algorithms for weight space optimization appears to be a promising alternative to gradient descent due to its global search capacity, as well as the ability to use a wider range of objective functions as compared to gradient descent.

- BPNN are a good candidate for exploring the applicability of measure-based learning due to the very clear distinction between their representational and procedural bias, as well as the fact that it is fairly simple to correlate the learning problem to optimization for BPNN. However, several other machine learning algorithms are widely used in various application areas, and it would be interesting to extend this investigation to these algorithms.

BIBLIOGRAPHY

- [1] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*, second edition. Morgan Kaufmann, 2005.
- [2] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. "Learning Internal Representations by Error Propagation". *Parallel Distributed Processing*, Vol.1, pp.318–362, 1986.
- [3] C. Cortes and V. Vapnik. "Support Vector Networks". *Machine Learning*, Vol.20, pp.273-297, 1995.
- [4] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
- [5] Leo Breiman. "Random Forests". *Machine Learning*, Vol. 45, No.1, pp.5-32, 2001.
- [6] D. H. Wolpert. "Stacked Generalization". *Neural Networks*, Vol.5, pp.241–259, 1992.
- [7] Cortes, C. and Mohri, M. "AUC Optimization vs. Error Rate Minimization". In *Advances in neural information processing systems 16*. MIT Press.
- [8] Niklas Lavesson and Paul Davidsson. "A Multi-dimensional Measure Function for Classifier Performance". In Proc. of the *Second IEEE International Conference on Intelligent Systems*, 2004.
- [9] Urney, P. "How to Shift Bias: Lessons from the Baldwin Effect". *Evolutionary Computation*, Vol. 4, pp.271-295, 1997.
- [10] Lawrence O. Hall, Xiaomei Liu, Kevin W. Bowyer, Robert Banfield. "Why are Neural Networks Sometimes Much More Accurate than DecisionTrees: An Analysis on a Bio-Informatics Problem". In the Proc. of the *IEEE International Conference on Systems, Man & Cybernetics*, 2003.
- [11] Peter Jarvis, John Holford, Colin Griffin. *The Theory and Practice of Learning*. RoutledgeFalmer, 2005.
- [12] Herbert Simon. "Why Should Machines Learn?". *Machine learning: An artificial intelligence approach*, Morgan Kaufmann, 1983.
- [13] Tom M. Mitchel. *Machine Learning*. McGraw-Hill, 1997.
- [14] Dorronsoro, J. R., Ginel, F., Sanchez, C., Cruz, S. "Neural Fraud Detection in Credit Card Operations". *IEEE Transactions on Neural Networks*, Vol. 8, pp.827–834, 1997.
- [15] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
- [16] Ethem Alpaydin. *Introduction to Machine Learning*. MIT Press, 2004.
- [17] Johannes Fürnkranz. "Separate-and-Conquer Rule Learning". *Artificial Intelligence Review*, Vol. 13, pp.3-54, 1999.
- [18] D. W. Aha, D. Kibler, and M. K. Albert. "Instance-based Learning Algorithms". *Machine Learning*, Vol. 6, pp.37–66, 1991.

- [19] R. Caruana and A. Niculescu-Mizil, "Data mining in metric space: an empirical analysis of supervised learning performance criteria". In Proc. of the *10th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004.
- [20] Matjaz Kukar and Igor Kononenko. "Cost-Sensitive Learning with Neural Networks". In Proc. of the *13th European Conference on Artificial Intelligence*, 1998.
- [21] J. P. Egan. "Signal Detection Theory and ROC Analysis". *Cognition and Perception*, Academic Press, 1975.
- [22] Gordon and Desjardins, "Evaluation and selection of biases in machine learning," *Machine Learning*, Vol. 20, pp.5-22, 1995.
- [23] M. Holt and S. Semnani, "Convergence of back-propagation in neural networks using a log-likelihood cost function". *Electronics Letters*, Vol. 26, pp.1964-1965, 1990.
- [24] Sumiyoshi Fujiki, Mitsuyuki Nakao, Nahomi M. Fujiki. "Error Measures of the Back-Propagation Learning Algorithm". *Journal of the Korean Physical Society*, Vol. 40, pp.1091-1095, 2002.
- [25] C. Ennett, M. Frize, N. Scales. "Evaluation of the logarithmic-sensitivity index as a neural network stopping criterion for rare outcomes," In Proc. of the *4th International IEEE EMBS Special Topic Conference on Information Technology Applications in Biomedicine*, 2003.
- [26] C. Ferri, P. Flach, and J. Hernandez-Orallo. "Learning Decision Trees using the Area under the ROC Curve". In Proc. of the *International Conference on Machine Learning*, 2002.
- [27] T. Joachims, "A support vector method for multivariate performance measures." In Proc. of the *22nd international conference on Machine learning*, 2005.
- [28] E. Suzuki, and T. Ohno. "Prediction Rule Discovery Based on Dynamic Bias Selection". In Proc. of the *3rd Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining*, 1999.
- [29] Niklas Lavesson and Paul Davidsson. "Generic Methods for Multi-criteria Evaluation". In Proc. of the *SIAM International Conference on Data Mining*, 2008.
- [30] Alison A. Motsinger, Scott M. Dudek, Lance W. Hahn, and Marylyn D. Ritchie. "Comparison of Neural Network Optimization Approaches for Studies of Human Genetics". *Lecture Notes in Computer Science*, Springer, 2006
- [31] Lae-Jeong Park. "Learning of Neural Networks for Fraud Detection Based on a Partial Area Under Curve". *Advances in Neural Networks*, Vol. 3497/2005, pp. 922-927, 2005.
- [32] Anil K. Jain and Jianchang Mao. "Artificial Neural Networks: A Tutorial." *Computer*, Vol. 29, pp.31-44, 1996.
- [33] Michael A. Arbib. *Brains, Machines and Mathematics*. Springer-Verlag, 1987.
- [34] Simon Haykin. *Neural Networks: A Comprehensive Foundation*, Macmillan College Publishing Company, 1994.
- [35] W.S.McCulloch and W.Pitts. "A Logical Calculus of Ideas Imanent in Nervous

- Activity”. *Bulletain of Mathematical Bio-physics*, Vol. 5, pp.115-133, 1943.
- [36] R. Rosenblatt. *Principles of Neurodynamics*. Spartan Books, 1962.
- [37] M. Minsky and S.Papert. *Preceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.
- [38] P. Werbos. “Beyond Regression. New Tools for Prediction and Analysis in Behavioural Sciences”. PhD thesis, Department of Applied Mathematics. Harvard Univerity, 1974.
- [39] Judith E. Dayhoff and James M. DeLeo. “Artificial Neural Networks: Opening the Black Box”. In Proc. of the *Conference on Prognostic Factors and Staging in Cancer Management: Contributions of Artificial Neural Networks and Other Statistical Methods*, 1999.
- [40] Ronald L. Rardin. *Optimization in Operations Research*. Prentice Hall, 1997.
- [41] P. Baldi. "Gradient descent learning algorithm overview: a general dynamical systems perspective". *IEEE Transactions on Neural Networks*, Vol. 6, pp.182-195, 1995.
- [42] P. A. Flach. “The Geometry of ROC Space: Understanding Machine Learning Metrics through ROC Isometrics”. In Proc. of the *20th International Conference on Machine Learning*, 2003.
- [43] J. Fürnkranz and P. Flach. “An Analysis of Rule Evaluation Metrics”. In Proc. of the *20th International Conference on Machine Learning*, 2003.
- [44] Andrew P. Bradley. “The use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms”. *Pattern Recognition*, Vol. 30, pp.1145-1159, 1997.
- [45] A. Freitas. “Are we really discovering “interesting” knowledge from data?”. *Expert Update (the BCS-SGAI Magazine)*, Vol. 9, pp.41-47, 2006.
- [46] A. Herschtal and B. Raskutti. “Optimising Area Under the ROC Curve using Gradient Descent”. In the proc. of the *21st international conference on Machine learning*, 2004.
- [47] Randall S. Sexton, Robert E. Dorsey, John D. Johnson. “Toward Global Optimization of Neural Networks: A Comparison of the Genetic Algorithm and Backpropagation”. *Decision Support Systems*, Vol. 22, pp.171-185, 1998.
- [48] Mattew A. Kupinski and Mark A. Anastasio. “Multiobjective Genetic Optimization of Diagnostics Classifiers with Implications for Generating Receiver Operating Characteristic Curves”. *IEEE Transactions on Medical Imaging*, Vol.18, pp.675-685, 1999.
- [49] Y.L. Cun, J.S. Denker, S.A. Solla. “Optimal brain damage”. *Advances in neural information processing systems*, Vol. 2, pp. 598-605, 1990.
- [50] J. Elman. “Learning and development in neural networks: The importance of starting small”. *Cognition*, Vol. 48, pp.71-99, 1993.
- [51] Michael D. Richard and Richard P. Lippmann. “Neural Network Classifiers Estimate Bayesian a posteriori Probabilities”. *Neural Computation*, Vol. 3, pp.461-483, 1991.
- [52] J. Hampshire and A. Waibel, “A novel objective function for improved phoneme

recognition using time-delay neural networks". *IEEE Transactions on Neural Networks*, Vol. 1, pp. 216-228, 1990.

- [53] A. Andersson, P. Davidsson, J. Lindén. "Measure-based Classifier Performance Evaluation". *Pattern Recognition Letters*, Vol. 20, pp.1165–1173, 1999.
- [54] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2003.
- [55] R. Mattison. *Data Warehousing and Data Mining for Telecommunications*. Artech House, 1997.