UNIVERSITY OF NAIROBI

COLLEGE OF BIOLOGICAL AND PHYSICAL SCIENCES

SCHOOL OF COMPUTING AND INFORMATICS

Project Title:

Prediction of River Discharge Using Neural Networks

A project report submitted in partial fulfilment for the Award of a Master of Science Degree in Information Systems

BY

CHARLES ODIRA MAXWELL

REG NO: P56/66952/2011

SUPERVISOR: DR. Elisha O. Abade

NOVEMBER 2014

ABSTRACT

Obtaining accurate monthly river flow discharge prediction has always been a challenging task in water resources management. In this study we developed and tested feedforward neural network architectures optimized with two algorithms namely Levenberg-Marquardt and resilient back-propagation with different activation functions in hidden and output layers in predicting monthly river flow discharge. The networks were trained and tested with 58 years of data using Matlab software.

Prediction accuracy was measured by means of mean square error (mse) and correlation coefficient (r).The results showed more accurate prediction for Levenberg-Marquardt algorithm with sigmoid activation function at the hidden layer and linear activation function at the output layer.

The study concludes that accuracy in river flow discharge prediction can be improved by feedforward neural network optimized with Levenberg-Marquardt algorithm with sigmoid function in hidden layer and linear activation function in output layer.

DECLARATION

I, Charles Odira Maxwell, do hereby declare that this research project is entirely my own work and where there is work or contributions of other individuals, it has been duly acknowledged.

To the best of my knowledge, similar research work has not been carried out before or previously presented to any other educational institutions in the world of similar purposes or forum.

Sign: --------------------------------- Date: -----------------------------------

Name: Charles Odira Maxwell

Reg.No. P56/66952/2011

Supervised By:

This project has been submitted in partial fulfilment of the requirement for the Master of Science degree in Information Systems of the University of Nairobi with my approval as the University supervisor.

Sign: ------------------------------- Date: ------------------------------------

Dr. Elisha O. Abade

DEDICATION

To my dear wife, Rose Adhiambo Odira

Daughters, Lynn, Belinda and Sandra Odira

And

The entire family of Maxwell Oloo Sala

ACKNOWLEDGEMENT

First, I wish to thank God for his kindness for without him, I wouldn't have reached this far.

Second, my sincere gratitude goes to my supervisor Dr. Elisha Abade and panellists, Mr. Christopher Moturi, Mr. Joseph Ogutu and Mr. Evance Miriti for their guidance, positive criticism and above all their valuable time and advice throughout my project work.

Thanks also to Dr. Simon Otieno Dulo, Senior lecturer department of Civil Engineering, University of Nairobi for the valuable link provided to WRMA which made my research data acquisition possible, Prof S.K Mwea Chairman, department of Civil Engineering for the continued support and encouragement and Mr.Januarius Agullo lecturer department of Environmental and Biosystems Engineering for valuable discussions related to my project.
Titus Muhambe and Stephen Ochieng; your support and encouragement and prayers enabled me to put more efforts whenever things appeared difficult.

Finally, I wish to express my appreciation to my parents Maxwell and Roselyn for the sacrifice they made for the sake of my education, My wife Rose and children; Lynn, Belinda and Sandra, for their unconditional support, love, joy, and encouragement throughout my study. Without them, this project would not be successful.

God bless you all

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

DEFINITIONS OF KEYWORDS

Artificial Neural Networks- Artificial neural networks are computational metaphor which were inspired by the studies of brain and nervous systems in biological organisms (Haykin, 2008).

Training algorithm – is a mathematical procedure used to automatically adjust the network's weights and biases in search of the optimal ANN connection weights by minimizing the error between the network output and the target output (Talaee, 2012).

Activation functions - also known as transfer function determines the relationship between inputs and outputs of anode and a network.

Learning in ANNs is self-adjustment of the weights as a response to changes in the information environment (Haykin, 2008).

Epoch – each pass through all the training patterns (Majdi et al., 2013).

Learning rate - determines the magnitude of weight changes (Kumar et al., 2004).

Error function - is the function that is minimized during training (Haykin, 2008).

Gauging station- a section in a river where flow is measured.

Network architecture - is the number of hidden layers, number of nodes in the hidden layer and the number of connections (Gomes et al., 2011).

Stream flow/discharge – quantity of water passing through a river channel measured in m^3/s.

Matlab Neural Network Tool - is simulation software programme that allows one to build, train and evaluate neural networks

Hydropower station- is where falling water is used to generate renewable energy. It uses a dam on a river to store water in a reservoir. Water released from the reservoir flows through a turbine, spinning it, which in turn activates a generator to produce electricity.

Overfitting

Case in which the error on the training set is driven to a very small value, but when new data is presented to the network, the error is large.

LIST OF ABBREVIATION AND ACRONYMES

ANN – Artificial Neural Network

NNA – Neural Network Architecture

FNN – Multi-Layer Feed-Forward Neural Networks

MLR - Multiple Linear Regression

MLP – Multilayer Layer Perceptron

LM – Levenberg-Marquardt

K-NN - K-nearest Neighbour

KenGen - Kenya Energy Generation Company

WRMA – Water Resources Management Authority

ARMA - Autoregressive Moving Average

MSE -Mean Square Error

R – Correlation Coefficient

GDX - Gradient Descent with Adaptive Learning Rate

BPA – Back-Propagation Algorithm

RBP – Resilient Back-Propagation

sqrt – square root

CHAPTER ONE: INTRODUCTION

1.1 Background

Flow prediction is an essential aspect in many of the activities associated with the planning and operation of the components of water resources system. Reliable stream flow prediction model is important in assisting water resources managers and engineers in allocation of water to competing users like hydropower, irrigation and domestic. Hydrologic component requires both short term and long term forecasts of stream flow events in order to optimize the system or to plan for future expansion or reduction (Kisi 2005).

Several models for stream flow prediction exist which according to (Talaee, 2012 and Kisi et al., 2007) may broadly be grouped under three main categories: (1) conceptual models (2) physically based models and (3) black-box models or data-driven models. Conceptual models rely on simple arrangement of relatively small number of interlinked conceptual elements, each representing a segment of land phase of a hydrologic cycle. The physically based models are specifically designed to mathematically simulate or approximate (in some physically realistic manner) the general internal sub-processes and physical mechanisms that govern the stream flow process, whereas the black-box models or data-driven models are designed to identify the connection between the inputs and the outputs, without going into the analysis of the internal structure of the physical process. Data-driven models provide an alternative to physically based model due to its complexity (Lin et al., 2009).

Of the many types of data-driven approaches from simple regression models to more complex fuzzy logic type models, the artificial neural network (ANN) approach has many attractions. They are relatively easy to formulate, insensitive to noise, parallel in nature and can be adopted in real time situation. According to Wang et al., (2005) since 1990's artificial neural network (ANN), based on the understanding of the brain and nervous system is gradually used in hydrological prediction. Kisi (2005) posits that artificial neural networks (ANN) have been successfully applied in a number of diverse fields including water resources and as noted by Tombul et al., (2006) the artificial neural network models have been used increasingly in various aspects of science and engineering because of its ability to model both linear and nonlinear systems without the need to make any assumptions as are implicit in most traditional statistical approaches.

Artificial neural network (ANN) mimic the way the human brain learns from examples through a process that involves finding an optimal set of weights for the interconnections between nodes (neurons) of the network arranged in groups called layers. A network usually has three layers: the input layer—a set of data, in this case historical time lagged average monthly river flow for the six preceding months; the hidden layer where data are processed; and the output layer where the results, in this case predicted average monthly flow for the following month. Learning (or training) in ANNs is defined as self-adjustment of the weights as a response to changes in the information environment, a transfer function that controls the generation of the output of a neuron, and learning laws that describe how the adjustment of the weights are made during training (Haykin,2008 and Kisi, 2005).

1.2 Statement of the problem

Variations in the flow regime of Sondu-Miriu river poses great challenge to the operations of the 60 megawatt hydropower station situated 6.2 km from the intake in Sondu-Miriu River in Kisumu County western Kenya. The greatest challenge was experienced during the project inauguration in the year 2008 when the event had to be delayed due to inadequate water flow (www.reuters.com). According to Kenya Energy Generation Company (KenGen), Sondu-Miriu hydropower station unlike other stations in Kenya does not have a major dam and associated large reservoir but relies on the flow in the river with only a small storage capacity at the intake (www.kengen.co.ke). The flow variability is still a threat to the optimization of the hydropower station. The operation decisions rely on daily weather elements and this leads to erratic power production in west Kenya region.

Within water engineering, accurate prediction of river discharge is important. The prediction results are used by water resources managers in planning the operations of water resource systems such as hydropower station (Kisi et al., 2007 and Asaad, 2010).

In this research, artificial neural network trained with different algorithms is used with historical time series flow data to predict monthly Sondu-Miriu River discharge. A simple, accurate and efficient tool for discharge prediction is essential to the engineers and the water resource managers of this river to aid in operational planning.

1.3 Study objectives

1.3.1 General objective

The purpose of this study is to investigate the prediction ability of artificial neural networks trained with different algorithms namely Levenberg-Marquardt and resilient back-propagation with different activation functions in hidden and output layers to determine a network which gives an optimum result.

1.3.2 Specific objectives

To achieve the above overall objective, the following specific objectives are set;

a) To pre-process historical river flow data into a form that is suitable for training artificial neural networks.
b) To develop a neural network model.
c) To verify and validate the model performance.
d) To compare the models performance in predicting Sondu-Miriu River discharge under different algorithms and activation functions.

1.4 Research Questions

The following research questions will guide the study;

a) In what form should the historical river flow data be in order to train artificial neural networks?
b) To what extent does training algorithm affect artificial neural network model prediction accuracy?
c) What influence does activation function have in artificial neural network model prediction accuracy?
d) How accurate are the results obtained from the model?

1.5 Scope of the Study

The study does not cover other variables which contribute to river flow such as temperature, and precipitation. The research is applied to one river at one gauging station as a proof of concept, which is Sondu-Miriu River but the results obtained may be useful more generally to river flow predictions. The research is primarily focused on neural network performance evaluation under different algorithms and activation functions trained on historical river flow data to predict subsequent average monthly river discharge values.

1.6 Significance of the study

Convention models, including physical deterministic numerical models, are expensive and have many limitations when applied to solving water resource problems. The study intends to contribute to the current knowledge using the neural networks method to address a problem of accurate prediction of monthly river flow which is very important to engineers.

This study seeks to contribute in the following major areas to the existing body of knowledge:

- Evaluation and comparison of neural network model performance with different activation functions at the hidden and output layers through two learning algorithms.
- To the best knowledge of the researcher, neural network tool is applied for the first time to predict the discharge at Sondu-Miriu River. This is a significant contribution in terms of an alternative to the current traditional methods.
- The great adaptivity of the ANN models enables them to be simply and effectively trained for new situations or new events. Thus the model enhances the capacity and accuracy to find new patterns in river flow.

1.7 Study assumptions

- There are no hydrological developments on the upstream of the river which may alter the flow pattern.
- Climate change effects not included in the computations of the future flows.

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

Mathematical models have been used to predict the trend of river discharge. Nowadays time series analysis and multiple regression methods are the two most commonly used methods (Wang et al 2005). Since 1990s artificial neural networks based on the understanding of the brain and nervous systems is gradually used in hydrologic prediction. However, its application to water resources engineering with the aim of improving accuracy needs further investigation and study.

This chapter presents an overview of previous research work on river discharge prediction methods, introduction to artificial neural networks and its application in engineering follow the overview.

2.2 River discharge prediction methods

River flow discharge is an important parameter in the planning and operation of a water resources system. Many models have been developed to solve water resources engineering problems with powerful computers and software helping engineers to utilize the more complex models. Besaw et al., (2010) note that the current methods of predicting gauged and/or ungauged river discharge fall into four categories which are conceptual, metric, physics based and data driven. However, several authors in literature classify river discharge prediction methods as conceptual, physically based models and data driven models (Kisi eta al., 2007, Pierini et al 2012, Talaee, 2012, Yanik et al., 2011, and Kisi 2005).

Physically based models are specifically designed to mathematically simulate or approximate in some realistic manner the general internal sub-processes and physical mechanisms that govern the river discharge process such as soil moisture conditions, infiltration characteristics and land surface roughness (Talaee, 2012). They are aimed at incorporating simplified forms of physical laws of hydrological processes. Yanik et al., (2011) posit that physically based model requires an understanding of all the physical processes which impact a natural process and the interactions among them which make them very difficult. According to Jayawardena (2009) physically based models require a priori understanding of the mechanics of the underlying processes as well as a great deal of more spatially and temporally distributed data. These models are computationally resource intensive, depend on initial and boundary conditions and cannot be easily adopted in real- time environment. Due to the numerous parameters required, some parameters will be estimated using statistically based methods resulting to large uncertainties that can result to both random and systematic errors.

Conceptual models rely on simple arrangement of relatively small number of interlinked conceptual elements, each representing a segment of land phase of a hydrologic cycle (Kisi et al., 2007). They incorporate simple conceptualization of hydrologic processes and require considerable data and human effort to calibrate, validate and test. Most conceptual flow models have been developed to incorporate simplified forms of physical laws which are generally nonlinear, time-invariant, and deterministic, with parameters that are representative of watershed characteristics such as topology, vegetation, and soil type. While conceptual models are important in understanding hydrologic processes, there are many practical situations such as streamflow forecasting where the main concern is making accurate predictions at specific watershed locations. In such a situation, a hydrologist may prefer not to expend the time and efforts required in developing and implementing a conceptual model or numerical model, but instead implement a simpler system model, such as artificial neural network. Kisi (2005) attributes the problems with conceptual models as having empirical regularities or periodicities that are not always evident and can often be masked by noise. Due to limited recources associated with developing and calibrating physically based and conceptual models, data-driven hydrological methods have been widely adopted for streamflow prediction.

Data-driven models are based on the transfer functions that relate inputs with outputs and generally do not have any physical basis (Kisi et al.,2007). In hydrological cycle, they require pairs of input-output training data to capture the non-linear climate flow relationships. Avariety of data-driven models are available including K-nearest neighbour (K-NN) based on local approximation which makes use of only nearby observations of the point estimate. According to (Wu et al., 2009), K-NN method can not predict discharge values higher than the historical discharge and this severely restricts its use in actual forecasting. Other data driven methods such as vector support machines, instance-based learners, decision trees,multiple linear regression(MLR) and variations of autoregressive moving average (ARMA) are commonly applied in prediction applications(Carrier et al.,2013 and Wang et al.,2008).Govindarajo (2000) as cited in Besaw et al.,(2010) observe that even though the data-driven techniques often require similar data as the other models, they require much less development time, are useful for real-time applications and have proven capable of accurately predicting steam flow.

Comparative studies among the data-driven methods have been carried out by different authors on different applications and as noted by Kisi (2005), artificial neural network approach may provide a superior alternative to the autoregressive moving average models for developing input-output simulations and forecasting models in situations that do not require modelling of the internal structure of the watershed. Demissie, (2008) compared the performance of the four data-driven models which were artificial neural netwoks, support vector machine, decision tree and instance based weighting to study data-driven models to enhance physically based groundwater model prediction and concluded that the overall accuracy obtained using ANN is slightly better than that of the other three models but requires a larger training data and time.

With several authors pointing out the advantages of artificial neural networks (ANN) data-driven model over the other data-driven models, the thrust of this research is on the investigation of the ability to improve the prediction accuracy of a multi-layer feed-forward artificial neural networks which according to Coulibaly et al., (1999) as reported by Kisi (2005), about 90% of experiments make extensive use of the multi-layer feed-forward neural networks (FNN) trained by the standard back-propagation algorithm. Jayawardena (2009), applied multi-layer feed-forward neural networks (FNN) trained by the standard back-propagation algorithm in predicting daily flow in Mekong River in Asia, Pierini et al., (2012), applied multi-layer feed-forward neural networks (FNN) trained by the standard back-propagation algorithm in prediction of water flows in Colorado River, Argentina and concluded that artificial neural networks provide more reliable forecasts, especially of discharge flows. Neural networks have not only been successfully applied to time series predictions but also in other fields like in brain modelling, financial modelling to predict stock, shares and currency exchange rates, computer games, control systems, pattern recognition, data analysis and much more.

2.3 Artificial Neural Networks

Artificial neural networks are computational metaphor which was inspired by the studies of brain and nervous systems in biological organisms. It is a branch of artificial intelligence developed in the 1950s aiming at imitating the biological brain architecture which solves problems through a series of repeated observations between neurons and synapses within the brain (Tan et al., 2010). According to (Haykin, 2008) artificial neural network is a machine that is designed to model the way in which brain performs a particular task or function of interest. It is usually implemented using electronic components or is simulated in software on a digital computer. The software tools are designed as a system of nodes and connectors which find relationships between given sets of inputs and outputs. The nodes represent the neurons which are the processing elements within the neural network with the natural propensity for storing experiential knowledge and making it available for use. Artificial neural network is therefore a network of layered nodes connected with directed arcs each with a numerical weight specifying the strength of connection which are automatically adjusted during training of the network. According to (Nkoana, 2011) the weights of connections encode the knowledge embedded in the network and the "intelligence" of a neural network emerges from the collective behaviour of neurons, each of which performs only very limited operation with each individual neuron finding a solution by working in parallel.

According to (Xiaofeng et al., 2014), artificial neural network can be grouped into two major categories based on the connection pattern (architecture) as (i) feedforward networks in which no loop exists in the graph. The weighted connections feed activations only in the forward direction from the input layer to the output layer and (ii) feedback (or recurrent) networks in which loops exist because of feedback connections. Additional weighted connections are used to feed previous activations back into the network. The most common family of feedforward networks is a multilayer feedforward network in which neurons are organized into layers with connections strictly in one direction from one layer to another. It is important to point out that there are numerous variants of each of these networks. In this study, the focus is on multilayer feedforward networks but according to (Zhang et al., 1998), it is worth noting that recurrent networks also play important roles in forecasting and some studies such as Kumar et al., (2004) compared the performance of both feedforward network and recurrent networks in forecasting monthly river flows and concluded that recurrent networks performed better in forecasting monthly river flow.

2.3.1 Feed-forward Neural Networks

The multilayer feed-forward networks also referred to as multilayer perceptrons (MLP) are the most widely studied and used network models. Wong et al., (1997) posit that about 95% of business applications of neural network use this model while in water resources applications (Jain et al., 1996, Majdi et al., 2013, Kisi 2005 and Kumar et al., 2004) observe that multilayer feed-forward networks are the most commonly used architecture. The basic of the MLP consists of three layers; input layer, hidden layer and output layer which make it ideally suitable to model the relationship between a set of input variables and one or more output variables. According to

Majdi et al., (2013), they are appropriate for any functional mapping problem where we want to know how a number of input variables affect the output variables hence their use in non-linear engineering problems.

Structurally, an MLP is typically composed of several computing units called neurons, cells or nodes organized into layers. The number of nodes in the input layer depends on the number of input variables from the external environment. According to Joorabchi et al., (2007) the optimum number of hidden layer(s) is determined by trial and error but several authors note that one hidden layer has been shown to be sufficient and can learn to approximate virtually any function to any degree of accuracy (Talaee, 2012, Kisi, 2005, Pierini et al., 2012 and Kumar,2007). However, for determination of the number of nodes in the hidden layer, some rule of thumb is available such as the geometric pyramid rule proposed by Masters, (1993) which states that for a three layer network with *n* input and *m* output neurons the hidden layer would have sqrt(*n***m*) neurons. Karim, (2009) indicates that the number of neurons in the hidden layer can also be obtained using constructive algorithm. Carcano et al., (2008) observe that there are no hard and fast rules regarding the selection of network topology, network architecture, training algorithm and input selection decisions which affect the network performance but are left to the modeller's experience and judgment.

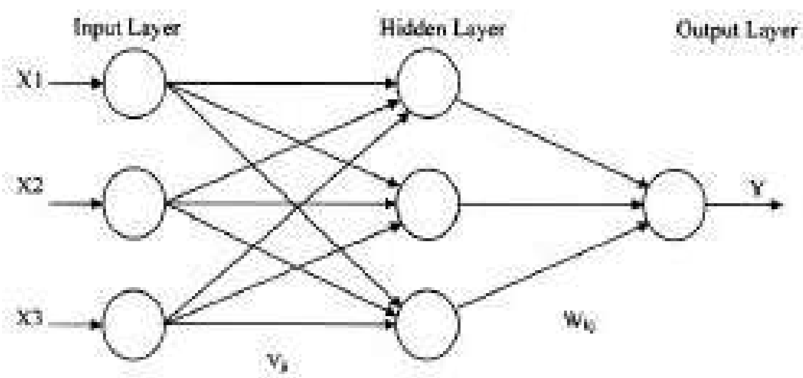

Figure 2.1: Schematic representation of a Multilayer feed-forward ANN (Source; Kisi 2005)

Each interconnection in an ANN has a strength that is expressed by a number referred to as weight. The data passing through the connections from one neuron to another are multiplied by weights that control the strength of a passing signal. When these weights are modified, the data transferred through the network changes; consequently, the network output also changes. The signal coming out from the output node(s) is the network's solution to the input problem. The weight adjustment is done according to some learning algorithm. Inputs to an ANN can be binary or real valued. In order to compute the single value output for each neuron, the weighted sum of the inputs to a neuron is used in an equation called a transfer or activation function.

2.3.2 Activation functions

Activation function also known as transfer function determines the relationship between inputs and outputs of anode and a network. They are applied to the weighted sum of the inputs of a neuron to produce the output by performing mathematical operations on the outputs of the neutron. The input layer receives the inputs from the training data and the hidden layer and output layer receives it from the interconnections. Neurons use transfer functions to produce their output.

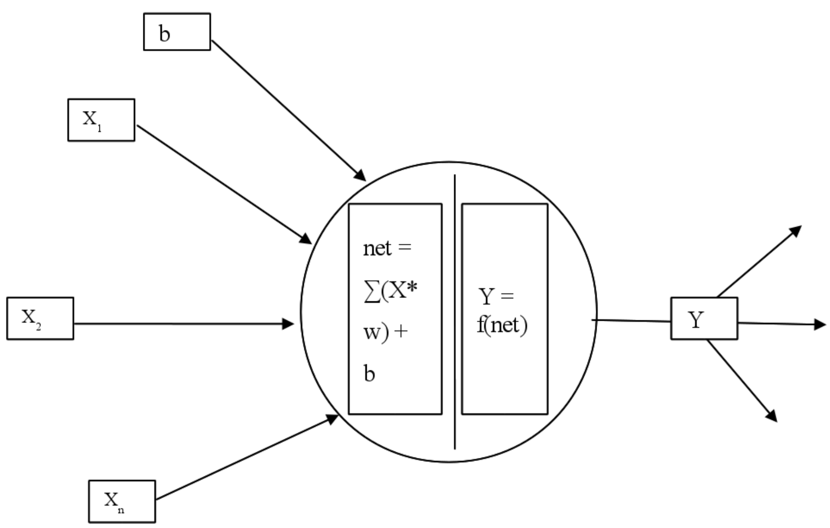

Figure 2.2: Schematic representation of the transformation inside a neuron unit (Nkoana, 2011)

According to Abhishek et al., (2012) and Majdi et al., (2013), activation functions are selected according to the types of problem to be solved by the network and the commonly used include;

1) The sigmoid (logistic) function:

   $f(x) = (1 + exp(-x))^{-1}$;

2) The linear function:

   $f(x) = x$;

Logistic transfer function is the most popular choice among researchers for both hidden and output nodes (Pierini et al., 2012, Talaee, 2012, Kisi, 2005, jayawardena,2009 and Abhishek,2012) all have applied the sigmoid function to both hidden and output nodes of a multi-layer feed-forward neural network. The sigmoid function is considered attractive activation function in ANN, because it combines nearly linear behavior, curvilinear behavior and nearly constant behavior, depending on the value of the input. The sigmoid function is sometimes called a squashing function, since it takes any real valued input and returns an output bounded between (0,1)Pierini et al.,(2012).

However, Zhang et al., (1998) observe that sigmoid activation function seems well suited for the output nodes for many classification problems where the target values are often binary while for forecasting problems which involves continuous target values, it is reasonable to use a linear activation for output nodes. Majority of researchers use sigmoid activation functions for hidden nodes while there is no consensus on which activation function should be used for output nodes. As cited in Zhang et al., (1998) Klimasauskas, (1991) suggests logistic activation functions for classification problems in the output layer where target values are often binary. Sivapragasam et al., (2014) note that several researchers also concentrate on ANN architecture without much attention to the learning algorithms and activation functions used in the ANN while Aqil et al., (2007) observe that some authors look at the learning algorithms in isolation without including activation functions in their analysis.

2.3.3 Training of Artificial Neural Networks

The artificial neural network training is a problem of nonlinear optimization, which searches the optimal ANN connection weights by minimizing the error between the network output and the target output. Neural networks are trained with a set of typical input/output pairs called training set and it is generally assumed that a network does not have any a priori knowledge about the problem before it is trained (Karunanithi et al., 1994). At the beginning of training, the weights are initialized either with a set of random values or based on previous experience. Next, the weights are systematically changed by the learning algorithm such that, for a given input, the difference between the ANN output and the actual output is small. Many learning examples are repeatedly presented to the network, and the process is terminated when this difference is less than a specified value. At this stage, the ANN is considered trained and the final weight vector of a successfully trained neural network represents its knowledge about the problem. There are three main training methods namely;

Supervised Learning: Where networks inputs as well as corresponding output is given to the networks. The errors or discrepancies between the desired and actual response for each node in the output layer are found to determine weight changes in the net according to the prevailing learning rule.

Unsupervised Learning: In this type of learning external teacher is not present or the desired output is not known in advance and learning is based upon clustering technique.

Reinforcement learning: This is a hybrid of both supervised and unsupervised learning. It has reward for correct outputs and penalty for wrong outputs.

Performance of a network is sensitive to learning parameters which include learning rate, momentum value, error function, epoch size and gain of the transfer function (Kumar et al., 2004)

Learning rate: is a constant used in error back-propagation learning that affects the speed of learning. The smaller the learning rate, the more steps it takes to get to the stopping criterion (Nkoana, 2011). Talaee, (2012) observes that it is not practical to determine the optimal setting for the learning rate before training but should be decreased as training progresses since the network weights tend to approximate the desired function more closely as training continues.

Momentum value: this parameter is used to prevent the system from converging to a local minimum. It is an additional term that is added to the weight values, when the weights of the network are updated after each epoch and is a fraction of the previous weight update values. If the momentum value is low, it can prevent the network from learning (Kumar et al., 2004).

Error function: it is the function that is minimised during training and can be used as the criteria to decide when the training process should be stopped. It determines whether the neural network has been optimally or sub-optimally trained when it reaches a sufficiently small value or when changes in the training error remain within a small interval.

Epoch size: is equal to the number of iterations or training samples presented to the network between weight updates. Network weight adaptation can be done in on-line where the weights are updated after the presentation of each training pair or batch mode where the weight changes for each training pair is accumulated, and the sum of these changes is applied after presentation of the entire training set, for example after one epoch (Nkoana, 2011).

The most commonly used method in many neural network applications is supervised learning because in most classifications and predictions, there are known outputs and their associated inputs to present to the networks (Zhang et al., 1998, Pierini et al., 2012 and Kisi 2005). Network training requires a prescribed set of well defined rules for the solution of a learning problem called learning algorithm (Haykin, 2008). Several learning rules have been proposed depending on the chosen minimising method. The most popular method has been the back-propagation learning rule introduced by Rumelhart et al., (1986) and other extensions of back-propagation which use the steepest gradient descent. However, back-propagation has some drawbacks such as long training duration with a high number of iterations. Due to the back-propagation drawbacks, other more efficient rules have been proposed, for example a descent inspired by second order minimisation methods (Talaee, 2012 and Kisi et al., 2007). Amongst these second order methods the "Levenberg-Marquardt" learning rule is at present the most powerful and leads in a few iterations "epoch" to a very satisfactory solution (Talaee, 2012). According to Zhang et al., (1998), it is important to note that so far there is no algorithm which can guarantee global solution for general nonlinear optimization problems like those in neural network training therefore the best thing which can be done is to use the available optimization method which can give the "best" local optima if true global solution is not available. The existence of many different optimization methods provides various choices for neural network training.

Back-propagation Training algorithm

Training algorithms are local or global depending on the error minimization strategy. The back-propagation algorithm introduced by Rumelhart et al., (1986) is the best local algorithm which follows the steepest-descent approach based on the first-order gradient of the slope of the objective function. A set of inputs and outputs is selected from the training set and the network calculates the output based on the inputs. This output is subtracted from the actual output to find the output-layer error. The error is back-propagated through the network, and the weights are suitably adjusted. This process continues for the number of prescribed sweeps or until a pre-specified error tolerance is reached. The mean square error over the training samples is the typical objective function to be minimized.

Traditionally, a fixed step size, α is used throughout the training process resulting in a form of stochastic approximation. Maier et al.,(2010), as cited Kasiviswanathan et al.,(2013), observe that in order to optimise the performance of feed-forward networks trained with the back-propagation algorithm, it is essential to have a good understanding of the impact that step size has on training. The training becomes slow with too little steps in weight space and the network falls into oscillatory traps when the steps are too large. Traditionally, trial and error approach has been used to optimize step size.

Despite the diversity in neural network learning algorithms, Besaw et al, (2010) observe that even with the limitations of back-propagation algorithms of more iteration and not always converging, it is by far the most common accounting for more than 90 percent of the published applications. Joorabchi et al., (2007) note that back-propagation is the most used method of artificial neural network training algorithm in engineering problems.

Levenberg-Marquardt Training algorithm

In light of the weakness of the conventional back-propagation algorithm, a number of variations or modifications of back-propagation such as second-order methods have been proposed. Among them Levenberg-Marquardt methods are more efficient nonlinear optimization methods and are used in most optimization packages (Zhang et al., 1998).

Their faster convergence, robustness, and the ability to find good local minima make them attractive in ANN training. In their recent study, Cigizoglu and Kisi (2007) have shown that the Levenberg–Marquardt (LM) algorithm decreases the training time significantly.

Resilient back-propagation algorithm

The purpose of the resilient back-propagation training algorithm is to eliminate harmful effects of the magnitudes of partial derivatives. Only the sign of the derivative is used to determine the direction of the weight update and the magnitude of the derivative has no effect on the weight update. The size of the weight change is determined by a separate update value. The update value for each weight and bias is increased by a factor delt_inc whenever the derivative of the performance function with respect to that weight has the same sign for two successive iterations. The update value is decreased by a factor delt_dec whenever the derivative with respect to that weight changes sign from the previous iteration. If the derivative is zero, the update value remains the same. Whenever the weights are oscillating, the weight change is reduced. If the weight continues to change in the same direction for several iterations, the magnitude of the weight change increases. It is not very sensitive to the settings of the training parameters. A study by Talaee, (2012) on the performance of the training algorithms concluded that Levenberg–Marquardt (LM) algorithm is more efficient than resilient back-propagation training algorithm which is generally much faster than the standard steepest descent algorithm.

2.3.4 Advantages and disadvantages of Artificial Neural Networks

Neural networks have already achieved significant progress and success in forecasting and prediction applications. The approach has several advantages over other traditional data driven approaches. Particularly among them are the fact that they can be used to model non-linear processes and that they do not require 'a priori' understanding of the detailed mechanics of the processes involved. Because of the parallel nature of the data processing, the approach is also quite robust and insensitive to noise present in the data and when an element of the neural network fails, it can continue without any problem due to their highly parallel nature (Jayawardena, 2009).

The ANN learns how to relate the inputs to the outputs without being given any explicit equations the only real requirements for the ANN model are for sufficient data and the specification of appropriate neural network parameters values to be used. ANNs have relatively low computational demands and can easily be integrated with other techniques. The ANN approach offers a viable alternative which does not require a high level of expertise. Unlike most traditional statistical approaches, ANNs are able to model both linear and nonlinear systems without the need to make any assumptions (Kisi 2005).

With several achievements of ANNs in forecasting and prediction applications, it is however, important to point out that they also have limitations. One disadvantage of ANNs is that the optimal form or value of most network design parameters can differ for each application and cannot be theoretically defined in general. However, these values are commonly approximated using trial and error approaches (Kumar et al., 2007 and Talaee, 2012). The other disadvantage of neural networks is that they require training to operate which may require much processing time for large neural networks. Using neural networks requires thorough understanding of the data, prudent design of modeling strategy, and careful consideration of modeling issues. Neural network design and architecture selection are important yet difficult tasks. Not only are there many ways to build a neural network model and a large number of choices to be made during the model building and selection process, but also numerous parameters and issues have to be estimated and experimented before a satisfactory model may emerge. Even though several methods have been suggested by authors, the model architecture of an ANN is generally selected through a trial and error procedure as currently there is no established methodology for selecting the appropriate network architecture prior to training (Talaee, 2012)

2.3.5 Artificial Neural Network applications in water resources engineering

Neural networks find their field of excellence when they are applied to model the real world or natural environment (Johannet et al., 2007). Artificial neural networks have been applied to solve real life problems since Rumelhart et al., (1986) introduced back-propagation algorithm. Since then much research has been carried out to solve a wide range of problems from business to engineering. Artificial neural networks have been used to solve complex engineering problems and in this section studies related to application of neural network to water resources engineering is summarized.

Wang et al., (2005) applied artificial neural network trained with back-propagation and sigmoid activation function in both hidden and output node in predicting long-term flow discharges in Manwan based on historical records from 2001 to 2003 and from 1953 to 2003 to predict daily and monthly flow respectively. Their results and conclusion indicate that artificial neural network model can give good prediction performance. Tombul et al., (2006) carried out a study on modeling of rainfall-runoff relationship at the semi-arid small catchments using artificial neural networks and also compared the results with classical regression model and concluded that artificial neural network could provide a very useful and accurate tool to solve problems in water resources studies and management.

Talaee (2012), applied Multilayer Perceptron (MLP) networks optimized with three training algorithms where antecedent streamflow with 1, 2, 3 and 4 day time lag constituted the input vector and concluded that proper selection of training algorithms could help improve the accuracy of the artificial neural network models for streamflow forecasting. Yanik et al., (2011) found out that ANN models are capable of good estimations of inflow into a dam from a catchment when they applied variations of the back-propagation training algorithm such as Gradient Descent with Adaptive Learning Rate (GDX) and Levenberg-Marquardt (LM) with sigmoid activation function and a training set consisting of data between 1999 and 2009 years for Inflow Estimation of Yuvacık Dam Catchment Area located at the northwest of Turkey. The results obtained were satisfactory with slight over-estimation of extreme conditions.

Besaw et al., (2010) developed and tested two artificial neural networks (ANNs) to forecast streamflow in ungauged basins using publically available climate and US Geological Survey streamflow records from sub-basins in Northern Vermont. The study applied a recurrent neural network and time-lagged input data. From the study it was observed that the ANNs, trained on a climate-discharge record from one basin, proved capable of predicting streamflow in a nearby basin as accurately as in the basin on which they were trained. A study comparing the performance of Artificial Neural Networks and Auto-Regressive Models in forecasting daily river flow for the Filyos Stream in Turkey by Kisi (2005), Altunkaynak (2007) in the study of Forecasting surface water level fluctuations of lake van by artificial neural networks and Pierini et al.,(2012) applied both artificial neural network and Auto-Regressive Models in Prediction of water flows in Colorado River, Argentina and all concluded that with the same data set as input, artificial neural network produce better results than Auto-Regressive Models.

The other examples of the neural networks to solve water resources related problems were studied by Ishak et al.,(2011) and Wang et al., (2005) applied neural network to support reservoir water release decision and results shown that performance of neural network for both forecasting and decision models are acceptably good but can be improved .

2.4 Research Conceptual Framework

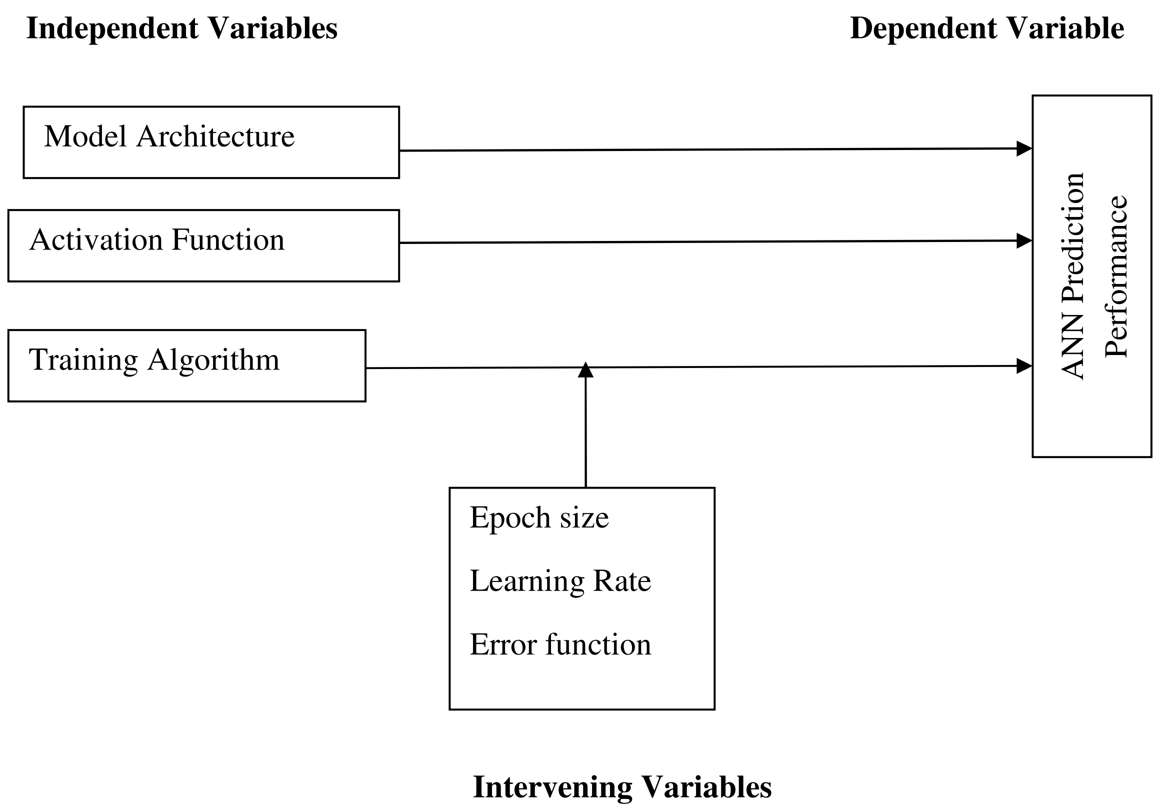

Figure 2.3: Research conceptual framework (Source; Author 2014)

According to the diagram, ANN prediction performance is affected by the network architecture which is the number of hidden layers, number of nodes in the hidden layer and the number of connection, optimization algorithm and the activation function (Gomes et al., 2011). The intervening variables which for this case are training parameters have been observed by Zhang et al., (1998) to have major contributions to neural network performance. Epoch size determines the number of training samples presented to the network between weight updates, learning rate determines the magnitude of weight changes and the error function parameter E is the function that is minimized during training.

2.5 Description of study area

Sondu-Miriu River watershed (Figure 2.4) the study area of this project is located in Kisumu County western Kenya. It drains a total area of 3,508 km^2 in the western part of Kenya. The river originates from the western slopes of the Mau Escarpment and inflows through a narrow Gorge, penetrating the Odino Falls before entering the flood plains of Nyakwere where it drains into the Winam Gulf of the Lake Victoria.The flow measuring sites (gauging stations) along the river are operated by Water Resources Management Authority. At the study site, the gauging stations are operated by both Water Resources Management Authority and the Kenya Energy Generation Company (KenGen).

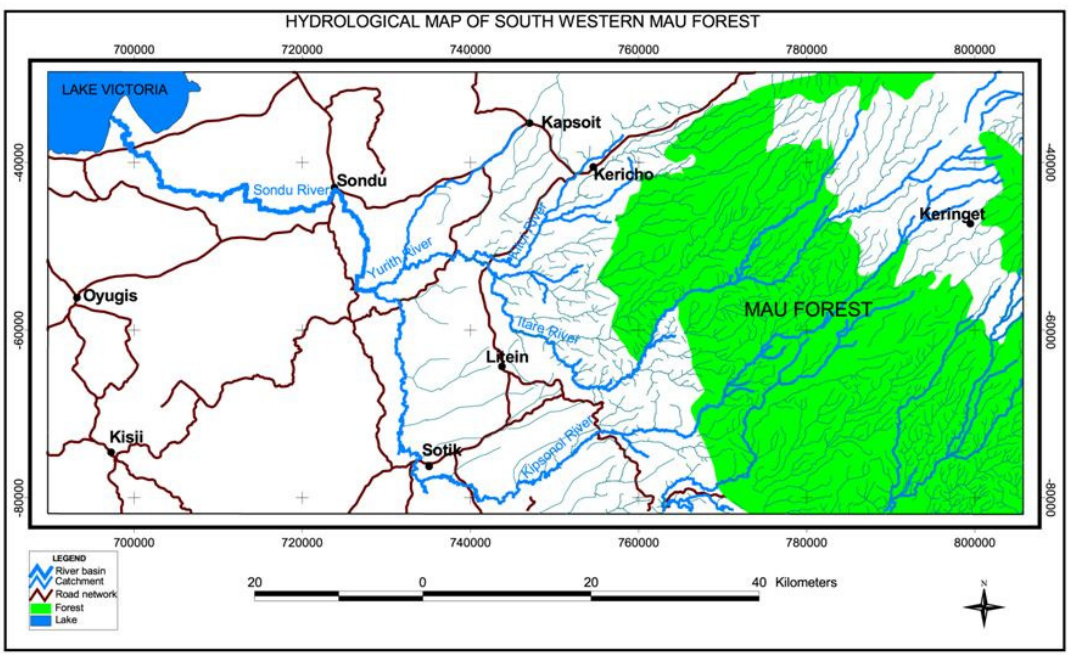

Figure 2.4: Sondu-Miriu Catchment (Source; Otuoma et al., 2012)

The daily and monthly river flow data at one of the stations operated by Water Resources Management Authority and Kenya Energy Generation Company (KenGen) at the site where the hydropower project is located is available from 1950 – 2009. The data provided has been fitted using statistical methods to fill the gaps.

CHAPTER THREE: RESEARCH METHODOLOGY

3.1 Introduction

This chapter illustrates ways through which solution to the problem under study was achieved. It gives clear methods applied throughout this study which was divided into;

i. Research Design
ii. Model Design

3.2 Research Design

The approach for this study was quantitative using correlational research method. The study used case study method to establish how artificial neural network model can improve river discharge prediction at Sondu-Miriu hydropower plant. The study was based on the data obtained from a river gauging station by WRMA along Sondu-Miriu River next to the intake point. Quantitative design uses numerical methods and statistical tools for data collection and analysis. Correlational research method attempts to determine the extent of a relationship between two or more variables using statistical methods. Relationships between and among a number of facts are sought and interpreted (Carrie 2007). It recognizes trends and patterns in data which in this study involved historical river flow data but it does not go so far in its analysis to prove causes for these observed patterns. Variables are not manipulated; they are only identified and are studied as they occur in a natural setting.

Case study method was chosen because it allowed for in-depth investigation into the actual river flow prediction procedure and a model developed to be tested for Sondu-Miriu station before implementing the idea to other stations.

3.3 Artificial Neural Network Model

In this study, a multilayer feed forward neural network was adopted. Neural network architecture defines its structure including number of hidden layers, number of hidden nodes and number of output nodes. The literature reviewed observed that one hidden layer with a sufficient number of hidden neurons is capable of approximating any continuous function (Kumar, 2007). In this study one hidden layer was adopted and in order to determine the optimum number of hidden nodes in the hidden layer, several approaches were considered including a rough estimate proposed by Masters (1993), which states that for a three layer network with n input and m output neurons the hidden layer would have sqrt($n*m$) neurons, Zhang et al.,(1998) suggested an approach that the network with the number of hidden nodes being equal to the number of input nodes have better forecasting results. Due to varying suggestions on how to determine the neural network structure, trial and error was used taking into considerations the suggestions by different authors to determine optimum neural network structures.

3.3.1 Artificial Neural Network Model Design

Development of the neural network model to predict average monthly flow for Sondu-Miriu River followed the following steps summarized in the flowchart figure3.1.

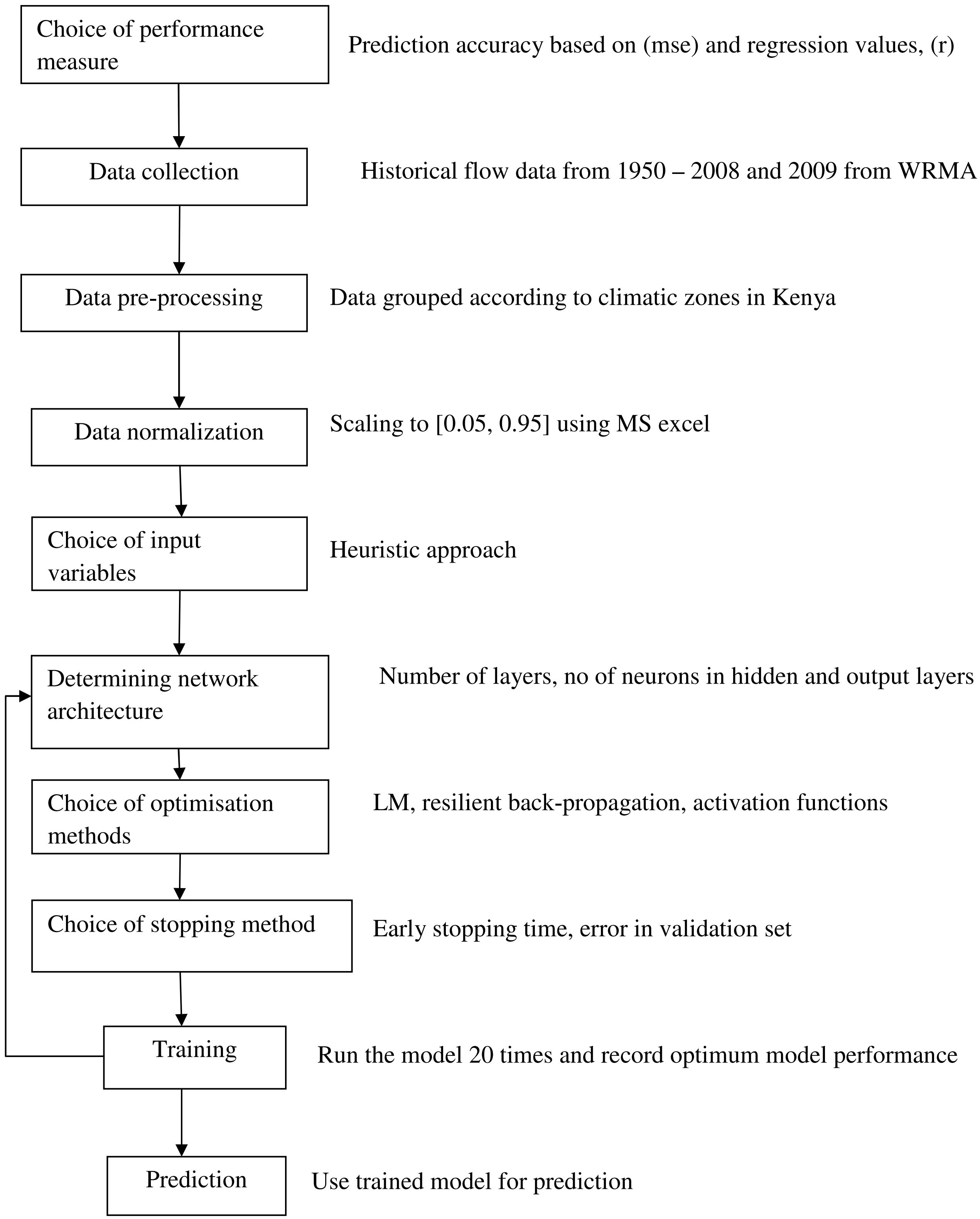

Figure 3.1: Summary of Neural Network development steps

3.3.2 Data Description and Collection

In this study a neural network model was developed to predict river flow discharge. The model is based on the data of 58 years of average monthly discharge measured from 1950 to the end of 2008 at a gauging station located along Sondu-Miriu River by Water Resources Management Authority (WRMA) which is the organization responsible for river flow measurements in Kenya.

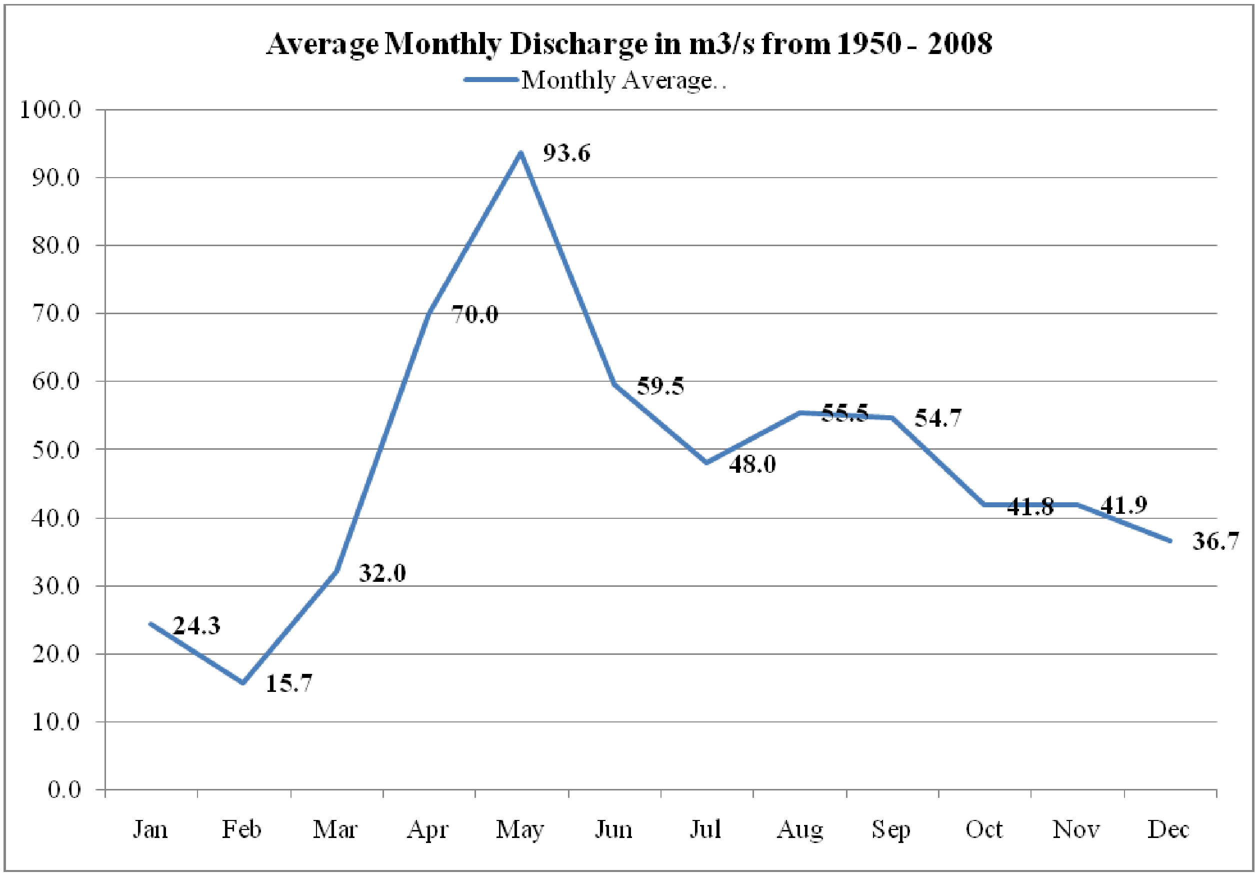

Figure 3.2: Sondu-Miriu Average monthly discharge 1950-2008

Kisi et al., 2007 and Pierini et al., 2012 posit that the more the training data for artificial neural network, the more accurate the results because many training patterns can be obtained. Therefore in this study, to achieve more training data pairs, average monthly flow data were combined for three months based on the Kenyan climatic zones as classified by Kenya Meteorological Department which are;

January to March as warm and dry season, April to June as long rainy season, July to September as cool dry season and October to December as short rainy season.

Figure 3.2 illustrates the average monthly flow regime of Sondu-Miriu River for a period of 58 years which tends to follow the climatic zones classification by Kenya Meteorological Department. Sivapragasam et al., (2014), justify the grouping by noting that climatic and meteorological conditions during a given month for a large catchment may be the same.

3.3.3 Data Preparation

The aim of developing the neural network model was to predict the Sondu-Miriu River flow discharge in the next one month based on the previous six years data of the same month. The available data for 58 years was organized into months from January to December average flow. Data for the months within the same climatic zone were then combined to obtain more training patterns which was 159 pairs for each set with six average monthly river flow discharge values as the input and the seventh month flow representing the target.

The models were developed based on grouped monthly average data according to Kenyan climatic zones as shown in figure3.2 which resulted to the following models;

Model1-January-February-March Dataset

Model2 -April-May-June Dataset

Model3 - July-August-September Dataset

Model4 - October-November-December Dataset

3.3.4 Data Normalization

Due to the nonlinearity of the input data, sigmoid function was used in the ANN model and for the avoidance of the network getting trapped in local minima, the time series hydrologic data obtained from WRMA was normalized onto the range [0.05, 0.95] before presenting them to the model for training and prediction. The normalization formula proposed by Smith M, (1993) and adopted by Talaee, (2012) was applied for data normalization.

$$Q_n = 0.05 + 0.95\,\frac{Q_r - Q_{min}}{Q_{max} - Q_{min}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots .. (1)$$

Where Q_n is normalized value,

Q_r is original value,

Q_{min} and Q_{max} are minimum and maximum values in the data set respectively.

Data normalization was done using Microsoft excel as a pre-processing step where Q_{max} for the entire data set was $312.5m^3/s$ and Q_{min} was $1.7m^3/s$ respectively. The output from the model was rescaled for ease of interpretation by reorganizing formula (1) to obtain formula;

$Q_{predicted} = (Y-0.05)*327.1579+1.7$ …………………………………………………………… (2)

Where Y is normalized prediction from the model.

3.3.5 Artificial Neural Network Architecture

In this research, procedures and libraries from Matlab Neural Network Toolbox were used to develop the neural network models. The research adopted a multilayer feed forward neural network with one hidden layer. The architecture of the model and other parameters remained constant during the simulation period while learning algorithms and activation functions in both hidden and output layers were varied to obtain an optimum model.

Different numbers of neurons between 2 to15 were tested to find the best structure. By comparing performance of developed ANN models, the optimum number of neurons in the hidden layer was obtained as 10 for the models. Sigmoid and linear activation functions were the only functions used in the study.

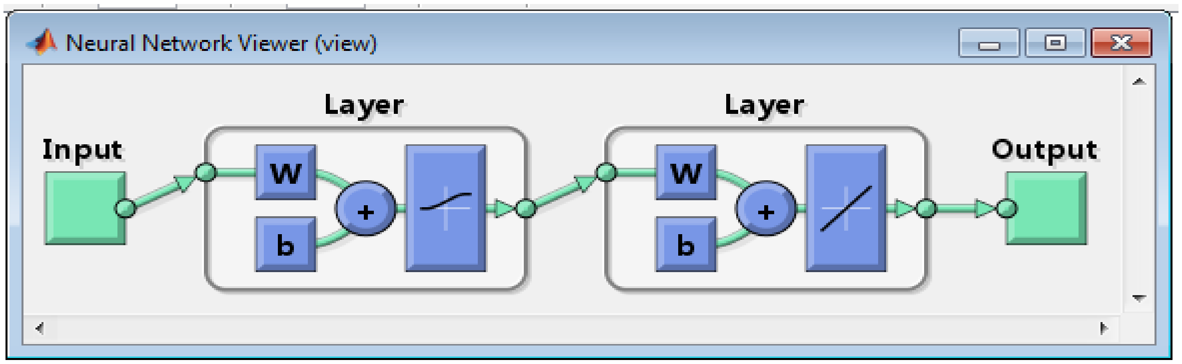

Figure 3.3: Developed NNA with sigmoid function at the hidden layer and linear at the output layer

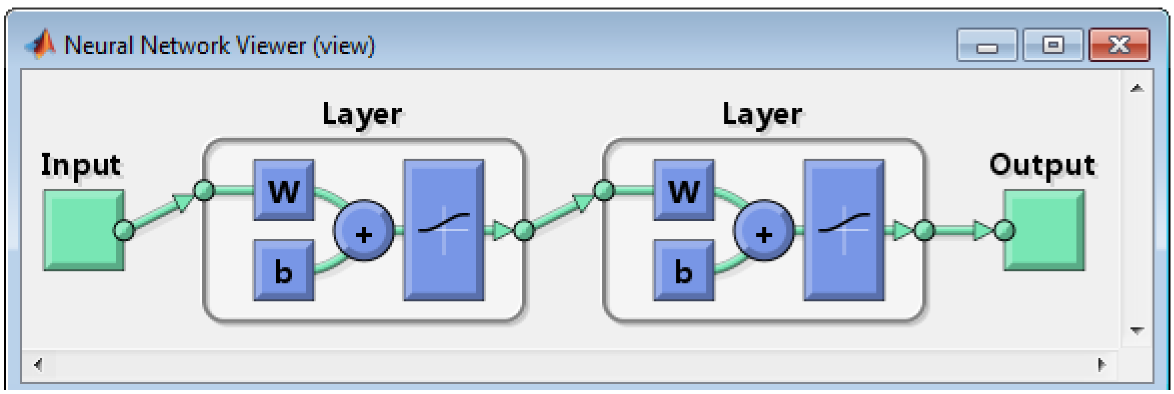

Figure 3.4: Developed NNA with sigmoid function at both hidden and output layers

3.3.6 Neural Network Training

The study applied supervised learning algorithm and the network was trained using the extensions of the most popularly used back-propagation training algorithm known as Levenberg–Marquardt (LM) and resilient back-propagation with both linear and sigmoid activation functions. In back-propagation algorithm, interconnection weights are adjusted according to the error convergence technique to obtain the required output for the given inputs. The default values of initial weights and biases were used in network training. Figure 3.5 demonstrates the flowchart for back-propagation algorithm.

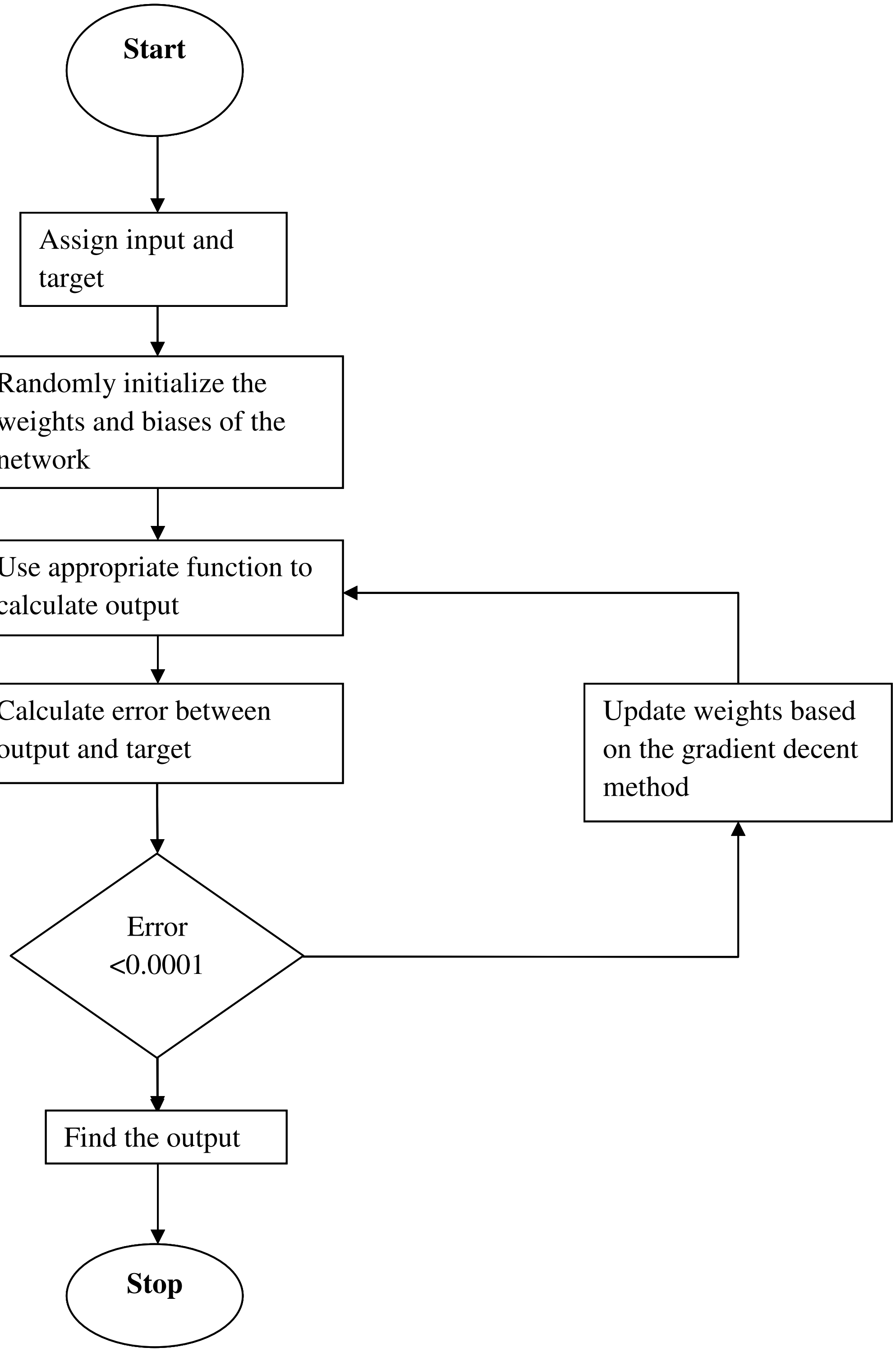

Figure 3.5: Flowchart for back-propagation training

Normalized input data and output data were imported to the Matlab Back-propagation Algorithm (BPA) in the form of input matrix X (6*159) and target matrix T (1*159) respectively. The algorithm takes only 60 percent of the input data for training. So out of 159 samples only 95 were taken for training and these were selected randomly from the set of data. For every attempt of training the data, the algorithm selected the training sample randomly from the whole set and not a fixed set of data and so every time the network was trained, different values of mean square error (mse) and correlation coefficients (r) values were obtained depending upon which 60 percent of the input data was chosen for training. The remaining 32 samples were kept for validation and 32 samples for testing.

The configuration that gave the minimum mean square error (mse) and highest correlation coefficients (R) at the test data was chosen for the model. For each data set, the network was trained in batch mode (offline learning) to minimize the (mse) at the output layer. Multiple test runs were done on the model structures under different training algorithms through the two activation functions to determine the combination leading to best performance.

3.4 Performance Evaluation and Validation

Each model was run 20 times to obtain the best (mse) and (r) values under different training algorithms and different activation functions at the hidden and output layers while other parameters remained constant. The performance of the models under different algorithms and different activation functions were quantified by statistical measures addressing the magnitude of the variables in terms of mean square error (mse) and regression (r) between the predicted and observed average monthly flow values. Mean square error (mse) measures how closely predictions match observations and according to Besaw et al., (2010) and Karim (2009), ANN response is precise when mse is close to zero. Matlab Neural toolbox displays graphically both values after successful run.

CHAPTER FOUR: RESULTS ANALYSIS AND INTERPRETATION

4.1 Introduction

This chapter focuses on the results and analysis obtained from running the models. The best values of the parameters obtained during the training stage of the models were used to evaluate the competence of the network. Performance evaluation was determined by high value of correlation coefficient (r) in test data and smaller mean square error (mse). According to (Hung et al., 2002), the network architecture which is a combination of weights, biases, number of hidden neurons and training algorithm including activation function which gives the smallest validation set error is chosen and the network`s performance is evaluated by using the (r) value of test data set.

4.2 Model Results

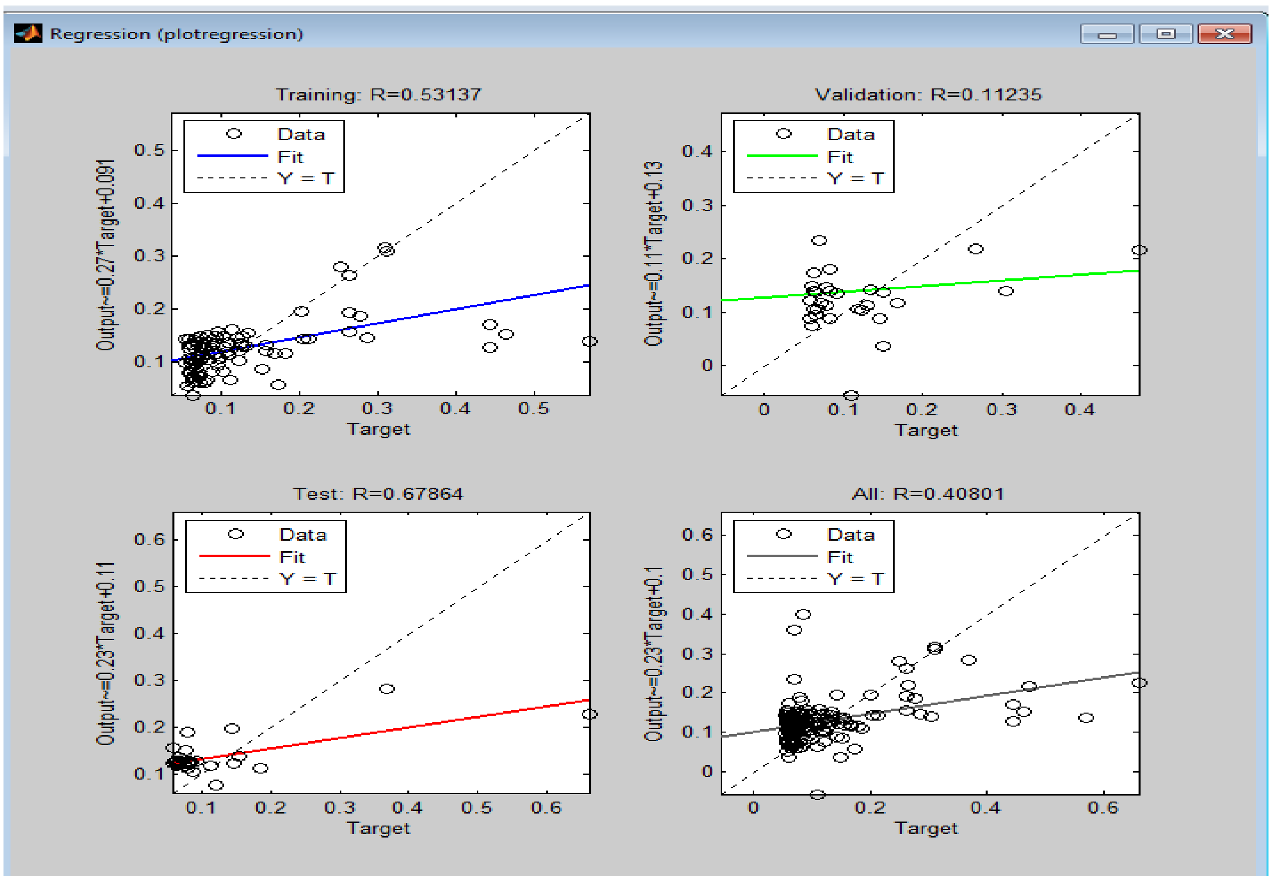

Figure 4.1: ANN training result showing best correlation value on test data.

The figure illustrates the determination of the best result obtained during training which is dictated by r value for test data set. According to Sivapragasam et al., (2014), in Matlab, a high value of correlation coefficients (r) in test set demonstrates good prediction. Regression (r) value ranges from 0 to 1 and describes the amount of observed variance explained by the model. If r is equal to 1, it implies that the prediction replicates observation 100% of the time (Besaw et al., (2010).The model for figure 4.1 therefore gave 67% prediction accuracy based on the r value for the test data set.

Each model had 6 inputs and 1 target with 159 instances. The models were run 20 times and the best combinations of mean square error, regression on test data, epoch and iterations recorded. The results for each model configurations after training and testing for prediction were tabulated as shown in the following tables;

Table1: LM Algorithm with sigmoid function at hidden layer and linear at the output layer

Model	**R-Test Value**	**MSE**	**Epoch**	**Iterations**	**Prediction**	**Rescaled Value (m^3/s)**	**Actual Value (m^3/s)**	**% Prediction**
Model1	0.67864	0.0063819	2	8	0.1336	29.0560	31.5	92
Model2	0.46575	0.021808	1	7	0.3621	103.7943	130.5	80
Model3	0.37361	0.0034833	1	7	0.1978	50.0494	44.2	113
Model4	0.37145	0.012463	2	8	0.2099	54.0030	42.5	127

The best recorded prediction accuracy was from model1 with r test value of 0.67864, 2 epochs at 8 iterations and a percent accuracy of 92%. Model4 recorded the poorest performance with very low r test value of 0.37145 recorded from the 20 test runs. It had an over-prediction of 127%.

Table2: Levenberg-Marquardt Algorithm with sigmoid function at both hidden and output layers

Model	**R-Test Value**	**MSE**	**Epoch**	**Iterations**	**Prediction**	**Rescaled Value (m^3/s)**	**Actual Value (m^3/s)**	**% Prediction**
Model1	0.77985	0.06301	8	8	0.3568	102.0846	31.5	324
Model2	0.45492	0.069169	9	9	0.5273	157.8411	130.5	121
Model3	0.2195	0.013549	9	9	0.2802	77.0209	44.2	174
Model4	0.36671	0.089246	9	9	0.4432	130.3394	42.5	307

Model1 gave the poorest performance with high values of r, epochs and iterations. Irrespective of the number of test runs, the prediction remained the same.

Table3: RBP Algorithm with sigmoid function at hidden layer and linear at the output layer

Model	**R-Test Value**	**MSE**	**Epoch**	**Iterations**	**Prediction**	**Rescaled Value (m^3/s)**	**Actual Value (m^3/s)**	**% Prediction**
Model1	0.61169	0.0059245	9	15	0.1162	23.3609	31.5	74
Model2	0.50727	0.052902	6	11	0.3526	100.7033	130.5	77
Model3	0.38363	0.0091881	12	18	0.1963	49.5696	44.2	112
Model4	0.41837	0.039056	9	15	0.2082	53.4641	42.5	126

Table3 displayed similar characteristics with table1 however, the epochs and iterations are higher. The predictions are not as accurate as in table1.

Table4: RBP Algorithm with sigmoid function at both hidden and output layers

Model	**R-Test Value**	**MSE**	**Epoch**	**Iterations**	**Prediction**	**Rescaled Value (m^3/s)**	**Actual Value (m^3/s)**	**% Prediction**
Model1	0.66762	0.065591	10	10	0.3568	102.0846	31.5	324
Model2	0.47185	0.079785	10	10	0.5273	157.8411	130.5	121
Model3	0.2381	0.011618	11	11	0.2802	77.0209	44.2	174
Model4	0.3159	0.084209	11	11	0.4432	130.3394	42.5	307

Table4 has similar results with table2 except for r test values, epochs and iterations. More test runs do not bring any change to the prediction.

4.3 Discussion and interpretation of Results

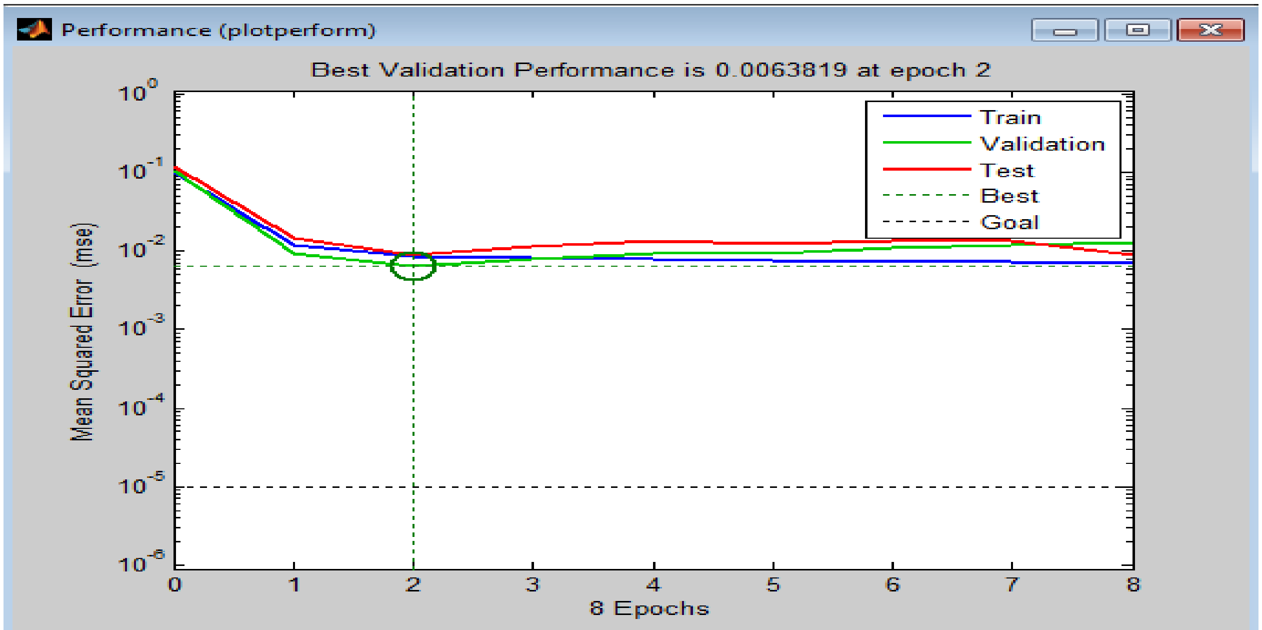

Figure 4.2: Graphical display of an Artificial Neural Network training result

Training result of the network shown in Figure 4.2 illustrates that when the number of epochs increases, the errors of all three sets decline. At the beginning of training, the decrease in squared error is very sharp and then decreases gradually. According to Demuth et al., (2002), training is stopped when error for validation set starts increasing which in this case, training is stopped after 2 epochs. If the model is constructed successfully, the test set and validation set error should show similar characteristics. Figure 4.2 shows that they both follow the same pattern which proves that the model is reliable. It does not appear that any over-fitting has occurred, because neither testing nor validation error increased before iteration 8. The performance of the model is measured with the mean squared error which reduces to 0.0063819 after 2 epochs.

The results from tables 1 and 2 show that Levenberg-Marquardt algorithm with sigmoid function at hidden layer and linear function at the output layer have faster convergence as shown by the number of epochs and iterations for the best runs and better prediction accuracy but when the same conditions are maintained and sigmoid function used in both layers as shown in table2, convergence becomes slower as shown by the number of epochs and iterations for the best runs and poor prediction accuracy. Training parameters under this configuration do not affect the prediction which therefore supports the suggestions that sigmoid activation functions in the output layer is more suitable for classification problems but not prediction problems.

ANNs are suitable in analyzing non-linear situations where data does not display linear relationships. According to figure3.2 on page 28 of average monthly discharge for 1950-2008, models 3 and 4 data tends to be linear which has led to low r test set values and poor prediction accuracy with the worst being model4 which also appear to be more linear.

Prediction results from table2 and table4 are similar while different training algorithms were used. The results show that when sigmoid function is used in both hidden and output layers, the other training parameters do not affect the output which is very inaccurate in prediction applications. The output only depends on the data set.

Generally, all the models evaluated under Levenberg-Marquardt algorithm with sigmoid function at hidden layer and linear function at the output layer configuration either over-predicted or under-predicted within acceptable range with low errors at low epochs and at less iterations while resilient back propagation algorithm produced less accurate result at higher epochs and more iterations.

4.4 Research Assessment

To aid in the assessment of the value and contribution of the study project, a framework developed by Whetten, D.A. (1989) was adopted. The framework outlines seven key questions that must be answered in order to measure whether or not a study has made significant contributions to the subject area. The final output of this study is therefore evaluated against this framework.

a) What is new?

Currently there is debate among researchers on the suitability of using sigmoid function in both hidden and output layers in ANN prediction applications but still there is no consensus. By establishing and demonstrating performance of both sigmoid and linear functions at both hidden and output layers for a prediction application, the study adds to the view that linear activation function in ANN is most suitable for an output layer in prediction applications.

b) So what? How will the study change river flow prediction?

The findings of the study show that ANN trained with Levenberg-Marquardt training algorithm and sigmoid activation function at the hidden layer and linear activation function at the output layer give accurate predictions to river flow. Considering the time and resources involved in other river flow predictions methods, the study offers an option for cheaper and convenient method for practitioners' especially for Sondu-Miriu River system.

c) Are the underlying logic and supportive evidence compelling?

The study has its foundation on concrete theories established and proven by previous studies. The identification of the independent variables in the conceptual model for this study was based on the fact that several authors have studied their effect on ANN performance and have suggested further research studies to their improvements. Research questions were also formulated based on solid theoretical foundation of previously conducted and proven study findings.

d) How thorough was the study

The study first established that ANN has not been applied in the area under study. Historical time series flow data for the river was obtained from WRMA then preprocessed to a required mode. A model was developed in Matlab to test the theories from the literature on factors affecting ANN performance. Upon establishing the gaps from the literature review, a strategy was put in place to help in bridging the gap which was followed systematically and thoroughly to achieve the end results.

e) Is the thesis well written? Does it flow logically?

The study starts by giving background information and problem statement, objectives for the study are well stated. To check with what other researchers have studied on this study area, related literature were reviewed and clear methodology followed which led to the attainment of the study objectives. Results from the study, conclusion and recommendations are then given hence the project write-up is considered well structured.

f) Why now? Is it of interest to the people?

Reliable stream flow prediction model is important in assisting water resources managers and engineers in allocation of water to competing users like hydropower, irrigation and domestic. Hydrologic component requires both short term and long term forecasts of stream flow events in order to optimize the system or to plan for future expansion or reduction (Kisi 2005). There is need for improvement on the performance of such prediction models especially to areas where such models have never been applied like in Sondu-Miriu River system therefore this research is of more interest now as demand on water systems grow.

g) Who else including academic researchers are interested in this study?

Researchers on ANN applications will be interested in ways to improve performance of ANN in prediction applications by varying the model parameters.

To the practitioners, the tool can be used in predicting one month ahead average monthly flow by providing past successive yearly flow of the same month. This will lead to efficient operations and planning. The tool can also be applied as an early warning system in flood forecasting hence assisting in saving the lives in flood prone areas.

CHAPTER FIVE: CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

The chapter evaluates whether the objectives of the study have been met, assesses the value of the research, shows the limitations of the study and finally gives conclusions and recommendations.

5.2 Evaluation of Research Objectives

Objective one: *To pre-process historical river flow data into a form that is suitable for training artificial neural networks.*

Because sigmoid function was used in network to combine nearly linear, curvilinear and constant behavior, its squashing property which takes any real valued input and returns an output bounded between (0,1) according to Pierini et al.,(2012) and in order to avoid the network from getting trapped in local minima, the time series hydrologic input data was normalized onto the range [0.05, 0.95] using Smith M, (1993) normalization formula;

$$Q_n = 0.05 + 0.95\,\frac{Q_r - Q_{min}}{Q_{max} - Q_{min}}$$

before presenting them to the model for training and prediction. This was done in Ms Excel and a sample of pre-processed data is shown in appendix B. This is the indicator that the objective was met.

Objective two: *To develop a neural network model*

To address this objective, procedures and libraries from Matlab Neural Network Toolbox were used to develop the neural network models. Multilayer feed forward neural network with one hidden layer, 10 neurons and one output node was developed from Matlab code as shown in appendix A hence meeting the stated objective.

Objective three: *To verify and validate the model performance*

After developing and running the models, verification and validation was done by comparing the results obtained from the model for a given year that is 2009 with the actual measured values. This formed the basis of determining the models' prediction accuracy hence the objective was met.

Objective four: *To compare the models performance in predicting Sondu-Miriu River discharge under different algorithms and activation functions.*

To address this objective, the models developed were run under different configurations with the two algorithms and the two activation functions. The results shown in tables1,2,3 and 4 confirmed (Kisi et al., 2007, Yanik et al., 2011 and Talaee, 2012) observations that Levenberg-Marquardt training algorithm converges faster and give more accurate results and sigmoid function being more suitable when used only in the hidden layer for a prediction application. The objective was therefore met.

5.3 Limitations

Artificial neural network requires a lot of training data which could not be obtained from WRMA which led to combining of data for different months with justification from Kenya Meteorological Department. Since the data gaps had been statistically filled by WRMA, this may affect the reliability of the developed prediction model.

The study sort to establish the effect of other training parameters like learning rate and error function on the performance of an ANN. This could not be done since it would have introduced many varying parameters which could not be achieved during the period of study.

Despite these challenges and limitations, the main goal of the study was still met and future work on this area would only improve the model.

5.4 Conclusion

The following conclusions were made from the study;

- Under similar training data set, artificial neural network trained with both Levenberg-Marquardt and resilient back-propagation algorithms with sigmoid activation function at hidden layer and linear activation function at the output layer are all capable of producing good results. However, Levenberg-Marquardt performs better and has fast convergence than resilient back-propagation.
- It is noted that application of sigmoid activation function at both hidden and output layer with either training algorithm under same data set lead to similar result and other training parameters do not affect the model performance under such configuration.
- Sigmoid activation function in the output layer is not suitable for prediction applications.
- Finally, the study achieved its goal of developing a tool that could be used by stakeholders to conveniently predict average monthly river flow from previous six years flow of the same months without considering other meteorological parameters like

rainfall, evapo-transpiration affecting river flow. It can also be adopted for any river where there is measured historical flow discharge data.

5.5 Recommendations

Neural network model application in flow prediction has not been fully exploited by water resources experts in Kenya. The developed model has demonstrated its viability in the use for flow prediction especially in Sondu-Miriu River which currently lack formal flow prediction model for their operations.

The study therefore recommends the use of neural network model from Levenberg-Marquardt with sigmoid activation function at hidden layer and linear activation function at the output layer due to better results and faster convergence. The model can be improved further and customized to other rivers which require flow prediction information. A similar study should be carried out but with different combination of input variables which affect river flow such as rainfall, wind speed, humidity and soil properties of the catchment in order to verify the results.

REFERENCES

Abbas Majdi and Mohammad Rezaei (2013). **Prediction of unconfined compressive strength of rock surrounding a roadway using artificial neural network**: *Neural Comput & Applic 23:381–389.*

Abdusselam Altunkaynak (2007). **Forecasting surface water level fluctuations of Lake Van by artificial neural networks:** *Water Resources Management (2007) 21:399–408. DOI 10.1007/s11269-006-9022-6.*

Asaad Y. Shamseldin (2010). **Artificial neural network model for river flow forecasting in a developing country:** *Journal of Hydroinformatics 12.1.2010.*

A.W. Jayawardena (2009). **Riverflow Prediction with Artificial Neural Networks**; EANN 2009, CCIS 43, pp. 463–471, 2009.

Anil K. Jain, Jianchang Mao and K. Mohiuddin, (1996). **Artificial Neural Networks: A Tutorial.** *IEEE Computer, Vol.29, No.3. March, 1996. pp.31-44.*

Bahattin Yanik, Melih Inal, and Erhan Butun (2011). **Application of Artificial Neural Networks for Inflow Estimation of Yuvacik Dam Catchment Area:** International Symposium on DCAI, AISC 91, pp. 389.

Carrier, Christopher, Ajay Kalra, and Sajjad Ahmad (2013). **Using Paleo Reconstructions to Improve Streamflow Forecast Lead Time in the Western United States**. *Journal of the American Water Resources Association (JAWRA) 1-16. DOI: 10.1111/jawr.12088.*

Carrie Williams (2007). **Research Methods,** *Journal of Business & Economic Research – March 2007 Volume 5, Number 3.*

C. L. Wu, K. W. Chau, and Y. S. Li (2009), **Predicting monthly streamflow using data-driven models coupled with data pre-processing techniques,** *Water resources research, vol. 45, w08432, doi: 10.1029/2007wr006737, 2009*

C. Sivapragasam, K. Suganya, R. Selvi, S. Suji, S. Vanitha, S. Jeya Sudha, Nitin Muttil, M. Thamarai Selvi (2014). **Monthly flow forecast for Mississippi River basin using artificial neural networks** *Neural Comput & Applic (2014) 24:1785–1793.*

D. Nagesh Kumar, K. Srinivasa Raju and T. Sathish (2004): **River Flow Forecasting using Recurrent Neural Networks** *Water Resources Management 18: 143–161, 2004.*

Demuth, H., Beale, M., and Hagan, M., (2002), "**Neural Network Toolbox User's Guide for Use with MATLAB**", Version 4, The Mathworks, Natick, MA.

Elena Carla Carcano, Paolo Bartolini, Marco Muselli and Luigi Piroddi (2008).**Jordan Recurrent Neural Network Versus IHACRES In Modelling Daily Streamflows,** *Journal of Hydrology Volume 362*

Girish Kumar Jha.(2007). **Artificial Neural Networks**. *Indian Agricultural Research Institute, New Delhi, India, 2007.*

Guoqiang Zhang, B. Eddy Patuwo, Michael Y. Hu (1998). **Forecasting with artificial neural networks: The state of the art**. *International Journal of Forecasting 14 (1998) 35–62.*

Haykin,Simon (2008). **Neural N e t w o r k s and Learning Machines,** 3rd Edition. Prentice Hall.

http://www.reuters.com/article/2008/09/03/us-kenya-forest-idUSL246908720080903; *date accessed 17th March 2014.*

http://www.kengen.co.ke/index.php?page=business&subpage=hydro&id=10**;** *date accessed 17th March 2014.*

http://www.meteo.go.ke/; date accessed 22nd August 2014.

Johannet, P.A. Ayral and B. Vayssade (2007). **Modelling non Measurable Processes by Neural Networks: Forecasting Underground Flow Case Study of the *Cèze* Basin (*Gard* - *France*),** *Advances and Innovations in Systems, Computing Sciences and Software Engineering journal*, 53–58.

John Otuoma, David Langat, Johnson Maina, James Maina and Pius Mwanje (2012).**Effect of Watershed Degradation on Hydrological Functions in South West Mau;** *Conference paper (2012)*

Joorabchi, A. Zhang, H. And Blumenstein, M, (2007). **Application of artificial neural networks in flow discharge prediction for the Fitzroy River**, *Australia. Journal of Coastal Research, SI 50 (Proceedings of the 9th International Coastal Symposium), 287 – 291. Gold Coast, Australia, ISSN 0749.0208.*

Jorge O. Pierini, Eduardo A. Gómez & Luciano Telesca (2012). **Prediction of water flows in Colorado River, Argentina***; Latin American Journal of Aquatic Research volume 40 issue4: 872-880, 2012.*

J. Wang, X. Liao, and Z. Yi (Eds.) (2005). **Long-Term Prediction of Discharges in Manwan Reservoir Using Artificial Neural Network Models;** LNCS 3498, pp. 1040–1045, 2005.

K. S. Kasiviswanathan and K. P. Sudheer (2013). **Quantification of the predictive uncertainty of artificial neural network based river flow forecast models.** *Stoch Environ Res Risk Assess 27:137–146 DOI 10.1007/s00477-012-0600-2.*

Karim Solaimani (2009): **Rainfall-runoff Prediction Based on Artificial Neural Network (A Case Study: Jarahi Watershed)** *American-Eurasian J. Agric. & Environ. Sci., 5 (6): 856-865, 2009.*

Karunanithi, N., Grenney, W., Whitley, D., and Bovee, K. (1994). **Neural Networks for River Flow Prediction**: J.Comput. Civ. Eng., 8(2), 201–220.

Kumar Abhishek, Abhay Kumar, Rajeev Ranjan and Sarthak Kumar (2012): **A Rainfall Prediction Model using Artificial Neural Network;** *2012 IEEE Control and System Graduate Research Colloquium (ICSGRC 2012).*

Lance E. Besaw , Donna M. Rizzo , Paul R. Bierman , William R. Hackett (2010). **Advances in ungauged streamflow prediction using artificial neural networks;** *Journal of Hydrology 386 (2010) 27–37.*

Li Xiaofeng and Yang Chunshan (2014). **Research and Application of Data Mining and NARX Neural Networks in Load Forecasting** International Journal of Database Theory and Application Vol.7, No.2, pp.13-24.

Lin, G.F., G.R. Chen, M.C. Wu, and Y.C. Chou, (2009). **Effective Forecasting of Hourly Typhoon Rainfall Using Support Vector Machines.** *Water Resources Research 45:W08440, doi: 10.1029/ 2009WR007911.*

Muhammad Aqil, Ichiro Kita, Akira Yano and Soichi Nishiyama (2007).**Neural Networks for Real Time Catchment Flow Modeling and Prediction;** *Water Resour Manage (2007) 21:1781–1796 DOI 10.1007/s11269-006-9127-y*

Mustafa Tombul and Ersin Ogul (2006), **Intelligent Control and Automation: Modeling of Rainfall-Runoff Relationship at the Semi-arid Small Catchments Using Artificial Neural Networks,** *International Conference on Intelligent Computing, ICIC 2006 Kunming, China, August 16–19, 2006*

Ozgur kisi and H. Kerem Cigizoglu (2007). **Comparison of different ANN techniques in river flow prediction;***Civil Engineering and Environmental Systems Vol. 24, No. 3, September 2007, 211–231.*

Ozgur kisi (2005). **Daily River Flow Forecasting Using Artificial Neural Networks and Auto-Regressive Models:** *Turkish J. Eng. Env. Sci.29 (2005), 9 - 20.*

P. Hosseinzadeh Talaee (2012). **Multilayer perceptron with different training algorithms for streamflow forecasting**. Neural Comput & Applic Journal Volume 5, 2012.

Ramapulana Nkoana, (2011). **Artificial Neural Network Modelling of Flood Prediction and Early Warning**. Masters thesis, Faculty of Natural and Agricultural Science DIMTEC, University of the Free State Bloemfontein.

Rumelhart, D.E., Hinton, G.E., and Williams, R.J., (1986). **Learning representations by Back-propagating errors.** *Nature 323, 533–536.*

Smith M (1993) **Neural networks for statistical modeling**. Wiley, New York

Timothy Masters (1993). **Practical Neural Network Recipies in C++,** *Academic press New York.*

Wan Hussain Wan Ishak, Ku Ruhana Ku-Mahamud, and Norita Md Norwawi (2011). **Intelligent Decision Support Model Based on Neural Network to Support Reservoir Water Release Decision,** *ICSECS 2011, Part I, CCIS 179, pp. 365–379, 2011.*

Wang, Y.-C., Chen, S.-T., Yu, P.-S., Yang, T.-C., (2008). **Storm-Even Rainfall–Runoff Modelling Approach for Ungauged Sites in Taiwan**. *Hydrological Processes 22, 4322–4330.*

Whetten, D.A. (1989). **What constitutes a theoretical contribution?** *Academy of Management Review, 14(4), p. 490 – 495.*

Wong B.K., Bodnoyich T.A.,Selvi (1997). **Neural network applications in business: A review and analysis of the literature** (1988- 1995). *Decision Support Systems 1997:19:301-320.*

Gecynalda S. da S. Gomes, Leyla M. M. R. Lima and Teresa B. Ludermir (2011): **Comparison of new activation functions in neural network for forecasting financial time series;** *Neural Comput & Applic (2011) 20:417–439*

Hung, M. S., Shanker, M., and Hu, M. Y., (2002), "*Estimating Breast Cancer Risks Using Neural Networks*", **The Journal of the Operational Research Society**, Vol. 53, Issue 2, pp. 222-231.

Y. Tan, Y. Shi, and K.C. Tan (Eds.) (2010). **Air Quality Prediction in Yinchuan by Using Neural Networks**; *ICSI 2010 Part II, LNCS 6146, pp. 548–557, 2010.*

Yonas Kassa Demissie (2008), **Data-driven Models to Enhance Physically-based Groundwater Model Predictions,** *ProQuest.*

APPENDICES A: ***Matlab Neural Network Code to Predict Sondu-Miriu River Flow***

Model1

```
close all;
clear all;
clc;
X=xlsread ('Jan-Feb-Mar-Input.xls');%import normalized imput data to matlab
T=xlsread ('Jan-Feb-Mar-Target.xls');% import normalized target data to matlab
size(X) %check the size of the imported input matrix
size (T) %check the size of imported target matrix
%create a two layer feedforward network and set parameters
net = newff(X,T,10,{'logsig','purelin'},'trainlm');%creation of feed-forward network with one hidden layer 10 neurons in the hidden layer, sigmoid function at the hidden layer and linear function at the output layer with Levenberg-Marquardt as the training algorithm.
net.trainParam.epochs = 159; % training pattern
%net.divideParam=dividerand,divideblock,divideint,divideind
net.divideParam.trainRatio = 60/100;
net.divideParam.valRatio = 20/100;
net.divideParam.testRatio = 20/100;
net.trainParam.goal = 0.0001; % performance goal
net.trainParam.lr=0.5; %learning rate
net.trainParam.mu=0.5; % Momentum value
view (net) %view created network
net = train(net,X,T); %train the created network with set parameters

%the trained network is used to simulate or predict future output
predict=[0.0589;0.0831;0.1061;0.1561;0.1806;0.1348];%past six years Feb flow from 2003 to 2008
Y = sim (net, predict); %Prediction in normalized value
Q = {327.1579*(Y-0.05)+1.7,'m3/s'};%rescaled predicted value in m3/s
```

Model2

```
close all;
clear all;
clc;
X=xlsread ('Apr-May-Jun-Input.xls');%import normalized imput data to matlab
T=xlsread ('Apr-May-Jun-Target.xls');% import normalized target dat to matlab
size(X)%check the size of the imported input matrix
size(T)% check the size of imported target matrix
%Create a two layer feedforward network and set parameters
net=newff(X,T,10,{'logsig','purelin'},'trainrp');%creation of feed-forward network with one hidden layer 10 neurons in the hidden layer, sigmoid function at the hidden layer and linear function at the output layer with resilient back propagation as the training algorithm.
net.trainParam.epochs = 159; % training pattern
%net.divideParam=dividerand,divideblock,divideint,divideind
net.divideParam.trainRatio = 60/100;
net.divideParam.valRatio = 20/100;
net.divideParam.testRatio = 20/100;
net.trainParam.goal = 0.00001;
net.trainParam.lr = 0.5;
net.trainParam.mu = 0.5;
view (net) %view created network
net = train(net,X,T); %train the created network with the set parameters

%The trained network is used to simulate or predict future output
predict=[0.9837;0.5135;0.2768;0.4388;0.2918;0.4366];%past six years April flow from 2003 to 2008
Y = sim(net,predict); %Prediction in normalized value
Q = {327.1579*(Y-0.05)+1.7,'m3/s'}; %Rescaled predicted value in m3/s
```

Model3

```
close all;
clear all;
clc;
X=xlsread ('Jul-Aug-Sep-Input.xls');%import normalized imput data to matlab
T=xlsread  ('Jul-Aug-Sep-Target.xls');%  import  normalized  target  data  to matlab
size(X)%check the size of the imported input matrix
size(T)% check the size of imported target matrix
%Create a two layer feedforward network and set parameters
net=newff(X,T,10,{'logsig','purelin'},'trainlm');%creation of feed-forward network with one hidden layer 10 neurons in the hidden layer, sigmoid function at the hidden layer and linear function at the output layer with Levenberg-Marquardt as the training algorithm.net.trainParam.epochs = 159; % training pattern
%net.divideParam=dividerand,divideblock,divideint,divideind
net.divideParam.trainRatio = 60/100;
net.divideParam.valRatio = 20/100;
net.divideParam.testRatio = 20/100;
net.trainParam.goal = 0.00001;
net.trainParam.lr = 0.5;
net.trainParam.mu = 0.5;
view (net) %view created network
net = train(net,X,T); %train the network with the set parameters
%The trained network is used to simulate or predict future output
predict=[0.1537;0.2972;0.3242;0.2335;0.1769;0.1715];%past six years September flow from 2003 to 2008
Y = sim(net,predict); %Prediction in normalized value
Q = {327.1579*(Y-0.05)+1.7,'m3/s'}; %Rescaled predicted value in m3/s
```

Model4

```
close all;
clear all;
clc;
X=xlsread ('Oct-Nov-Dec-Input.xls');%import normalized imput data to matlab
T=xlsread  ('Oct-Nov-Dec-Target.xls');%  import  normalized  target  data  to
matlab
size(X)%check the size of the imported input matrix
size(T)% check the size of imported target matrix
%Create a two layer feedforward network and set parameters
net=newff(X,T,10,{'logsig','purelin'},'trainlm');%creation of feed-forward network
with one hidden layer 10 neurons in the hidden layer, sigmoid function at the hidden layer and
linear function at the output layer with Levenberg-Marquardt as the training algorithm.
net.trainParam.epochs = 159; % training pattern
%net.divideParam=dividerand,divideblock,divideint,divideind
net.divideParam.trainRatio = 60/100;
net.divideParam.valRatio = 20/100;
net.divideParam.testRatio = 20/100;
net.trainParam.goal = 0.00001;
net.trainParam.lr = 0.5;
net.trainParam.mu = 0.5;
view (net) %view created network
net = train(net,X,T); %train the created network with the set parameters
%The trained network is used to simulate or predict future output
predict=[0.0917;0.3150;0.0954;0.3104;0.1931;0.2338];%past  six  years  November
flow from 2003 to 2008
Y = sim(net,predict);%Prediction in normalized value
Q = {327.1579*(Y-0.05)+1.7,'m3/s'}; %Rescaled predicted value in m3/s
```

Model1b with 10 input variables

```
close all;
clear all;
clc;
X=xlsread ('Jan-Feb-Mar-Input.xls');%import normalized imput data to matlab
T=xlsread ('Jan-Feb-Mar-Target.xls');% import normalized target dat to matlab
size(X)%check the size of the imported input matrix
size(T)% check the size of imported target matrix
%Create a two layer feedforward network and set parameters
net=newff(X,T,10,{'logsig','logsig'},'trainlm');%creation of feed-forward network with one hidden layer 10 neurons in the hidden layer, sigmoid function at both hidden and output layer with Levenberg-Marquardt as the training algorithm.
net.trainParam.epochs = 159;
%net.divideParam=dividerand,divideblock,divideint,divideind
net.divideParam.trainRatio = 60/100;
net.divideParam.valRatio = 20/100;
net.divideParam.testRatio = 20/100;
net.trainParam.goal = 0.00001;
net.trainParam.lr = 0.5;
net.trainParam.mu = 0.5;
view (net); %view created network
net = train(net,X,T); %train the created network with the set parameters
%The trained network is used to simulate or predict future output
predict=[0.0617;0.0845;0.0643;0.0923;0.0589;0.0831;0.1061;0.1561;0.1806;0.134
8];%past 10 years Feb flow from 2003 to 2008
Y = sim(net,predict);
Q = {327.1579*(Y-0.05)+1.7,'m3/s'}; %Rescaled predicted value in m3/s
```

APPENDICES B: ***Sample Normalized Input and Target Data used in Training ANN***

Q_1	0.0510	0.0692	0.0514	0.0506	0.0514	0.0618	0.0523	0.0515	0.0523
Q_2	0.0692	0.0514	0.0506	0.0514	0.0618	0.0523	0.0515	0.0523	0.0527
Q_3	0.0514	0.0506	0.0514	0.0618	0.0523	0.0515	0.0523	0.0527	0.0504
Q_4	0.0506	0.0514	0.0618	0.0523	0.0515	0.0523	0.0527	0.0504	0.0735
Q_5	0.0514	0.0618	0.0523	0.0515	0.0523	0.0527	0.0504	0.0735	0.0599
Q_6	0.0618	0.0523	0.0515	0.0523	0.0527	0.0504	0.0735	0.0599	0.0605
T_1	0.0523	0.0515	0.0523	0.0527	0.0504	0.0735	0.0599	0.0605	0.0528