PHD THESIS

Study branch: COMPUTER SCIENCE

PhD Thesis:
Bio-Inspired Computing

Author: Hieu NGOC DUONG

Supervisor: Prof. RNDr. Vaclav SNASEL
Abstract

The main objective of this thesis is to investigate and tackle urgent practical problems involving Vietnamese agriculture. In Vietnam, agriculture is one of the major industries and contributes significantly to the national Gross Domestic Product (GDP). Thus it is necessary to drastically improve Vietnamese agriculture in many aspects, such as national policies, advanced agriculture technologies, applications of computer science and so on. Two problems that are investigated in this thesis are river runoff prediction and boiler efficiency optimization. Since neural networks have proven to be effective methods for modeling, characterizing and predicting several types of sophisticated data, they are chosen as key methods in this thesis.

For the first problem, we investigate some appropriate methods for predicting river runoff. The Srepok River is chosen as a case study. The task of prediction is divided into two cases: long-term and short-term prediction. To deal with the task of long-term prediction, three methods are utilized, such as recurrent fuzzy neural networks (RFNN), a hybrid of RFNN and genetic algorithms, and a physical-based method called SWAT. The experimental results show that the hybrid of RFNN and genetic algorithm is the most effective method.

To predict short-term river runoff, we propose a hybrid of chaotic expressions, RFNN and clustering algorithms consisting of K-means and DBSCAN. Chaotic expressions are used to transform the river runoff data into new data, called phase space, containing much temporal information. Whereas the combination of RFNN and clustering algorithms, which is based on the principle of mixture of experts, is trained and tested with the phase space. The experimental results are conducted with many combinations of RFNN, K-means, DBSCAN, Euclid distance, and Dynamic Time Warping (DTW). The experimental results indicate that the combination of RFNN, DBSCAN and DTW is superior to others.

For the second problem, RFNN and clustering algorithms are used to simulate boiler efficiency. The module of boiler simulation is an important component of a sophisticated soft sensor, namely BEO, which has been deployed at Phu My Fertilizer Plant since 2013. Then the boiler efficiency is forecasted multi-step-ahead and real-time. This task is tackled by using three methods including RFNN, a hybrid of RFNN and stochastic exploration, and RFNN improved by a reinforcement learning algorithm. The experimental results show that BEO is effective and can bring increased benefits to the plant.

Keywords: Neural networks, clustering algorithms, river runoff prediction, boiler efficiency optimization.
Acknowledgments

First, I would like to express my sincere gratitude to my two supervisors, Prof. Vaclav Snasel and Dr. Nguyen Thanh Hien, for their continuous support, patience, motivation, enthusiasm, and immense knowledge. Their guidance helped me in doing research, writing papers, and writing this thesis. I also would like to thank them for all the activities they organized, such as course-work, writing papers, which introduce me to the world of research.

In addition, I would like to thank the organizers, professors, and secretaries of the sandwich program for allowing me to participate to the international network. Especially, I would like to give special thanks to Dr. Phan Dao for his advice and support while I studied in Vietnam and Czech Republic.

My sincere thanks also go to my colleagues, Dr. Tran Van Hoai and Dr. Bui Ta Long, who are working in the Ho Chi Minh City University of Technology. They gave me much helpful advice on doing research during the past three years.

I would like thank my colleagues working at Phu My Fertilizer Plant, Petro Vietnam Fertilizer and Chemical Corporation, Petro Vietnam Group who provided insight and expertise that greatly assisted the thesis, especially Mr. Nguyen Minh Tam. I also thank the members of the EMSLab, Ho Chi Minh City University of Technology, especially Ms. Nguyen Thi Ngoc Quyen, for providing me with the data on which this thesis is based.

I would like to thank all members of the committees of my state examination and thesis, for their insightful comments, and hard questions.

Last but not least, I would like to thank my family: my parents, for their constant support in my life, and my wife for all her love and understanding.
Acronyms

ANN  Artificial Neural Network
MLP  MultiLayer Perceptron
RFNN Recurrent Fuzzy Neural Network
GA  Genetic Algorithm
SWAT Soil and Water Assessment Tool
RL Reinforcement Learning
ME  Mixture of Experts
MILE Mixture of Implicitly Localized Experts
MELE Mixture of Explicitly Localized Experts
MME Mixture of MLP-Experts
BP Back-Propagation
DBSCAN Density-Based Spatial Clustering of Applications with Noise
DTW Dynamic Time Warping
RFNN-KM-Euclid RFNN combining with K-means and Euclid distance
RFNN-KM-DTW RFNN combining with K-means and DTW
RFNN-DBSCAN-DTW RFNN combining with DBSCAN and DTW
MSA  Multi-Step-Ahead
SE-RFNN Stochastic Exploration and RFNN
RTRL-RFNN Real-Time Reinforcement Learning and RFNN
RMSE Root Mean Square Error
MARE Mean Absolute Relative Error
BEO Boiler Efficiency Optimization
Contents

1 Introduction .......................................................... 1
  1.1 Introduction ......................................................... 1
  1.2 Motivations .......................................................... 3
    1.2.1 Climate Change and Problems of River Runoff Prediction ... 3
    1.2.2 Boiler Efficiency Optimization ............................... 5
  1.3 Contributions ........................................................ 6
    1.3.1 River Runoff Prediction ......................................... 7
    1.3.2 Boiler Efficiency Optimization ............................... 7
  1.4 Organization of Thesis .............................................. 8

2 Background ............................................................ 11
  2.1 Artificial Neural Networks .......................................... 11
    2.1.1 Artificial Neurons .............................................. 12
    2.1.2 Multilayer Perceptron .......................................... 13
    2.1.3 Training MLP .................................................. 16
  2.2 Recurrent Fuzzy Neural Networks ................................... 22
    2.2.1 Fuzzy Systems .................................................. 22
    2.2.2 Recurrent Fuzzy Neural Networks ............................. 23
    2.2.3 Training RFNN .................................................. 25
  2.3 Mixture of Experts .................................................. 26
    2.3.1 Mixture of Implicitly Localized Experts ...................... 28
    2.3.2 Mixture of Explicitly Localized Experts ..................... 30
    2.3.3 Comparing MILE with MELE ................................... 31

3 Improvements of RFNN ............................................... 33
  3.1 Enhancing Back-Propagation with Genetic Algorithms ............. 33
    3.1.1 Genetic Algorithm ............................................. 34
    3.1.2 Binary Encoding ............................................... 34
    3.1.3 A Hybrid of Back-Propagation and Genetic Algorithm ......... 35
  3.2 Chaotic expressions ............................................... 36
    3.2.1 Data Pre-processing ........................................... 37
    3.2.2 Reconstructing Phase Space by Chaotic Expressions ......... 38
    3.2.3 False-nearest-neighbor method ................................ 38
    3.2.4 Minimum delay time ............................................ 39
  3.3 Mixture of Recurrent Fuzzy Neural Networks ....................... 39
    3.3.1 Architecture .................................................. 39
    3.3.2 Clustering Algorithms ......................................... 42
    3.3.3 Dynamic Time Warping ......................................... 44
4 RFNN and River Runoff Prediction 47
4.1 Context ................................................. 47
  4.1.1 Study Area .............................................. 47
  4.1.2 Dataset ................................................. 49
4.2 Experimental Results ................................. 49
  4.2.1 Short-term Prediction ............................... 50
  4.2.2 Long-term Prediction .............................. 54
4.3 Summary ............................................... 59

5 RFNN and Boiler Efficiency Optimization 61
5.1 Context ............................................... 61
5.2 Simulation of Boiler Efficiency ......................... 62
  5.2.1 Background .......................................... 63
  5.2.2 Soft Sensor ........................................... 63
  5.2.3 Simulating Boiler Efficiency by RFNN ............. 64
  5.2.4 Experimental Results ............................... 65
5.3 Forecasting Multi-Step-Ahead Real-Time Boiler Efficiency 67
  5.3.1 MSA Forecasting Strategies ......................... 69
  5.3.2 Hybrid of Stochastic Exploration and RFNN ........ 69
  5.3.3 A Reinforcement Learning Algorithm for RFNN .... 71
  5.3.4 Experimental Results ............................... 72
5.4 Summary ............................................... 77

6 Related Works ........................................... 79
  6.1 River Runoff Prediction ............................... 79
  6.2 Boiler Efficiency Optimization ....................... 82

7 Conclusion and Perspectives 87
  7.1 Conclusion .............................................. 87
  7.2 Perspectives ............................................ 89
  7.3 Publications ........................................... 90

Bibliography ............................................... 93

A The Soft Sensor - BEO .................................. 103
  A.1 Architecture of BEO ................................. 103
    A.1.1 Real-Time Monitoring ......................... 104
    A.1.2 Data Pre-Processing ......................... 104
    A.1.3 Data Clustering ................................. 104
    A.1.4 Anomaly Detection .............................. 104
    A.1.5 Efficiency Calculator ......................... 104
    A.1.6 Boiler Efficiency Simulation .................. 105
    A.1.7 Boiler Efficiency Optimization ............... 105
    A.1.8 Boiler Controller ............................... 106
    A.1.9 Multi-Step-Ahead Real-Time Forecasting ...... 107
Chapter 1

Introduction

Chapter 1 begins with the history of artificial neural networks and then introduces the context of Vietnamese urgent practical problems that are investigated and tackled in this thesis. They consist of river runoff prediction and boiler efficiency optimization. Chapter 1 also clarifies the contributions of this dissertation. Finally, the organization of the dissertation is presented.

Contents

1.1 Introduction ............................................. 1
1.2 Motivations ............................................ 3
   1.2.1 Climate Change and Problems of River Runoff Prediction .. 3
   1.2.2 Boiler Efficiency Optimization .......................... 5
1.3 Contributions ......................................... 6
   1.3.1 River Runoff Prediction ............................. 7
   1.3.2 Boiler Efficiency Optimization ........................ 7
1.4 Organization of Thesis ................................. 8

1.1 Introduction

Bio-Inspired Computing, which is short for Biologically Inspired Computing, exploits computer strength to model and study living phenomena, as well studying life to improve the usage of computers. Bio-Inspired Computing is an exciting and relatively recent field and belongs to natural computation [de Castro 2005]. Over the last few decades, many computing methods of Bio-Inspired Computing have been used successfully to find good solutions to difficult problems in diverse areas, such as optimization, decision support systems, pattern recognition, machine learning, computer security, time series prediction, image processing, etc. In short, Bio-Inspired Computing provides a powerful set of computing methods that can be applied for optimizing and modeling in many diverse areas - not only in science, but also in business, industry, environment, healthcare and so on.

Among several study areas of Bio-Inspired Computing such as Evolutionary Computation, Cellular Automata, Computer Immune Systems, and Artificial Neural Networks (ANNs) have been applied widely in various fields [Kar 2014, Haykin 2009, Kamruzzaman 2006]. The first ANN was invented in 1958 by psychologist Frank
Rosenblatt, who was inspired by human brain operation [Frank 1958]. Called Perceptron, it was intended to simulate how the human brain processes and learns data to recognize some sophisticated features of the data. By the late 1980s, many scientists had started using ANNs for a variety of purposes. To date, there have been extensive amounts of research involving ANNs. In addition to studying the application of neural networks in the real world, the majority of research has explored different aspects of ANNs to improve their performance.

Typically, computers can solve many real-world problems quite well. They accomplish tasks quite fast and do exactly what people tell them to do. It is important to note that these problems must be fully described within a language that computers can understand; the descriptions are called algorithms. Unfortunately, computers can’t help people if people themselves don’t fully understand the problems they want to solve. Further, standard algorithms don’t deal well with complex problems involving sophisticated or incomplete data. For example, people have a dataset of stocks; they want computers to learn from the dataset and predict what happens in the future. Obviously, this is difficult for the computers if people don’t know which algorithms to use to guide the computers. Fortunately, ANN is a brilliant solution for the kinds of problems.

Due to the strength of ANNs, they have been widely used to solve various problems such as time series prediction, fitness approximation, speech recognition, handwriting recognition, image classification and so on [Kar 2014, Haykin 2009, Kamruzzaman 2006]. It has been particularly noted that ANN is an interesting tool to solve problems of time series forecasting and prediction. To date, there have existed several methods, such as linear regressions, nonlinear regressions, fuzzy systems, support vector machines, etc., that are able to do the same tasks as artificial neural networks. Each method has many advantages and also disadvantages depending on the specific dataset; it is atypical to use a single method that achieves the best results for the overall problem domain [Dietterich 2000]. Among these methods, ANN is especially interesting because of its effectiveness and straightforward idea.

In Vietnam, agriculture is one of the major industries and contributes significantly to the national Gross Domestic Product (GDP). Despite the trending away from agriculture, agriculture has still contributed approximately 15 – 20% to the Vietnamese GDP in the last few years [McCaig 2013]. Moreover, Vietnam is among the top 5 rice export countries\(^1\) in the world, contributing about 7.4% (equivalent to 1.8 billion USD), in 2015. Thus, it is necessary to significantly support Vietnamese agriculture in many aspects, such as national policies, deep agriculture technologies, applications of computer science and so on. To date, there has been scant significant research on computer science applications to support Vietnamese agriculture. The distance between theory and practice has been vast. The main reason is that there are few Vietnamese scientists whose knowledge of theory and practice involves agriculture.

\(^1\)http://www.worldstopexports.com/rice-exports-country
1.2 Motivations

Considering the practical demand, in this thesis, we focus on applying ANNs to solve some urgent practical challenges affecting Vietnamese agriculture, including the hydrology and industry of fertilizer production. In particular, we use ANNs, which are improved by evolutionary algorithms, fuzzy systems, chaotic expressions, and clustering algorithms, to predict river runoff and optimize boiler efficiency.

1.2 Motivations

1.2.1 Climate Change and Problems of River Runoff Prediction

Climate change is one of the greatest challenges for humanity in the 21st century. It seriously affects economic production, life, the environment, etc., of many countries in the world generally and Vietnam particularly. Therefore, most countries in the world have made it a high priority to accommodate climate change in their national development plans. The Vietnamese Prime Minister, on December 02, 2008, approved a national target program accommodating climate change. Two of eight important missions in the program are: (i) to consider how climate change affects
production and civilians and (ii) to determine relevant solutions. Consequently, some researchers are investigating river runoff prediction.

Figure 1.3: The salinization in the Mekong Delta and its damage, 2016

Source: Southern Institute of Water Resources Research, Vietnam

Rice areas damaged by the salinization

Source: Ministry of Agriculture and Rural Development, Vietnam

---

\(^2\)Source: http://vnexpress.net/infographics/thoi-su/
1.2. Motivations

In Vietnam, agriculture is a major industry, and thus rivers play a central role in livelihoods and in production around the basin areas. Some important Vietnamese rivers include the MeKong River (in southern Vietnam), the Srepol River (in the Central Highland of Vietnam), and the Hong River (in northern Vietnam). In recent years, climate change has seriously impacted these rivers. In 2015, due to the El Nino phenomenon, the rainy season ended early in southern Vietnam. As a result, the MeKong River is almost out of water; salinization began attacking the MeKong River in the beginning of 2016. Consequently, agriculture in the area is extremely impacted. Figure 1.1 and Figure 1.3 illustrate the salinization and the damage of rice areas in the Mekong Delta. Recently, similar situations have occurred in other areas such as the Srepol basin and the Hong basin; and people's livelihoods and production around these basin areas have been threatened. For example, in 2015, there was a serious drought in a large area upstream of the Srepol River; and most coffee trees died due to lack of water (Figure 1.2).

Due to these abnormalities, it is necessary to develop some tools that can predict what happens to the rivers. People can apply many different methods such as physical-driven methods and data-driven methods. In this thesis, ANNs are employed to predict river runoff and the Srepol River is chosen as a case study.

1.2.2 Boiler Efficiency Optimization

In Vietnam, due to the demand for fertilizer in agricultural production, fertilizer plants play an important role. Among many such plants, Phu My Fertilizer Plant was established March 28, 2003 and officially went into operation on February 19, 2004. It is the biggest plant in Vietnam. The functions and duties of Phu My Fertilizer Plant are to produce and trade urea fertilizer, liquid ammoniac, industrial gas and other chemical products. Currently, Phu My Fertilizer Plant production fulfills roughly 50% of the total urea domestic demand (2 million tons per year) in Vietnam.

Critically fierce competition in the modern industrial economy forces companies to seek strategies to reduce cost, increase productivity, and improve production efficiency. If a large quantity of goods is produced, growth of even one percent in a year can bring considerable profits. At Phu My Fertilizer Plant, the managers are constantly exploring new solutions to increase productivity.

In fertilizer plants generally, and Phu My Fertilizer Plant particularly, boilers are the most important components. The managers of the plant always pay close attention to improving efficiency of boilers, or at least keeping efficiency of boilers stable. During operation, boiler efficiency sometimes decreases and causes some damage at the plant. In this thesis, some hybrid methods of ANN are used to optimize the boiler efficiency at Phu My Fertilizer Plant. First, a hybrid of ANN and fuzzy systems called RFNN is applied to simulate boiler efficiency. Second, RFNN and its hybrid methods are applied to forecast real-time boiler efficiency many steps ahead of time. Figure 1.4 summarizes the process of using a soft sensor

to optimize boiler efficiency. At Phu My Fertilizer Plant, boilers are monitored in real-time via a Distributed Control System (DCS), namely DCS Centum CS3000 [Yokogawa 2006] which was deployed by Yokogawa Electric Corporation. Boiler efficiency is regularly forecasted by the Multi-Step-Ahead Real-Time Forecasting Module to detect downtrends of boiler efficiency. When the downtrends are about to manifest, the Boiler Efficiency Optimization Module looks in the database for some adjustments of control parameters that can keep the boiler efficiency going up. Then Boiler Efficiency Simulation Module will be used to verify if the adjustments can really increase the boiler efficiency or not. In the case of positive adjustments, the Boiler Controller Module will adjust some control parameters of the boilers to increase the boiler efficiency.

![Figure 1.4: Process of Boiler Efficiency Optimization](image)

1.3 Contributions

In short, the aim of the research reported in this thesis is to apply ANNs combined with other theories such as fuzzy systems, chaotic expressions, and clustering algorithms to predict Vietnamese river runoff and optimize boiler efficiency.
1.3. Contributions

1.3.1 River Runoff Prediction

For the first problem, we attempt to predict river runoff in two scenarios: short-term prediction and long-term prediction. We choose the Srepok River as a case study.

**Short-term prediction.** For the objective of short-term prediction, we propose two methods involving ANN. In this case, we consider river runoff as one-dimension time series data.

- We use Recurrent Fuzzy Neural Network (RFNN) which is a hybrid method combining fuzzy systems and artificial neural networks to predict the Srepok runof.
- We improve the performance of prediction by applying chaos expressions to highlight temporal features of the Srepok runoff. Then, we predict the Srepok runoff based on the highlighted data. Due to some new characteristics of the highlighted data, we propose a new hybrid approach that combines RFNNs and clustering algorithms. We test the hybrid approach with two clustering algorithms consisting of K-means and DBSCAN. Moreover, we also test the hybrid approach with two distance measures, including Euclid distance and Dynamic Time Warping.

**Long-term prediction.** For the objective of long-term prediction, we also propose two methods. In this case, we use the data set consisting of river runoff and climate data.

- We also use RFNN to explore correlations between climate data and river runoff. Via these correlations, we simulate and predict the Srepok runoff in long-term.
- We continue improving the performance of RFNN by utilizing an evolutionary algorithm called Genetic Algorithm to expand the search space of the learning phase of RFNN.
- To prove the effectiveness of the proposed methods, we compare the experimental results of two methods with Soil and Water Assessment Tool (SWAT), which is a physical-based method.

1.3.2 Boiler Efficiency Optimization

For the second problem, we attempt to optimize boiler efficiency. We solve two sub-problems as follows.

- We utilize RFNN and clustering algorithms to simulate boiler efficiency. The module of boiler simulation is an important component of a sophisticated soft sensor, namely BEO.
We attempt to forecast multi-step-ahead real-time boiler efficiency motivated by practical issues. We solve this problem by using three methods: RFNN, the hybrid of RFNN and stochastic exploration, and RFNN improved by a reinforcement learning algorithm.

1.4 Organization of Thesis

The rest of this thesis is organized as follows.

- Chapter 2: Background. In this chapter, fundamentals of artificial neural networks and fuzzy systems are presented. In particular, the theory of RFNN is presented in detail. The fundamentals of a mixture of experts are also introduced in this Chapter.

- Chapter 3: Improvements of RFNN. Chapter 3 firstly introduces an improvement of ANN by utilizing genetic algorithms. Then the chapter presents chaotic expressions which are employed to enrich the temporal characteristic of time series data. Finally, a hybrid approach is proposed according to the concept of mixture of experts. Some algorithms used in the hybrid approach include Dynamic Time Warping, K-means, and DBSCAN, and are also presented in detail.

- Chapter 4: RFNN and River Runoff Prediction. Some experimental results of river runoff prediction are presented. We present the comparison of these experimental results to find out which are the most suitable methods for real deployments.

- Chapter 5: RFNN and Boiler Efficiency Optimization. We present the experimental results of our proposed methods for solving some problems of boiler efficiency optimization. The solutions include boiler efficiency simulation and multi-step-ahead real-time boiler efficiency forecasting.

- Chapter 6: Related works. A few recent works relating to river runoff prediction and boiler efficiency optimization are introduced.

- Chapter 7: Conclusion and Perspectives. The conclusion and the future directions, which we intend to take, are presented in this Chapter.

- Appendices
  - Appendix A: The Soft Sensor - BEO. In this Appendix, we present the architecture of BEO and demonstrate the benefits it brought to Phu My Fertilizer Plant.

Figure 1.5 shows the organization of this thesis with all the chapters and their principal notions and links.
1.4. Organization of Thesis

<table>
<thead>
<tr>
<th>CHAPTERS</th>
<th>IMPORTANT NOTIONS AND LINKS</th>
</tr>
</thead>
</table>
| CHAPTER 1 Introduction | Context (Section 1.1)  
River Runoff Prediction (Section 1.2.1)  
Boiler Efficiency Optimization (Section 1.2.2) |
| CHAPTER 2 Backgrounds | Artificial Neural Networks (Section 2.1)  
Recurrent Fuzzy Neural Networks (Section 2.2)  
Mixture of Experts (Section 2.3) |
| CHAPTER 3 Improvements of RFNN | Enhancing Back-Propagation by Genetic Algorithm (Section 3.1)  
Chaotic Expressions (Section 3.2)  
Mixture of RFNNs (Section 3.3) |
| CHAPTER 4 RFNN and River Runoff Prediction | Short-term Prediction (Section 4.2.1)  
Long-term Prediction (Section 4.2.2) |
| CHAPTER 5 RFNN and Boiler Efficiency Optimization | Boiler Efficiency Simulation (Section 5.2)  
Forecasting Multi-Step-Ahead Real Time Boiler Efficiency (Section 5.3)  
Hybrid of Stochastic Exploration and RFNNs (Section 5.3.2)  
A Reinforcement Learning Algorithm for RFNNs (Section 5.3.3) |
| CHAPTER 6 Related Works | River Runoff Prediction (Section 6.1)  
Boiler Efficiency Optimization (Section 6.2) |
| CHAPTER 7 Conclusion and Perspectives | |

Figure 1.5: Organization of this thesis
Chapter 2 begins with the theory of artificial neurons and some basic concepts of artificial neural networks (ANNs). Then we present fuzzy systems and a hybrid of fuzzy systems and ANNs, called recurrent fuzzy neural network (RFNN). The Chapter finishes with the theory of mixture of experts (ME).

Contents

2.1 Artificial Neural Networks ................................................. 11
    2.1.1 Artificial Neurons ............................................... 12
    2.1.2 Multilayer Perceptron ........................................... 13
    2.1.3 Training MLP .................................................... 16

2.2 Recurrent Fuzzy Neural Networks ..................................... 22
    2.2.1 Fuzzy Systems .................................................... 22
    2.2.2 Recurrent Fuzzy Neural Networks .............................. 23
    2.2.3 Training RFNN ................................................... 25

2.3 Mixture of Experts ....................................................... 26
    2.3.1 Mixture of Implicitly Localized Experts ....................... 28
    2.3.2 Mixture of Explicitly Localized Experts ...................... 30
    2.3.3 Comparing MILE with MELE ................................. 31

2.1 Artificial Neural Networks

In 1958, the first ANN was invented by psychologist Frank Rosenblatt [Frank 1958]. Since then, there have been significant amounts of research that attempt to improve the performance of ANNs and apply ANNs to real-world problems [Haykin 2009]. These researchers on artificial neural networks (ANNs) were inspired by simulations of how the brain works in humans and other mammals [Frank 1958, Haykin 2009]. The authors think of the human brain as a highly complex, nonlinear and parallel computer or information processing system capable of performing highly complex tasks. It is a fact that the brain is composed of cells called neurons. These neurons are responsible for performing complex computations as pattern recognition, perception or control. Typically, an artificial neural network is built up by a network of computing units, known as artificial neurons. These computing units are represented as nodes in the network and they are connected with each other through weights.
2.1.1 Artificial Neurons

The computing units that are important components of a neural network, are called artificial neurons, or neurons for short. Figure 2.1 shows a typical model of an artificial neuron. The neural model is composed of the following elements:

- A set of synapses or connection links, each of which is represented by a weight. A signal $x_j$ at the input of synapse $j$ connected to neuron $k$ is multiplied by the synaptic weight $w_{kj}$.
- All the input signals after multiplied by the respective synaptic weights, are summed together. These operations form a linear combiner.
- An activation function, $\phi(\cdot)$, is responsible for nonlinearizing the output of a neuron.

In the neural model presented in Figure 2.1, we can see a bias, $b_k$. The effect of the bias is to decrease or increase the net input of the activation function depending on whether it is negative or positive. A mathematical representation of the artificial neuron in Figure 2.1 is given by Equation 2.1.

$$y_k = \varphi\left(\sum_{j=1}^{n} (w_{kj}x_j) + b_k\right),$$

(2.1)

where $x_1, x_2, ..., x_n$ are the input signals and $w_{k1}, w_{k2}, ..., w_{kn}$ are the respective synaptic weights of neuron $k$.

Some activation functions $\varphi(\cdot)$ which are commonly used, are presented in Table 2.1. So far, the sigmoid function, sometimes called the logistic function, is the most common activation function used in several types of ANNs. It is regarded as a strictly increasing function that allows a level of balance between linear and nonlinear behaviors. One of the most interesting properties of the sigmoid function is that it is differentiable, a very useful property, to train neural networks. Hyperbolic tangent function is also common. The softmax activation function is commonly used...
in output layers of ANNs applied for problems of classification. In these ANNs, the softmax function converts a crisp value into a posterior probability.

### Table 2.1: Activation Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Formula</th>
<th>Figure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sigmod</td>
<td>$\varphi(x) = \frac{1}{1+e^{-ax}}$</td>
<td><img src="image1.png" alt="Fig1" /></td>
</tr>
<tr>
<td>Hyperbolic</td>
<td>$\varphi(x) = \frac{e^x-e^{-x}}{e^x+e^{-x}}$</td>
<td><img src="image2.png" alt="Fig2" /></td>
</tr>
<tr>
<td>Binary step</td>
<td>$\varphi(x) = \begin{cases} 1 &amp; \text{if } x \geq 0 \ 0 &amp; \text{if } x &lt; 0 \end{cases}$</td>
<td><img src="image3.png" alt="Fig3" /></td>
</tr>
<tr>
<td>Softmax</td>
<td>$\varphi(x)<em>j = \frac{e^{x_j}}{\sum</em>{k=1}^{K} e^{x_k}}$, where $j = 1, \ldots, K$</td>
<td><img src="image4.png" alt="Fig4" /></td>
</tr>
</tbody>
</table>

2.1.2 Multilayer Perceptron

According to the different combinations of the artificial neurons, many types of neural networks have been proposed, such as Multilayer FeedForward Neural Networks (Figure 2.2), Recurrent Neural Networks (Figure 2.3) and so on. The Multilayer Perceptron (MLP), a type of Multilayer FeedForward Neural Network, consists of neurons whose activation functions are differentiable [Haykin 2009]. MLP has one or more hidden layers containing computation nodes. The computation nodes sometimes are called hidden neurons or hidden units. The term "hidden" is used because
those layers are not seen from either the input or the output layers. The task of these hidden units is to take part in the analysis of data between the input and output layers. By adding one or more hidden layers, the network can be capable of discovering many sophisticated relations between input and output of MLP.

![Multilayer FeedForward Neural Network](image)

**Figure 2.2:** A Multilayer FeedForward Neural Network with 4 input nodes, 3 hidden nodes, and 1 output node

![Recurrent Neural Network](image)

**Figure 2.3:** A Recurrent Neural Network with recurrent relations from output nodes to hidden nodes

Let $u_i^{(k)}$ and $O_i^{(k)}$ be the input and the output of the node $i_{th}$ in the layer $k$ respectively. A short following description presents the operations of a MLP consisting of one input layer of $N$ input nodes, one hidden layer of $M$ hidden nodes, and one output layer of $P$ output nodes.

**Input layer**
2.1. Artificial Neural Networks

\[ O_i^{(1)} = u_i^{(1)} = x_i, \text{ where } i = 1 \div N. \] (2.2)

**Hidden layer**

\[ u_j^{(2)} = O_j^{(1)} \]

\[ O_j^{(2)} = \varphi\left(\sum_{i=1}^{N} (w_{ji}x_i) + b_j\right), \text{ where } j = 1 \div M \] (2.3)

**Output layer**

\[ u_k^{(3)} = O_k^{(2)}, \]

\[ y_k = O_k^{(3)} \]

\[ = \varphi\left(\sum_{j=1}^{M} (w_{kj}u_j^{(3)}) + b_k\right) \]

\[ = \varphi\left(\sum_{j=1}^{M} w_{kj}(\varphi\left(\sum_{i=1}^{N} (w_{ji}x_i) + b_j\right)) + b_k\right), \text{ where } k = 1 \div P \] (2.4)

When MLP has more one hidden layer, its working process is the same, but in this case, Equation 2.3 is calculated \( H \) times, in which \( H \) is the number of hidden layers. Observing the working process of MLP, we can easily realize that the relation between input and output of MLP is modeled by a nonlinear function \( y = f(x) \). In this nonlinear function, \( x \) is input vector, \( y \) is output vector, \( f(.) \) is a nonlinear function which is formed by mixing a large number of sum and activation functions and it is described in detail as Equation 2.4. In [Haykin 2009], the authors stated and proved the capability of function approximation of MLP by Theorem 1.

**Theorem 1** Let \( \varphi(.) \) be a non-constant, bounded and monotone-increasing continuous function. Let \( I_{m_0} \) be the \( m_0 \)-dimensional unit hypercube \([0,1]^{m_0}\). The space of continuous functions on \( I_{m_0} \) is denoted by \( C(I_{m_0}) \). Then, given any function \( f \in C(I_{m_0}) \) and \( \varepsilon > 0 \), then there exists an integer \( m_1 \) and sets of real constants \( \alpha_i, b_i \) and \( w_{ij} \), where \( i = 1, ..., m_1 \) and \( j = 1, ..., m_0 \) such that we may define.

\[ F(x_1, x_2, ..., x_{m_0}) = \sum_{i=1}^{m_1} \alpha_i \left( \sum_{j=1}^{m_0} w_{ij}x_j + b_j \right), \] (2.5)

as an approximation realization of the function \( f(.) \), that is

\[ |F(x_1, x_2, ..., x_{m_0}) - f(x_1, x_2, ..., x_{m_0})| < \varepsilon, \] (2.6)

for all \( x_1, x_2, ..., x_{m_0} \) that lie in input space.
MLPs can be used for many tasks such as remote sensing, voice detection, time series forecasting and prediction, and so on. Typically, these MLPs must be learned before they are used to solve the tasks. In the next section, we will discuss a popular training algorithm called Back-Propagation.

2.1.3 Training MLP

2.1.3.1 Supervised Learning

Supervised learning is a process that attempt to learn or train a model like MLP using labeling training data. The training data consists of many tuples, that is, a pair of input vector and corresponding output vector. According to the training data, supervised learning trains the model in order to produce an inferred function which can approximate the relation between input and output of the training data. While training the model, output vectors of the training data play the role of orientation to produce the best set of parameters that constitute the inferred function. To train MLPs, supervised learning is commonly chosen. Figure 2.4 illustrates the process of training MLP by supervised learning. After $y_{0}$ is produced by Equation 2.4, the difference of $y_{0}$ and real value (target value) $y_{0}^{d}$ called $E$ is used to adjust all parameters (weights and biases) of MLP toward decreasing the difference of $E$. The most common method under supervised learning strategy is the so-called steepest descent method, which is introduced in the next section.

Figure 2.4: Supervised learning on MLP with 4 input nodes, 3 hidden nodes, and 1 output node

2.1.3.2 Steepest Descent

Steepest descent updates the weights in the direction opposite to the gradient vector $-\frac{\partial E}{\partial w}$, in which $E = \frac{1}{2} \sum_{k=1}^{K} (y_{k}^{d} - O_{k}^{(3)})$, $y_{k}^{d}$ is the target vector and $K$ is the number of output nodes. The rule of MLP parameter updating is as follow.

- The weights and biases of connections between output layer and hidden layer are updated as seen in Equation 2.7, where $\eta$ is step size or learning rate.
2.1. Artificial Neural Networks

\[ w_{kj}^{new} = w_{kj}^{old} + \Delta w_{kj}, \text{ where } \Delta w_{kj} = -\eta \frac{\partial E}{\partial w_{kj}}. \] (2.7)

\[ b_{k}^{new} = b_{k}^{old} + \Delta b_{k}, \text{ where } \Delta b_{k} = -\eta \frac{\partial E}{\partial b_{k}}. \] (2.8)

- Similarly, the weights and biases of connections between hidden layer and input layer are updated as seen in Equation 2.9.

\[ w_{ji}^{new} = w_{ji}^{old} + \Delta w_{ji}, \text{ where } \Delta w_{ji} = -\eta \frac{\partial E}{\partial w_{ji}}. \] (2.9)

\[ b_{j}^{new} = b_{j}^{old} + \Delta b_{j}, \text{ where } \Delta b_{j} = -\eta \frac{\partial E}{\partial b_{j}}. \] (2.10)

Because \( E = \frac{1}{2} \sum_{k=1}^{K} (y_{k}^{d} - O^{(3)}_{k}) \) and \( O^{(3)}_{k} \) is calculated by Equation 2.4, \( E \) is differentiable on \( w_{ji} \), \( b_{j} \), \( w_{kj} \), and \( b_{k} \) if the chosen activation function is differentiable.

We simplify Equations 2.7, 2.8, 2.9, and 2.10 by assuming the biases \( b_{j} \) and \( b_{k} \) which are weights of the 1-valued inputs of hidden and output nodes, respectively. If the activation function is the sigmoid function, these equations of weight-updating are as follows.

\[ \Delta w_{kj} = \delta_{k}O^{(2)}_{j}, \] (2.11)

\[ \Delta w_{ji} = \mu_{j}x_{i}, \] (2.12)

\[ \delta_{k} = (y_{k}^{d} - O^{(3)}_{k})O^{(3)}_{k}(1 - O^{(3)}_{k}), \] (2.13)

\[ \mu_{j} = \left[ \sum_{k=1}^{P} \delta_{k}w_{kj} \right] O^{(2)}_{j}(1 - O^{(2)}_{j}). \] (2.14)

2.1.3.3 Back-Propagation

The Back-Propagation (BP) algorithm based on steepest descent method, is used to update the parameters of MLP. The BP algorithm, which was first published by Werbos in 1974 [Werbos 1974], works by passing the data in two opposite phases, called forward phase and backward phase.

Forward phase. Back-Propagation is a type of supervised learning algorithm. Thus it is necessary to build up a labeling training data consisting of many tuples, that is, a pair of input vector and corresponding output vector. In the forward phase, for each tuple of the training data, input vector of the tuple is passed through the
synaptic weights from one layer to the next, until the data finally emerges in the output nodes.

The function signal emitting out from the network is expressed as Equation 2.4. Then the output vector \( y \) produced by MLP is compared to the real corresponding output vector and gives an error \( E \).

**Backward phase.** In the backward phase, we start at the output nodes and go through all the layers in the network, and recursively compute the value of adjustment for each neuron in every layer. For each output neuron and hidden neuron, its bias is updated; for each synapse, its weights are updated. Updating principles are the steepest gradient method, described in detail in Equations 2.7 and 2.9.

Finally, we summarize BP algorithm as Figure 2.5.

### 2.1.3.4 Notions of Back-Propagation

#### Batch and On-Line Learning

**Batch Learning.** For the batch learning method, after \( N \) tuples are passed through MLP, the sum of all errors is used to adjust the weights and biases of the network one time. Using gradient descent with the batch learning method offers two advantages.

1. *Accurate estimation* of the gradient vector and convergence to a local minimum.
2. *Parallelization* of the training phase.

**On-Line Learning.** On-line learning means that adjustment to the weights of MLP is done by the way of tuple-by-tuple. Figure 2.5 shows the idea of the on-line learning method. Some advantages of this method are as follows.

1. It requires less storage resources.
2. It is well suited for large-scale and difficult pattern classification problems.
3. It is simple to implement.

**Terminating Conditions.** Back-Propagation algorithm converges into optima that can be local or global. Corresponding to the optima of BP, weight vector of MLP is \( w^* \). Typically, BP converges to the optima which is the nearest to initial values of weight vector; the optima are local. To reach the local optima, BP must be repeat *epoch* times of forward and backward phases. Depending on the value of learning rate and structure of MLP, the time consumption of the training phase is small or large. Until now, there has not existed a standard for choosing the best
2.1. Artificial Neural Networks

Start

Initialize randomly weights and biases

For each tuple in training data, its vector input is passed through all layers of network

Get one error $E$ at output layer

Update weights and biases according to the error $E$

Next tuple?

Evaluate overall error of network

is terminating conditions satisfied?

No

Yes

Forward phase

Backward phase

End

Figure 2.5: Procedure of Back-Propagation algorithm
MLP coefficients, such as learning rate, the number of hidden nodes, etc. People choose the MLP coefficient based on experiments and experience that are almost totally dependent on specific training data. Therefore, ANNs in general and MLP in particular are black-boxes for end users.

**Over-fitting.** To build a smarter MLP based on training data, we divide the training data into two sets: training set and testing set. As usual, we train the MLP with the training set and verify it with the testing set to ensure its generalisation capacity. Normally, the longer the training phase takes, the smarter the MLP will become. That means the longer the training phase takes, the better results the testing phase produces. Occasionally, the longer the training phase takes the worse the results produced by the testing phase. The problem is called over-fitting and Figure 2.6 illustrates the phenomena.

Therefore, some terminating conditions of BP are considered as follows.

- Overall error of MLP is less than a threshold that is entered by an end user.
- The number of iterations of forward and backward phases is larger than a threshold.
- When an over-fitting event appears.

![Figure 2.6: Over-fitting in the training phase](image)

**Momentum technique.** The momentum technique, first proposed by Polyak in 1964, can be applied for the BP algorithm to speed up the training phase and help BP jump over a few narrow local minima [Polyak 1964]. In this case, the adjustment of weights and biases of MLP at each iteration is based on the current and previous errors. The process of the adjustment at \( t^{th} \) iteration is presented in Equations 2.15, 2.16, 2.17, and 2.18, in which \( \beta \) is momentum value.

\[
w_{kj}(t + 1) = w_{kj}(t) - \eta \frac{\partial E}{\partial w_{kj}} + \beta \triangle w_{kj}(t - 1),
\]  

(2.15)
2.1. Artificial Neural Networks

\[ b_k(t+1) = b_k(t) - \eta \frac{\partial E}{\partial b_k} + \beta \Delta b_k(t-1), \quad (2.16) \]

\[ w_{ij}(t+1) = w_{ji}(t) - \eta \frac{\partial E}{\partial w_{ji}} + \beta \Delta w_{ij}(t-1), \quad (2.17) \]

\[ b_j(t+1) = b_j(t) - \eta \frac{\partial E}{\partial b_j} + \beta \Delta b_j(t-1). \quad (2.18) \]

2.1.3.5 Heuristics For Back-Propagation

There are some tested design choices which improve the back-propagation algorithm performance. Below is a list of proven methods [Haykin 2009].

1. **Update choice.** Selection of batch or on-line learning is depended on a specific training dataset. On-line learning is more interesting than batch because the BP converges faster.

2. **Activation function.** Sigmoid function is the most popular. However, hyperbolic tangent function is a better one.

3. **Target values.** The target values should be within the range of the activation function. For example, if the activation function is the sigmoid function, the target values should be normalized to \((0, 1)\).

4. **Normalizing input values.** All values of input vectors should be normalized to the same range so that all elements of input vectors contribute the same roles to the output values.

5. **Initialization.** Typically, the values of weights and biases are initialized randomly. Haykin shows that we should initialize the weights according to random values from a uniform distribution with mean zero and variance equal to the reciprocal of the number of synaptic connections of a neuron [Haykin 2009].

6. **Highlighting training data.** The main task of MLP is to represent the mapping of input space and output space. The representation is \( f(.) \) as Equation 2.4. Therefore, if input space and output space have close correlations, the training phase will have some guiding hints and produce a perfect mapping \( f(.) \) after finishing learning. Data highlighting is responsible for increasing the correlations. For example, for predicting river runoff, we know river runoff has seasonal rules, hence we add the time factor into training data as a new dimension of input space.

7. **Learning rates.** If learning rates are small, the BP algorithm takes a long time to converge. Whereas, when learning rates are large, BP algorithm is able to jump over meeting optima. Thus learning rates are usually chosen by experimenting or some simple heuristic techniques. In [Lecun 1993] LeCun introduced a heuristic technique which is very simple but efficient for choosing the optimal learning rate.
2.2 Recurrent Fuzzy Neural Networks

Fuzzy neural networks have been applied in numerous fields [Kar 2014] and RFNN is a well-known fuzzy neural network. The proposed RFNN in [Lee 2000] is re-implemented in this dissertation. RFNN is a hybrid of fuzzy systems and artificial neural networks; thus we briefly introduce fuzzy systems in the next section.

2.2.1 Fuzzy Systems

Typically, the architecture of a fuzzy system consists of four elements as seen in Figure 2.7 [Liu 2004].

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fuzzy_system_architecture.png}
\caption{Fuzzy system architecture}
\end{figure}

Fuzzifier. is responsible for converting a crisp input vector \( x \in \mathbb{R}^n \) into a singleton fuzzy set \( \tilde{x} \). Fuzzifier can be implemented by using a membership function such as Gauss, Sigmoid, Bell, etc. The membership function \( \mu \) will map \( x = \{x_1, x_2, \ldots, x_n\} \) into a singleton fuzzy set \( \tilde{x} = \{a_1, a_2, \ldots, a_n\} \) in which \( a_i \in (0 \div 1) \).

Fuzzy rule base. consists of \( M \) fuzzy rules which present the rules of causes and consequences. Each \( R_j, j \in (1 \div M) \) fuzzy rule contains an implication relation of \( A_1 \times A_2 \times \ldots \times A_n \rightarrow B_j \) in which \( A_i, B_j \) are fuzzy sets.

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{fuzzy_relation_crisp.png}
\caption{Fuzzy relation between crisp inputs and outputs}
\end{figure}
2.2. Recurrent Fuzzy Neural Networks

**Fuzzy inference engine.** uses the fuzzy rules in fuzzy rule base to make some logical decisions. The singleton fuzzy set \( \tilde{x} \) which is applied as fuzzy rule \( R_j \), will give a fuzzy set of inference \( \tilde{Y}_j \). In the fuzzy rule base, we have \( M \) rules. Thus, after inference by the fuzzy inference engine, \( \tilde{x} \) becomes \( \tilde{Y} = \{ \tilde{Y}_1, \tilde{Y}_2, ..., \tilde{Y}_M \} \). \( \tilde{Y} \) is called a synthesizing fuzzy set [Li 2000].

**Defuzzifier.** is responsible for establishing the crisp output vector \( y \) from the synthesizing fuzzy set \( \tilde{Y} \). Consequently, \( y = D_e(\tilde{Y}) \) in which \( D_e \) is a defuzzifier function.

In summary, fuzzy systems are wonderful tools that represent fuzzy relations between independent input \( x \) and dependent input \( y \). Figure 2.8 illustrates the fuzzy relation of input \( x \) and output \( y \) in two-dimensional space. Fuzzy systems are applied widely in practice, especially control systems, speech recognition, game programming, time series prediction, and so on. Furthermore, the combination of fuzzy systems with artificial neural networks creates many kinds of effective hybrid methods such as RFNN.

### 2.2.2 Recurrent Fuzzy Neural Networks

Figure 2.9: RFNN Architecture [Lee 2000]
Fig. 2.9 shows the structure of its four layers. Let \( u_i^{(k)} \) and \( O_i^{(k)} \) be the input and the output of the node \( i_{th} \) in the layer \( k \) respectively. The structure of the RFNN is presented as follows:

**Layer 1**
This is the input layer that has \( N \) nodes, each of which corresponds with a parameter.

\[
O_i^{(1)} = u_i^{(1)} = x_i(t), \quad \text{where } i = 1 \div N. \tag{2.19}
\]

**Layer 2**
This is a membership layer. Nodes in this layer are responsible for converting crisp data into fuzzy data by applying membership functions such as a Gauss function. The number of neural nodes in this layer is \( NxM \) where \( M \) is the number of fuzzy rules. Every node has three parameters, namely \( m_{ij}, \sigma_{ij} \) and \( \theta_{ij} \) respectively.

\[
O_{ij}^{(2)} = \exp \left[ -\frac{(u_{ij}^{(2)} - m_{ij})^2}{\sigma_{ij}} \right], \quad \text{where } i = 1 \div N, \ j = 1 \div M. \tag{2.20}
\]

In Equation 2.20 \( m_{ij} \) and \( \sigma_{ij} \) are the center and the variance of Gauss distribution function.

\[
u_{ij}^{(2)}(t) = O_i^{(1)} + \theta_{ij}O_{ij}^{(2)}(t-1), \quad \text{where } i = 1 \div N, \ j = 1 \div M. \tag{2.21}
\]

In Equation 2.21, \( \theta_{ij} \) denotes the weight of a recurrent node.

We easily realize that the input of nodes in this layer has the factor \( O_{ij}^{(2)}(t-1) \). This factor denotes the remaining information of the previous learning step. Therefore, after replacing \( u_{ij}^{(2)} \) in Equation 2.20 by Equation 2.19, we get Equation 2.21 as follows.

\[
O_{ij}^{(2)} = \exp \left[ -\frac{[O_i^{(1)} + \theta_{ij}O_{ij}^{(2)}(t-1) - m_{ij}]^2}{\sigma_{ij}} \right]
\]

\[
= \exp \left[ -\frac{[x_i(t) + \theta_{ij}O_{ij}^{(2)}(t-1) - m_{ij}]^2}{\sigma_{ij}} \right], \quad \text{where } i = 1 \div N, \ j = 1 \div M. \tag{2.22}
\]

**Layer 3**
This is the layer of fuzzy rules and has \( M \) nodes. Each node in this layer plays the role of a fuzzy rule. Connecting between Layer 3 and Layer 4 presents a fuzzy conclusion. Each node in this layer corresponds with an AND expression. Each AND expression is defined as follows:
2.2. Recurrent Fuzzy Neural Networks

\[ O_j^{(3)} = \prod_{i=1}^{N} O_{ij}^{(2)} \]
\[ = \prod_{i=1}^{N} \exp \left[ -\frac{\left( x_i(t) + \theta_{ij} O_{ij}^{(2)}(t-1) - m_{ij} \right)}{(\sigma_{ij})}^2 \right] \], where \( i = 1 \div N, j = 1 \div M. \]

(2.23)

**Layer 4**

This is the output layer including \( P \) nodes. For objectives of forecasting and prediction, \( P \) will be set to one. Nodes of this layer are responsible for converting fuzzy to crisp.

\[ y_k = O_k^{(4)} \]
\[ = \sum_{j=1}^{M} w_{jk}^{(4)} w_{jk}^{(3)} \]
\[ = \sum_{j=1}^{M} O_j^{(3)} w_{jk} \]
\[ = \sum_{j=1}^{M} w_{jk} \prod_{i=1}^{N} \exp \left[ -\frac{\left( x_i(t) + \theta_{ij} O_{ij}^{(2)}(t-1) - m_{ij} \right)}{(\sigma_{ij})}^2 \right] \], where \( k = 1 \div P. \]

(2.24)

### 2.2.3 Training RFNN

After defining the structure of a RFNN and operations of each layer in detail, we employ the back-propagation (BP) algorithm which is presented in Section 2.1.3.3 to train RFNN. In this research, we also improve BP by momentum technique. Algorithm 3 presents the idea of BP algorithm applied for RFNN.

In Algorithm 3, some derivations are fully presented as the following Equations, in which \( E = \frac{1}{2} \sum_{k=1}^{P} e(t), e(t) = (y_k^d(t) - y_k(t))^2 \) and \( y^d \) is the target vector.
\frac{\partial E}{\partial w_{jk}} = -e(t)O_j^{(3)}, \quad (2.25)

\frac{\partial E}{\partial m_{ij}} = -e(t)\sum_{j=1}^{M}w_{jk}O_j^{(3)}\frac{\partial O_j^{(3)}}{\partial m_{ij}}

= -e(t)\sum_{j=1}^{M}w_{jk}O_j^{(3)}\frac{2\left[x_i^{(t)} + O_i^{(2)}(t-1)\theta_{ij} - m_{ij}\right]}{(\sigma_{ij})^2}, \quad (2.26)

\frac{\partial E}{\partial \sigma_{ij}} = -e(t)\sum_{j=1}^{M}w_{jk}O_j^{(3)}\frac{\partial O_j^{(3)}}{\partial \sigma_{ij}}

= -e(t)\sum_{j=1}^{M}w_{jk}O_j^{(3)}\frac{2\left[x_i^{(t)} + O_i^{(2)}(t-1)\theta_{ij} - m_{ij}\right]^2}{(\sigma_{ij})^3}, \quad (2.27)

\frac{\partial E}{\partial \theta_{ij}} = -e(t)\sum_{j=1}^{M}w_{jk}O_j^{(3)}\frac{\partial O_j^{(3)}}{\partial \theta_{ij}}

= -e(t)\sum_{j=1}^{M}w_{jk}\frac{-2\left[x_i^{(t)} + O_i^{(2)}(t-1)\theta_{ij} - m_{ij}\right]O_i^{(2)}(t-1)}{(\sigma_{ij})^2}. \quad (2.28)

2.3 Mixture of Experts

The mixture of experts (ME) model which was first proposed in [Jacobs 1991], and consists of a set of experts modeling conditional probabilistic processes, and a gate combining the probabilities of the experts. ME is designed based on the Divide-and-Conquer (D&C) principle. In ME, the dataset that is used to train the model, is partitioned stochastically into a number of sub-datasets through a special employed error function. Then many experts are specialized on each sub-dataset. To judge the efficiency of all experts, a gating network is employed and trained together with the experts. The gating network during the training of the experts, simultaneously learns the differences between the efficiency of experts in the different sub-dataset. Therefore, instead of assigning a set of fixed connecting weights to the experts, the gating network is used to compute these weights dynamically from the inputs, according to the local efficiency of each expert.

To date, there has been a vast amount of research on ME that proposed many different kinds of ME models. In [Masoudnia 2012], the authors classified several kinds of ME models into two groups: mixture of implicitly localized experts (MILE) and mixture of explicitly localized experts (MELE). Criteria of the classifications are based on the characteristics of partitioning the training data implicitly or explicitly.
Algorithm 1: Pseudo-code of Back-Propagation

**input:** coefficients of RFNN structure, training set $D$

**output:** RFNN satisfies one of terminating conditions

1. while terminating conditions are not satisfied do
2.   foreach training tuple $X_t$ in training set $D$ do
3.     foreach input layer unit $i$ do
4.       $O_i^{(1)} \leftarrow u_i^{(1)} \leftarrow x_i(t)$
5.   end
6.   foreach membership layer unit $ij$ do
7.     $u_{ij}^{(2)}(t) \leftarrow (O_i^{(1)} + \theta_{ij} O_{ij}^{(2)}(t-1))$
8.   end
9.   foreach layer of fuzzy rules unit $j$ do
10.    $O_j^{(3)} \leftarrow \prod_{i=1}^N \exp \left[ -\frac{[x_i(t)+\theta_{ij} O_{ij}^{(2)}(t-1)-m_{ij}]}{(\sigma_{ij})} \right]$
11.   end
12.   foreach output layer unit $k$ do
13.     $y_k \leftarrow O_k^{(4)} \leftarrow \sum_{j=1}^M u_{jk}^{(4)} \leftarrow \sum_{j=1}^M O_j^{(3)} w_{jk}$
14.     $e_k(t) \leftarrow (y_k^{(d)}(t) - y_k(t))$
15.   end
16.   // $y_k^{(d)}(t)$ is the real river runoff and $y_k(t) \leftarrow O_k^{(4)}(t)$.
17.   // The target of the BP algorithm is how to minimize the sum square error (SSE): $E = \frac{1}{2} \sum_{k=1}^P (y_k^{(d)}(t) - y_k(t))^2$
18.   // update all parameters by gradient descent method.
19.   Denote $\eta$ is learning rate and $\beta$ is momentum
20.   foreach center of membership function $m_{ij}$ do
21.     $m_{ij}(t+1) \leftarrow (m_{ij}(t) - \eta \frac{\partial E}{\partial m_{ij}} + \beta \Delta m_{ij}(t-1))$
22.   end
23.   foreach variance of membership function $\sigma_{ij}$ do
24.     $\sigma_{ij}(t+1) \leftarrow (\sigma_{ij}(t) - \eta \frac{\partial E}{\partial \sigma_{ij}} + \beta \Delta \sigma_{ij}(t-1))$
25.   end
26.   foreach connection weight $w_{jk}$ do
27.     $w_{jk}(t+1) \leftarrow (w_{jk}(t) - \eta \frac{\partial E}{\partial w_{jk}} + \beta \Delta w_{jk}(t-1))$
28.   end
29.   foreach recurrent connection weight $\theta_{ij}$ do
30.     $\theta_{ij}(t+1) \leftarrow (\theta_{ij}(t) - \eta \frac{\partial E}{\partial \theta_{ij}} + \beta \Delta \theta_{ij}(t-1))$
31.   end
32. end
2.3.1 Mixture of Implicitly Localized Experts

The first ME model, introduced in [Jacobs 1991], is MILE. The architecture of the first ME consists of two experts, presented in Figure 2.10. The model consists of 2 experts, namely $\varepsilon_1$, $\varepsilon_2$ and a gate $G$. Each expert $\varepsilon_i$ predicts the value of the target $\hat{y}_i(t)$, according to the conditioned input $x(t)$. The outputs of the experts are weighted by the outputs of the gate $\hat{g}_i(t)$. The overall output of the model is given by the linear sum of the gate and expert outputs as $\hat{y}(t) = \hat{y}_1(t)\hat{g}_1(t) + \hat{y}_2(t)\hat{g}_2(t)$, where $\hat{g}_1(t) + \hat{g}_2(t) = 1$. According to the architecture of 2-experts ME model, we easily generalize the architecture of $n$-experts ME model.

Figure 2.10: Architecture of a mixture of experts consisting 2 experts

Moreover, the ME model can be structured as a tree that constitutes a type of
hierarchical ME (HME) as proposed in [Jordan 1994]. Figure 2.11 shows the structure of a hierarchical ME with two levels. We can enlarge the level of hierarchical ME that is unlimited and produces a structure of deep networks. The working process of HME can be inferred from the principal of the ME model [Jordan 1994].

### 2.3.1.1 Error functions

As mentioned above, the gating network during the training of the experts, simultaneously learns the differences between the efficiency of experts in the different sub-dataset. In other words, in the training phase, all experts and the gate are trained together and simultaneously. Therefore, an error function of the overall ME model is used and contributes significantly to the success of the ME model. Until now, after empirical research on many error functions, in [Jacobs 1991] Jacobs et al. introduced an effective error function based on the negative log probability of generating the desired output vector, assuming a mixture of Gaussian models.

\[
E_{EM} = -\log \sum_{j=1}^{n} g_j e^{-\frac{1}{2}(y^d-\hat{y}_j)^T(y^d-\hat{y}_j)}. \tag{2.29}
\]

In Equation 2.29 \(n\) is the number of experts and \(y^d\) is the target vector.

To evaluate this error function, its deviation with respect to the output of the \(i^{th}\) expert:

\[
\frac{\partial E_{EM}}{\partial \hat{y}_i} = -\left[ \frac{g_i e^{-\frac{1}{2}(y^d-\hat{y}_i)^T(y^d-\hat{y}_i)}}{\sum_{j=1}^{n} g_j e^{-\frac{1}{2}(y^d-\hat{y}_j)^T(y^d-\hat{y}_j)}} \right] (y^d - \hat{y}_i). \tag{2.30}
\]

The learning of each expert on the model is based on its individual error. Moreover, the weight-updating factor for each expert is proportional to the ratio of its error value to the total error. In other words, the larger the error of each expert, the more weight-updating each expert does. By applying an incremental learning strategy such as Back-Propagation, the gate gradually become a judge that can localize a expert on a corresponding sub-dataset. That means the gate limits the impact of each expert to each particular sub-dataset of the training dataset.

Replacing different types of experts and the gate in the ME model can produce many different types of ME models. Among these, one of the most applied implementations is the mixture of MLP-experts (MME) [Waterhouse 1997, Nguyen 2006]. In the next section, we discuss training for MME.

### 2.3.1.2 MME training algorithm

We use Equation 2.29 as the error function of ME. Thus its deviation with respect to weights of experts is as follows.

\[
\frac{\partial E_{EM}}{\partial w_i} = \frac{\partial E_{EM}}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial w_i}. \tag{2.31}
\]
According to the MLP training method presented in Section 2.1.3, the expert’s weight-updating is inferred:

\[
\delta w_{iy} = \eta e_h i (y^d - O_i)O_i (1 - O_i)O^T_{ih}, \quad (2.32)
\]

\[
\delta w_{ih} = \eta e_h i w^T_{iy} (y^d - O_i)O_i (1 - O_i)O_{ih} (1 - O_{ih})x_i, \quad (2.33)
\]

where \( \eta e \) is the learning rate for the experts. \( w_{ih} \) and \( w_{iy} \) are the weights of input-to-hidden and hidden-to-output layer for the \( i \)-th expert, respectively. \( O^T_{ih} \) is the transpose of \( O_{ih} \), the outputs of the hidden layer of \( i \)-th expert. \( h_i \) is inferred from Equation 2.30.

\[
h_i = - \left[ \sum_{j=1}^{n} g_j e^{-\frac{1}{2} \|y^d - \hat{y}_j\|^2} \right] \left( y^d - \hat{y}_i \right). \quad (2.34)
\]

Moreover, the error function of the gate is defined:

\[
E_G = \|h - O_g\|, \quad (2.35)
\]

where \( h = [h_i]_{i=1}^{n} \) and \( n \) is the number of experts. Based on the error function, the gate’s weights is updated by the Back-Propagation algorithm.

\[
\delta w_{gy} = \eta g (h - O_g)O_g (1 - O_g)O^T_{gh}, \quad (2.36)
\]

\[
\delta w_{gh} = \eta g w^T_{gy} (h - O_g)O_g (1 - O_g)O_{gh} (1 - O_{gh})x_i, \quad (2.37)
\]

where \( \eta g \) is the learning rate for the gate. \( w_{gh} \) and \( w_{gy} \) are the weights of input-to-hidden and hidden-to-output layer for the gate, respectively. \( O^T_{gh} \) is the transposing of \( O_{gh} \), the output of the hidden layer of the gate.

### 2.3.2 Mixture of Explicitly Localized Experts

In this group, there have been many types of ME models; some were inspired by MILE [Masoudnia 2012]. Whereas MILE methods stochastically partitions the training dataset and implicitly specializes each expert on an implicit sub-dataset, most of MELE methods explicitly partition the training dataset into many separable sub-datasets, and each expert is then specialized on an explicit sub-dataset. The task of explicit partitioning is often made by a clustering algorithm. For example, in [Goodband 2006], the authors utilized Fuzzy C-Means to partition data before proceeding with training of the experts. The experts then are combined together by Radial-Basis Function gating network. The structure of typical MELE methods is briefly described in Figure 2.12. In Figure 2.12, the judge plays the same role as the gate of MILE methods.
2.3.3 Comparing MILE with MELE

In [Masoudnia 2012], the authors compared the advantages and disadvantages of MILE and MELE. In summary, the advantages and disadvantages of two groups depend on the characteristics of the training dataset. If the distinctness of the training dataset is low, MILE methods are likely more suitable than MELE methods for partitioning the training dataset into many sub-dataset. In contrast, if the distinctness of the training dataset is high, the task of partitioning can proceed more easily by clustering algorithms of MELE than by the gate of MILE. The distinctness can be defined as follows: "The domain of the training dataset can be divided into several completely-separated sub-domains". The distinctness can be analyzed by manual (queries, graphs, etc), theoretical or empirical methods.
Chapter 3 first introduces an improvement of ANN by utilizing Genetic Algorithms. Then we present chaotic expressions which are employed to enrich the temporal characteristics of time series data. Finally, a hybrid approach is proposed according to the concept of a mixture of experts. Some algorithms used in the hybrid approach, such as Dynamic Time Warping, K-means, and DBSCAN, are also presented in detail.

Contents

3.1 Enhancing Back-Propagation with Genetic Algorithms . . . 33
  3.1.1 Genetic Algorithm ........................................ 34
  3.1.2 Binary Encoding ........................................... 34
  3.1.3 A Hybrid of Back-Propagation and Genetic Algorithm . . 35
3.2 Chaotic expressions ........................................... 36
  3.2.1 Data Pre-processing ....................................... 37
  3.2.2 Reconstructing Phase Space by Chaotic Expressions . . . 38
  3.2.3 False-nearest-neighbor method ........................... 38
  3.2.4 Minimum delay time ....................................... 39
3.3 Mixture of Recurrent Fuzzy Neural Networks ................. 39
  3.3.1 Architecture ............................................... 39
  3.3.2 Clustering Algorithms ................................. 42
  3.3.3 Dynamic Time Warping ................................... 44

3.1 Enhancing Back-Propagation with Genetic Algorithms

The back-propagation algorithm has a big disadvantage in that the training process usually falls into local minima although there is an improvement of momentum technique. One well-known solution to this problem is to train ANNs with an evolutionary algorithm such as Particle Swarm Optimization (PSO) algorithm [Du 2014], Genetic Algorithm [Sarangi 2013], Ant Colony Optimization (ACO) [Shi 2009] and so on. Moreover, ANNs are black boxes for end users. Thus it is hard for them to find the most appropriate combinations of ANN coefficients, e.g., the number
of hidden layers, the number of hidden nodes per layer, the value of learning rate, etc. Typically, the end users must rely on their experience and run the model several times with many different combinations of coefficients. While applying evolutionary algorithms for training ANNs, we can compensate for these disadvantages [Jia 2014, Sarangi 2013, Harpham 2004, Chang 2013, Du 2014, Shi 2009]. Furthermore, in [Branke 1995], the author summarized two popular combinations of ANNs and evolutionary algorithms that are meant not only to train ANNs, but also to design the structure of ANNs via an evolutionary algorithm.

However, one of the drawbacks of evolutionary algorithms is their running time. Due to the searching strategy of the evolutionary algorithms which is based on randomness (trial-and-error), the time consumption is very high. However, in this research, a hybrid of RFNN and GA will be applied for predicting long-term monthly river runoff. Thus the running time is not as important as if it is being compared with performance criteria.

### 3.1.1 Genetic Algorithm

Genetic Algorithm is an adaptive heuristic search algorithm based on the evolutionary ideas of natural selection and genetics. Genetic Algorithm was invented by John Holland in the early 1970s [Holland 1973]. GA is designed to simulate processes in the evolution of natural systems, particularly following the principles proposed by Charles Darwin of "survival of the fittest". That is, in nature, competition among individuals for scanty resources results in the fittest individuals dominating the weaker ones.

A typical Genetic Algorithm consists of three stages: 1) initial population generation: the genetic algorithm generates a set of chromosomes (individuals) called the first generation; 2) computing the fitness of every individual and 3) construction of a new generation in which the Genetic Algorithm establishes the next generation by performing three evolutionary operators: selection, crossover, and mutation. The Genetic Algorithm coefficients are population sizes, mating and mutation rates, and the numbers of generations. In order to combine GA with RFNN, three questions must be answered: 1) how to encode a RFNN individual (phenotype) as a chromosome (genotype); 2) how to execute evolutionary operators, such as selection, crossover, and mutation between two next generations; and 3) what fitness function is chosen. In this research, we employ a binary encoding algorithm called GENITOR to encode a RFNN individual as a chromosome [Whitley 1989, Whitley 1990]. The GENITOR algorithm is very popular because it is quite easy to understand and easy to implement. For the fitness function, we use the Sum Square Error (SSE).

### 3.1.2 Binary Encoding

As mentioned above, we employ the GENITOR algorithm to encode a RFNN individual as a chromosome. This straightforward genotype representation is simply meant to concatenate all the network's weights in a bit-string. To convert all weights
3.1. Enhancing Back-Propagation with Genetic Algorithms

(parameters) of a RFNN into a bit-string, Gray code [Gray 1947] is employed. Let call \( e_{\text{enum}} \) is the encoding function that converts a weight of RFNN to a \( b \)-bit-string namely \( S^b \) and \( e_{\text{enum}} \) is defined as follows.

\[
e_{\text{enum}} : [-w, w] \rightarrow S^b, e_{\text{enum}}(x) = \phi(x),
\]

where \( \phi : [-w, w] \rightarrow Z^b_2, \phi(x) = \lfloor (2^b - 1) \frac{x + w}{2w} \rfloor \). \hfill (3.1)

Each RFNN has \( N \) input nodes, \( M \) fuzzy rules and \( P \) output nodes, thus RFNN weights consists of \( N \times M \) of \( m_{ij} \), \( N \times M \) of \( \sigma_{ij} \), \( N \times M \) of \( \theta_{ij} \), and \( M \times P \) of \( w_{jk} \).

Denote \( e_{\text{net}} \) is the encoding function that converts a RFNN to a bit-string (chromosome) and \( e_{\text{net}} \) is defined as follows.

\[
e_{\text{net}} : R^{N \times M} \times R^{N \times M} \times R^{N \times M} \times R^{M \times P} \rightarrow S^l,
\]

where \( l = (n \times m) + (n \times m) + (n \times m) + (m \times p) \). \hfill (3.2)

\[
e_{\text{net}}(m, \sigma, \theta, w) = \bigotimes_{i=1}^{N} \bigotimes_{j=1}^{M} e_{\text{enum}}(m_{ij}) \otimes \bigotimes_{i=1}^{N} \bigotimes_{j=1}^{M} e_{\text{enum}}(\sigma_{ij}) \otimes \bigotimes_{i=1}^{N} \bigotimes_{j=1}^{M} e_{\text{enum}}(\theta_{ij}) \otimes \bigotimes_{j=1}^{M} \bigotimes_{k=1}^{P} e_{\text{enum}}(w_{jk}). \hfill (3.3)
\]

The biggest drawback of the binary encoding is that, for larger problems, the binary encoding results in very large strings (thousands of bits) which can slow down the evolution process.

3.1.3 A Hybrid of Back-Propagation and Genetic Algorithm

Typically, the combination of BP and GA to train neural networks, can be classified into two opposing strategies [Branke 1995, Sarangi 2013]:

1. BP first is used to train a number of neural networks (NNs), and then GA is employed with little change from generations to generations, to optimize the trained neural networks. This strategy initializes the first generation of GA with a number of trained individuals to expect that the evolution process is only taken place between better individuals makes individuals of generations get better and better and gradually leads to global optima.

2. GA first is used to explore the best NN individual, and then BP is employed to fine-tune the best NN. This strategy is inspired by the idea of which among many good individuals, it is enough to select the best one to train and lead to the global optima.
Algorithm 2: Pseudo-code of RFNN-GA

input: coefficients of RFNN individual structure, coefficients of back propagation and genetic algorithm
output: the best RFNN individual satisfying one of terminating conditions

1 Initialize the generation $G_0$ containing NP of RFNN individuals. Connection weights of every RFNN individual are random in range $[-1 \div 1]$;
2 while terminating conditions are not satisfied do
3 for $j \leftarrow 1$ to NP do
4 Train the $j^{th}$ RFNN by back propagation algorithm;
5 if terminating conditions are satisfied then
6 Break out For loop;
7 end
8 end
9 // Create the next generation $G_{i+1}$ from $G_i$ by applying evolutionary operators
10 if terminating conditions are not satisfied then
11 Selection;
12 Crossover;
13 Mutation;
14 end
15 end

In this research, we propose a hybrid learning algorithm that balances the above mentioned strategies. BP and evolutionary operators are interleaved to execute during the training phase. With this strategy, we hope to utilize the strength of BP to improve the quality of each individual before proceeding with the assessment and evolutionary operators on all individuals. The idea is inspired by the nature of human society in which people should be trained (education, physique, spirit, etc.) to become better humans and to produce better children. The main task of GA in the hybrid method is to expand the search space and not miss any potential areas of optima within it. The hybrid learning algorithm is presented as Algorithm 2.

3.2 Chaotic expressions

The analysis of time series data is an interesting field. Typically, time series data is a sequence of data points that are continuously collected in a long interval. In each even unit of time, one data point is collected. Domains of time series consist of various fields such as industry, environment, finance, hydrology, weather, and so on. Therefore, time series analysis has attracted several researchers who have contributed many valuable and significant achievements in the past few decades. Some important questions are considered when first looking at a time series, such
3.2. Chaotic expressions

as trend analysis, anomaly detection, time series forecasting and prediction, pattern matching, etc. Among these considerations, recent years have witnessed a growing interest in trend analysis, especially time series forecasting and prediction. Forecasting and prediction is a process to estimate what will happen in the future and what should be key decision-making elements of management. The term "forecasting" is reserved for estimation of values at certain specific future times, whereas the term "prediction" is used for more general estimates of values over a long period of time [Kar 2014].

In forecasting and prediction communities, researchers have paid attention to several aspects of forecasting and prediction problems, such as multi-step-ahead forecasting, real-time forecasting, long-term or short-term prediction, and so on. Depending on different specific aspects of time series forecasting and prediction, different methods are proposed—particularly involving time series forecasting and prediction. They range from simple to sophisticated, e.g., linear regression, non-linear regression, fuzzy systems, support vector machines, artificial neural networks, and combinations of these methods.

As mentioned in Section 1.1, in this thesis we consider two practical urgent problems: boiler forecasting and river runoff prediction. A hybrid of fuzzy theory and artificial neural network called RFNN is used as a key element. Due to some drawbacks of RFNN when it is used to predict river runoff, some improvements are proposed in this thesis. In Section 3.1, we discussed how to improve RFNN by utilizing Genetic Algorithms for the objective of long-term river runoff prediction. In this section, to adapt the objective of short-term river runoff prediction, we use techniques including chaotic expressions and mixtures of RFNNs and clustering algorithms.

3.2.1 Dada Pre-processing

In Vietnam, river runoff is typically monitored at several hydrology stations and one runoff value is collected daily. Thus river runoff is a kind of time series data. In order to employ a neural network model like RFNN for predicting river runoff, first, the data must be pre-processed. One main task of data pre-processing is data transformation. Among several transforming methods, the sliding window technique is widely used. To apply the sliding window technique, the original time series, denoted as $X = \{x_1, x_2, \ldots, x_n\}$, will be converted to $Y$ which includes several $Y_i = \{x_i, x_{i+\tau}, \ldots, x_{i+(m-1)\tau}\}, i = 1, 2, \ldots, n - (m - 1)\tau$. In which $\tau$ and $m$ are parameters used for configuration settings, normally $\tau = 1$ and $m$ is set by experience of users. The various values of $\tau$ and $m$ will influence overall performance of RFNN. Unfortunately, it is quite hard to choose the most suitable $m$ if it is just based on the experience of users. Moreover, the disadvantage of this technique is that it is only able to predict in the short term; accumulative errors become larger when predicting in the long term.
3.2.2 Reconstructing Phase Space by Chaotic Expressions

In this research, we employ chaotic theory to convert the time series data into a phase space, namely $Z$ including several $X_i$ and $X_i = (x_i, x_{i+\tau}, \ldots, x_{i+(m-1)\tau})$, $i = 1, 2, \ldots, n - (m - 1)\tau$ in which $\tau$ is delay time and $m$ is embedding dimension. The phase space $Z$ is able to enrich the temporal information of original time series data that will help RFNN learn several latent natural rules of the data faster and more accurately. Many researchers have applied this approach of data transformation and proved its effectiveness [Qian-Li 2008, Benmouiza 2013].

3.2.3 False-nearest-neighbor method

**Algorithm 3:** Pseudo-code of False-nearest-neighbor method

```plaintext
input: The original time series data $X$, the minimum delay time $\tau$, neighbor threshold $R$ and false neighbor threshold $R_{tol}$
output: The optimal embedding dimension $m$

1. $m \leftarrow 1$
2. Build the phase space $Z$ based on $X$, $\tau$ and the current value of $m$
3. foreach $Y_i$ in the new phase space $Z$ do
   4. Identify all nearest neighbors of $Y_i$ called $Y^j_i$ if their square of distance in $m$ dimension phase space $D_m^2(ij) = \sum_{k=0}^{m}(x_{i_k+\tau} - x^j_{i_k+\tau})^2$ is less than neighbor threshold $R$
5. endforeach
6. repeat
7. $m \leftarrow m + 1$
8. Rebuild the phase base $Z$ based on $X$, $\tau$ and the new value of $m$
9. foreach $Y_i$ in the phase space $Z$ do
10.   foreach neighbor $Y^j_i$ of $Y_i$ do
11.      if $\frac{|x_{i+m} - x^j_{i+m}|}{D_m(ij)} > R_{tol}$ then
12.         A false nearest neighbor is found and break to Repeat Loop
13.      end
14.   end
15. end
16. until no false nearest neighbor is found;
```

In [Cellucci 2003], the authors introduced many methods to effectively find embedding dimensions, such as Schuster’s method, Grassberger-Proccacia algorithm (G-P), False-nearest-neighbor method, etc. The authors concluded that G-P is the most popular method but False-nearest-neighbor method is the most effective one.

In False-nearest-neighbor method, we define two points are neighbors of each other if their distance (e.g. Euclid) is smaller than a threshold $R$. Let $y(r)(n)$ be the $r_{th}$ nearest neighbor of $y(n)$ and $m$ be the current dimension of $y(n)$ and $y(r)(n)$. When we increase $m$ by one, the distance of $y(n)$ and $y(r)(n)$ will also be increased and then $y(r)(n)$ may not be a neighbor of $y(n)$ anymore. In this case, we
3.3. Mixture of Recurrent Fuzzy Neural Networks

3.3.1 Architecture

Not all time series are able to generate chaos, so it is necessary to check the chaotic characteristic of the time series after it is converted to the phase space. Until now, the largest Lyapunov index has been a main indicator to judge if a time series dataset has chaotic characteristics or not. In this method, if the largest Lyapunov index is greater than zero, the chaotic characteristic of data can be determined [Rosenstein 1993].

For the task of short-term prediction of the Srepok runoff, we use a sub-dataset of relevant collected data ranging from 2006 to 2010. We calculate $\tau$ and $m$ for the total sub-dataset and receive values 11 and 23, respectively (revisited in Section 4.2.1). According to $m$ and $\tau$ values, the largest Lyapunov index calculated by TISEAN software [Hegger 2007] is -1.8. The value of the Lyapunov index confirms that our data do not have chaotic characteristics. That also means the received phase space cannot be learned by only one RFNN, even if the RFNN is learned very carefully. Meanwhile, we observe a few smaller sub-phase spaces and realize that they have the chaotic characteristics. In other words, the phase space contains several internal temporal rules. To simulate all these temporal rules, we can employ the mixture of experts model. As noted in the comparison of MILE and MELE in Section 2.3.3, MELE is likely more suitable than MILE for this case. Based on MELE principles, in this research we propose a combination method (the so-called Mixture of RFNNs) that consists of RFNNs and a clustering algorithm, as shown in Figure 3.1.
As shown in Figure 3.1, a Mixture of RFNNs has three steps. In the first step, the phase space is explicitly divided into many sub-phase spaces by utilizing a clustering algorithm such as K-means or DBSCAN. In the second step (i.e., the training step) each sub-phase space will be used as the input to train a single RFNN. In the last step (i.e., the testing step), given the input data (prediction input), the RFNN selection engine module compares and chooses the nearest sub-space with the prediction input. Then, the corresponding trained RFNN of the nearest sub-phase space will be used to calculate predicted values. In other words, the selection engine is simply constructed by the arguments of the minimum function of Radial Basis Functions (RBF) as follows.
\[ \pi(x) = \arg\min_i (\phi(i, x)), \]
\[ \pi(x) = \arg\min_{i \in \{1, 2, \ldots, k\}} (\|x - m_i\|), \quad (3.5) \]

where \( x \) is a given input data, \( m_i \) is mean values of the objects in \( i^{th} \) cluster and \( k \) is the number of clusters.

The model of mixture of RFNNs works in such a way that we can consider it as a kind of ensemble learning—in particular, a bagging model.

In the approach, the clustering algorithms are generally mentioned. However, it is hard to determine which clustering algorithm is the most suitable one. Due to some empirical and theoretical analysis, we realize that the K-means algorithm utilized in the model, consists of some drawbacks, caused by three reasons. First, we employ a K-means algorithm to divide the phase space into \( K \) sub-phase spaces, but it is difficult to choose the most appropriate \( K \) for K-means algorithm via manual or experience. Second, if we employ Euclid distance to measure the distance of two sequences, it will cause a problem: two sequences that have the same shape but different time or speed (out of phase), will have a larger distance. Through manual analysis, we observe that several sequences of the phase space are out of phase, as shown in Fig. 3.2. Therefore, several sequences will be clustered into an incorrect sub-phase space if we utilize the Euclid measure to calculate the distance of two sequences. Third, in the phase space, a few outliers occur in some events of natural abnormalities, such as storms, droughts, landslides and so on. These outliers cannot be detected by K-means, and thus the sub-phase spaces consist of plenty of noises. These three reasons raise two issues regarding the Mixture of RFNNs in which K-means and Euclid are utilized. The first issue is that it is difficult to train
corresponding RFNNs of sub-phase spaces containing many noises. Secondly, the RFNN selection engine may give an incorrect selection that corresponds with the nearest incorrect sub-phase space of a specific prediction input.

Thus, in this research, we not only utilize K-means but also DBSCAN. Through experiments, we attempt to verify the hypothesis. Moreover, distance measures are as important as the analysis above. Therefore, two distance measures are applied: Euclid distance and Dynamic Time Warping (DTW).

We also want to find out which combination of RFNNs, clustering algorithms, and distance measures is the most effective and appropriate for further real applications. In Section 4.2.1, the experimental results of these combinations are presented. We name some acronyms of our combinations, which are revisited in Section 4.2.1, as follows:

- RFNN: RFNN combining with sliding window technique.
- RFNN-KM-DTW: RFNN combining with K-means and DTW.
- RFNN-DB-DTW: RFNN combining with DBSCAN and DTW.

3.3.2 Clustering Algorithms

3.3.2.1 K-means Algorithm

Among several clustering algorithms, K-means is the most famous and widely used for clustering. K-means was first proposed by Steinhaus in 1955 [Jain 2010] and has been improved by many other scientists. The K-means algorithm tries to partition the dataset into $k$ clusters, each of which is represented by the mean value of objects in this cluster. The most popular version of K-means is presented as Algorithm 4.

Algorithm 4: K-means Algorithm

| input | the number of clusters $k$ and a dataset $D$ containing $n$ objects |
| output | A set of $k$ clusters |
| 1. Arbitrarily choose $k$ objects from $D$ as the initial cluster centers; |
| 2. repeat |
| 3. (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster; |
| 4. update the cluster mean, i.e, calculate the mean value of the objects for each cluster; |
| 5. until no change; |

The K-means algorithm has been widely applied due to its simplicity. However, it also has many drawbacks:
3.3. Mixture of Recurrent Fuzzy Neural Networks

- The results are hyperspherical clusters. Thus K-means algorithm is not suitable for clusters having non-convex shapes.
- The quality of the clusters is influenced by outliers of the dataset.
- The optimal result is often local optima.
- It is hard to identify the most appropriate value of $k$.

3.3.2.2 DBSCAN Algorithm

**Algorithm 5: DBSCAN Algorithm**

```plaintext
input : $\varepsilon$, $MinPts$ and a dataset $D$ containing $n$ objects
output: density-based clusters and outliers

1 foreach object $P$ in dataset $D$ do
2    if $P$ is visited then
3        continue next object
4    end
5    mark $P$ as visited
6    neighborPts ← $\varepsilon$-neighborhood of $P$
7    if sizeOf(neighborPts) < $MinPts$ then
8        mark $P$ as an outlier
9    end
10   else
11      // $P$ is a core object
12      create a new cluster $C$
13      add $P$ to $C$
14      foreach $P_n$ in neighborPts do
15        if $P_n$ is not visited then
16            mark $P_n$ as visited
17            neighborPts_n ← $\varepsilon$-neighborhood of $P_n$
18            if sizeOf(neighborPts_n) ≥ $MinPts$ then
19                // All objects that are density-connected from
c20                // each other, will be added to the same
21                // cluster
22                neighborPts ← neighborPts ∪ neighborPts_n
23            end
24        end
25        if $P_n$ is not yet a member of any clusters then
26            add $P_n$ to $C$
27        end
28    end
29 end
```
DBSCAN is a density-based clustering algorithm. It can find arbitrarily shaped clusters and is robust to outliers [Ester 1996]. We take this advantage of DBSCAN to identify outliers in our collected data and to achieve qualitative clusters. One can find some extensions of DBSCAN in [Sander 1998], [Georgoulas 2013], [M.Parimala 2011].

We summarize DBSCAN as published in [Ester 1996], which begins with some definitions on a dataset $D$ as follows. Then the DBSCAN algorithm is expressed as Algorithm 5.

**Definition 1:** (ε-neighborhood of an object) The \( \varepsilon \)-neighborhood of an object \( p \) is the set of objects which have distances from \( p \) smaller than \( \varepsilon \).

**Definition 2:** (core object) An object is called a core object if its \( \varepsilon \)-neighborhood has the number of inside objects larger than \( \text{MinPts} \).

**Definition 3:** (directly density-reachable) An object \( q \) is directly density-reachable from an object \( p \) with regards to \( \varepsilon \) and \( \text{MinPts} \) if \( q \) belongs to the \( \varepsilon \)-neighborhood of \( p \) and \( p \) is the core object.

**Definition 4:** (density-reachable) An object \( q \) is density-reachable from an object \( p \) with regards to \( \varepsilon \) and \( \text{MinPts} \) if there is a chain of objects \( p_1, p_2, ..., p_n \) with \( p = p_1, q = p_n \), and \( p_{i+1} \) is directly density-reachable from \( p_i \).

**Definition 5:** (density-connected) An object \( q \) is density-connected from an object \( p \) with regards to \( \varepsilon \) and \( \text{MinPts} \) if there is an object \( o \in D \) and \( p \) and \( q \) are density-reachable from \( o \) with regards to \( \varepsilon \) and \( \text{MinPts} \).

The advantages of DBSCAN are that it does not require the number of clusters as K-means does, and all outliers are completely detected. However, two coefficients of DBSCAN, \( \varepsilon \) and \( \text{MinPts} \), are required; the different values of these coefficients can lead to different quality of results.

### 3.3.3 Dynamic Time Warping

Dynamic Time Warping (DTW) is a well-known algorithm for measuring the distance between two sequences, and was developed originally for speech recognition [Sakoe 1978]. The main weakness of the naive DTW is that the complexity is \( O(n \times m) \) in which \( n \) and \( m \) are the sizes of two sequences, respectively. Therefore, for some applications, especially with big data, the disadvantage of DTW is obvious. In recent years, several improvements to DTW have been proposed. Most of them, e.g., SparseDTW [Al-Naymat 2009], FastDTW [Salvador 2004], and LB_Keogh [Keogh 2005], focused on speeding up the algorithm presented in [Senin 2008, Wang 2013]. In this research, FastDTW is employed to measure the distance between two sequences. FastDTW is a multilevel approach including three key operations [Salvador 2004]: i) Coarsening: Shrink a sequence to a smaller sequence which has nearly the same curve but has fewer points; ii) Projection: Find a minimum-distance warp path at the lowest resolution, and use this warp path as an initial point to end minimum-distance warp path at a higher resolution; and iii) Refinement: Fine-tune the warp path for a higher resolution from a lower resolution through local adjustments of the warp path. A main advantage of FastDTW is that
it is an $O(n)$ algorithm, where $n$ is the length of two sequences.
Chapter 4

RFNN and River Runoff Prediction

Chapter 3 begins with the context of the challenge of river runoff prediction. Then we present the experimental results of two scenarios consisting of short-term prediction and long-term prediction. Finally, we conclude by discussing some findings from the experiments.

Contents

| 4.1 Context | 47 |
| 4.1.1 Study Area | 47 |
| 4.1.2 Dataset | 49 |
| 4.2 Experimental Results | 49 |
| 4.2.1 Short-term Prediction | 50 |
| 4.2.2 Long-term Prediction | 54 |
| 4.3 Summary | 59 |

4.1 Context

4.1.1 Study Area

Climate change is one of the greatest challenges for humanity in the 21st century, seriously affecting to economic production, human life, the environment, and other aspects for people in the world in general and Vietnam in particular. Therefore, most countries have set a high priority in their national development plan to accommodate climate change. In this vein, the Vietnamese Prime Minister on December 02, 2008, approved a national target program addressing climate change. Two of its eight important missions are: (i) to consider how climate change affects to economic production and civilian life; and (ii) to determine solutions to address climate change. To meet the call of the national target program, some researchers are investigating solutions regarding accurate prediction of river runoff.
In Vietnam, rivers play a central role in people’s lives and production. The Srepok River is one of the most important rivers (Figure 4.1). The Srepok River basin in the Central Highlands of Vietnam has a total area of about 18,200 km$^2$ and is about 406 km long. The Srepok River is located between latitudes 1° 53’ to 13° 55’ and longitudes 107° 30’ to 108° 45’. The Srepok watershed has a plentiful lake system and evenly distributed lake system. Due to the sloped terrain, the water retention is not good; during the dry season, the small tributaries run out of water and the water levels of several big lakes drop dramatically. Together with the impact of climate change, the recently observed abnormal changes in the basin of the Srepok River has threatened the stability of Vietnam’s water resources. After monitoring the Srepok runoff from 1990 to 2011, we realize that it has been quite erratic and unpredictable. Naturally, the Srepok runoff varies seasonally; it is low in the dry season and high in the wet season. But in some years, the Srepok runoff decreased suddenly in the dry season or increased suddenly in the wet season; these changes have a direct impact on residents of the Srepok basin. The challenge is how to model and predict the Srepok runoff for the benefit of water resource managers and the basin’s population. This research uses many hybrid models of ANN to predict the Srepok runoff at designated hydrologic stations for two scenarios: short-term prediction and long-term prediction.
4.1.2 Dataset
In the Srepok basin, there are several hydrologic stations that operate in the same way. In our research, we use data from a specific station called BUON DON. The data were gathered over 22 years (1990-2011) and include daily climate conditions and runoff data. The data consist of many tuples; each tuple consists of nine fields capturing information of that day as follows: average of temperature, maximum temperature, minimum temperature, average humidity, minimum of humid degree, rain quantity, evaporation per day, number of daylight hours, and runoff. In total, we collected 8030 records of climate and runoff data. The data are used to create the experimental results of our models.

As introduced in Section 1, we attempt to predict the Srepok runoff in two scenarios: short-term and long-term. We use different methods according to the different objectives of each task. For the objective of short-term prediction, we use RFNN and some mixtures of RFNNs (RFNN-KM-Euclid, RFNN-KM-DTW, and RFNN-DB-DTW). For the objective of long-term prediction, we use RFNN, the hybrid of RFNN and Genetic Algorithm (namely RFNN-GA) and a physical-based method called SWAT. For short-term prediction, we use a sub-dataset of the collected dataset; the sub-dataset only contains the Srepok runoff data as one-dimensional time series, whereas we use the entirety of the river runoff data and corresponding climate data for long-term prediction.

4.2 Experimental Results
The performance of all models developed in this research is assessed by using various standard statistical performance evaluation criteria. The statistical measures considered, are coefficient of correlation (R), root mean squared error (RMSE), and mean absolute relative error (MARE) as follows.

\[
R = \frac{\sum_{i=1}^{n}(O_i - \bar{O})(P_i - \bar{P})}{\sqrt{\sum_{i=1}^{n}(O_i - \bar{O})^2} \sqrt{\sum_{i=1}^{n}(P_i - \bar{P})^2}},
\]

\[
RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n}(O_i - P_i)^2},
\]

\[
MARE = \frac{1}{n} \sum_{i=1}^{n} \frac{|O_i - P_i|}{O_i},
\]

where \(O_i\) is the observed runoff at time \(i\); \(\bar{O}\) is the average observed runoff; \(P_i\) is the simulated runoff at time \(i\); \(\bar{P}\) is the average simulated runoff; and \(n\) is the number of observed runoff data.
4.2.1 Short-term Prediction

We use a part of the dataset of the Srepok runoff collected between 2006 to 2010 at a hydrology station called BUON DON for evaluating RFNN, RFNN-KM and RFNN-DB. We divide the dataset into a training sub-dataset and a testing sub-dataset. The training sub-dataset consists of data from years 2006 to 2009 and the testing sub-dataset consists of data in 2010. We use chaotic expressions presented in Section 3.2.2 to calculate $\tau$ and $m$ for the dataset as a whole and receive values 11 and 23, respectively. For the sliding window technique, we choose $m = 15$ because 15 is the half-monthly tidal cycle of most Vietnamese rivers. We also set the number of clusters $k = 15$ for K-means algorithm. After running DBSCAN to cluster the phase space, we receive the number of outliers that contribute approximately 7% the data and 27 different clusters whose sizes are almost different. For those outliers, we employ the k-nearest-neighbors method in which $k$ is set to 10 to calculate the predicted output.

<table>
<thead>
<tr>
<th>Method</th>
<th>Training MARE</th>
<th>Training R</th>
<th>Training RMSE</th>
<th>Testing MARE</th>
<th>Testing R</th>
<th>Testing RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFNN</td>
<td>0.088</td>
<td>0.986</td>
<td>50.209</td>
<td>0.224</td>
<td>0.913</td>
<td>85.541</td>
</tr>
<tr>
<td>RFNN-KM-Euclid</td>
<td>0.049</td>
<td>0.983</td>
<td>34.010</td>
<td>0.163</td>
<td>0.952</td>
<td>76.139</td>
</tr>
<tr>
<td>RFNN-KM-DTW</td>
<td>0.053</td>
<td>0.989</td>
<td>29.192</td>
<td>0.166</td>
<td>0.963</td>
<td>37.454</td>
</tr>
<tr>
<td>RFNN-DB-DTW</td>
<td>0.023</td>
<td>0.991</td>
<td>10.591</td>
<td>0.124</td>
<td>0.977</td>
<td>19.451</td>
</tr>
</tbody>
</table>

The experimental results are summarized in Table 4.1. The results show that the four combinations are different from their performance. That is, RFNN is the least effective method and RFNN-DB-DTW outperforms the others. RFNN-DB-DTW has the smallest values of RMSE and MARE as well as the highest value of R in the training and testing settings. The four methods give predicted values that have closed correlation with observed values. Whereas RMSE values of RFNN and RFNN-KM-Euclid are quite large, RMSE values of RFNN-KM-DTW and RFNN-DB-DTW are small. The good prediction performances of RFNN-KM-DTW and RFNN-DB-DTW indicate that they can be applied to real applications; RFNN-DB-DTW is completely suitable. Particularly, in the testing setting, RFNN-DB-DTW has MARE = 0.124, $R = 0.977$ and RMSE = 19.451, thus it is selected as the best-fit method for predicting the Srepok runoff. Figure 4.13 visualizes the comparison of the performance of the four methods according to MARE, R, and RMSE. Figures 4.3-
4.2. Experimental Results

4.5 show the hydrograph and scatter plots of both the observed and the predicted values obtained by using RFNN, RFNN-KM-Euclid, RFNN-KM-DTW and RFNN-DB-DTW in the testing step, respectively. In experimental settings, we employ FastDTW [Salvador 2004] in place of DTW to measure the distance between two sequences.

![MARE comparison](image1)

![R comparison](image2)

![RMSE comparison](image3)

Figure 4.2: The comparison of the performance of four methods

The plots as shown in Figures 4.3-4.5 confirm again that RFNN-DB-DTW gives predicted values that are closer to the observed values than the rest. Moreover, the performance of RFNN-KM-Euclid and RFNN-KM-DTW are also different. MARE and R are slightly different, whereas RMSE indicator is significantly improved when the mixture of RFNNs method uses DTW instead of Euclid distance. This means that DTW distance contributes positively to the performance of the mixture of RFNNs method. Note that since RFNN-KM-DTW gives a better performance than that of RFNN-KM-Euclid, we do not consider the setting that combines RFNN with
DBSCAN and Euclid distance.

Figure 4.3: The observed runoff and predicted values by RFNN

Figure 4.4: The observed runoff and predicted values by RFNN-KM-Euclid
4.2. Experimental Results

Figure 4.5: The observed runoff and predicted values by RFNN-KM-DTW

Figure 4.6: The observed runoff and predicted values by RFNN-DB-DTW
4.2.2 Long-term Prediction

In the field of environment and resources, people often use physical-based methods, such as the combination of Soil and Water Assessment Tool (SWAT) and GIS techniques, to model current and future change in water resources. Typically, the physical-based models have complex structures and require significant mathematical knowledge. To use these models, people must provide a significant amount of calibration data, and must possess some degree of expertise and experience with the models. In this research, we use SWAT and GIS techniques to simulate the Srepok River runoff. The experimental results of the model are compared with RFNN and a hybrid of RFNN and Genetic Algorithm.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description of parameter</th>
<th>Calibrated value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Fitted value</td>
</tr>
<tr>
<td>CN2</td>
<td>Initial SCS CN II value</td>
<td>-0.17</td>
</tr>
<tr>
<td>ALPHA-BF</td>
<td>Base Flow Alpha factor</td>
<td>0.17</td>
</tr>
<tr>
<td>GW-DELAY</td>
<td>Groundwater delay</td>
<td>160.20</td>
</tr>
<tr>
<td>GWQMN</td>
<td>Threshold water depth in the shallow aquifer for flow</td>
<td>1.26</td>
</tr>
</tbody>
</table>

Table 4.3: Model performance for the simulation of the Srepok runoff

<table>
<thead>
<tr>
<th>Period</th>
<th>Time step</th>
<th>$R^2$</th>
<th>NSI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before calibration</td>
<td>Monthly</td>
<td>0.70</td>
<td>0.41</td>
</tr>
<tr>
<td>Calibration (2004-2008)</td>
<td>Monthly</td>
<td>0.75</td>
<td>0.68</td>
</tr>
<tr>
<td>Before calibration</td>
<td>Monthly</td>
<td>0.82</td>
<td>0.77</td>
</tr>
</tbody>
</table>

According to SWAT features, we have to gather some types of extra data such as soil data and map data, and pre-process these data by using ArcGis software before simulating the Srepok runoff at the BUON DON station. Then we calibrate and validate the SWAT model. Four parameters were chosen to calibrate the model: Curve Number (CN2), Base flow Alpha factor (ALPHA-BF), Groundwa-
4.2. Experimental Results

After Delay (GW-DELAY) and Threshold water depth in the shallow aquifer for flow (GWQMN). The result of calibration is shown in Table 4.2. Then, we used the calibrated result to run the SWAT model again. Consequently, we obtain higher values of NSI. Table 4.3 shows the calibration of the SWAT model in 2004-2008. The fit of the simulated and observed runoff is acceptable because NSI is 0.68 and $R^2$ is 0.75. Finally, we use the parameters obtained from the calibration to validate the model. In the result, the NSI value reaches 0.77 and $R^2$ is 0.82. Figure 4.7 compares the simulated and observed runoff whereas Figure 4.8 presents the degree of correlation between the simulated and observed runoff at the validation phase. If we use the mean absolute relative error (MARE) to assess the model, the MARE of SWAT is quite large, approximately 0.4. Therefore, the results show that the SWAT model is just barely acceptable to simulate the Srepok runoff.

![Diagram comparing observed and simulated runoff](image1)

**Figure 4.7:** Observed runoff and simulated runoff after validation by SWAT

![Diagram showing degree of correlation between observed and simulated runoff](image2)

**Figure 4.8:** The degree of correlation between observed runoff and simulated runoff by SWAT

While reaping experimental results of RFNN and RFNN-GA, we used the dataset of BUON DON Station in the 1990-2008 period for training and in 2009-2011 for testing our models. Due to the seasonal rules of the Srepok runoff, we highlighted and added to the dataset some temporal features such as days per month, months per year, and years. Consequently, the performance of the models is improved. In
addition, we also pruned out some redundant attributes of the dataset that have low correlations with the Srepok runoff, such as maximum temperature, minimum temperature, and minimum humidity degree.

Table 4.4: Structure and performance of RFNN during training and testing phases

<table>
<thead>
<tr>
<th>Fuzzy rules</th>
<th>Epoch</th>
<th>MARE of training phase</th>
<th>MARE of testing phase</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>MARE</td>
</tr>
<tr>
<td>5</td>
<td>200.000</td>
<td>0.8297</td>
<td>0.8474</td>
</tr>
<tr>
<td>10</td>
<td>200.000</td>
<td>0.8755</td>
<td>0.8226</td>
</tr>
<tr>
<td>15</td>
<td>200.000</td>
<td>0.8830</td>
<td>0.8895</td>
</tr>
<tr>
<td>20</td>
<td>200.000</td>
<td>0.8760</td>
<td>0.8311</td>
</tr>
<tr>
<td>25</td>
<td>200.000</td>
<td>0.8770</td>
<td>0.8704</td>
</tr>
<tr>
<td>30</td>
<td>200.000</td>
<td>0.8726</td>
<td>0.8645</td>
</tr>
<tr>
<td>40</td>
<td>200.000</td>
<td>0.8688</td>
<td>0.8608</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td><strong>0.8750</strong></td>
<td><strong>0.8478</strong></td>
</tr>
</tbody>
</table>

Table 4.5: Structure and performance of RFNN-GA during training and testing phases

<table>
<thead>
<tr>
<th>GA coefficients</th>
<th>BP coefficients</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$MARE_1$</td>
</tr>
<tr>
<td>$P$ $G$ $CP$ $MP$ Epoch Fuzzy Rules</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100 50 0.4 0.1 10.000 15</td>
<td></td>
<td>0.8805</td>
</tr>
<tr>
<td>100 50 0.5 0.1 10.000 15</td>
<td></td>
<td>0.8766</td>
</tr>
<tr>
<td>100 50 0.6 0.1 10.000 15</td>
<td></td>
<td>0.8838</td>
</tr>
<tr>
<td>100 50 0.4 0.2 10.000 15</td>
<td></td>
<td>0.8804</td>
</tr>
<tr>
<td>100 50 0.5 0.2 10.000 15</td>
<td></td>
<td>0.8879</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td><strong>0.88184</strong></td>
</tr>
</tbody>
</table>
4.2. Experimental Results

Figure 4.9: Observed runoff and average values of predicted runoff by RFNN in testing phases

\[ y = 0.9685x + 18.291 \]
\[ R^2 = 0.9305 \]

Figure 4.10: The average correlation between observed runoff and predicted runoff by RFNN in testing phases

Figure 4.11: Observed runoff and average values of predicted runoff by RFNN-GA in testing phases
Figure 4.12: The average correlation between observed runoff and predicted runoff by RFNN-GA in testing phases

![Graph showing the correlation between observed and predicted runoff](image)

Figure 4.13: The Srepok river runoff in 2015-2018 predicted by RFNN

![Graph showing Srepok river runoff](image)

Table 4.4 presents the structure and performance of RFNN whereas Table 4.5 presents the structure and performance of RFNN-GA (Note: P is the numbers of populations; G is the numbers of generations; CP is crossover probability values; MP is mutation probability values; MARE_1 is MARE value in training phases; and MARE_2 is MARE value in testing phases). Figure 4.9 and Figure 4.11 present the predicted runoff and the observed runoff of RFNN and RFNN-GA, respectively, during the testing phases. Figure 4.10 and Figure 4.12 present the degree of correlation between the observed runoff and predicted runoff of RFNN and RFNN-GA, respectively, in the testing phases. In the testing phases, MARE of RFNN is about 0.1359 and MARE of RFNN-GA is about 0.1262, whereas MARE of SWAT is about 0.4. Moreover, $R^2$ of SWAT, RFNN and RFNN-GA are 0.82, 0.9305 and 0.9528, respectively. Thus, we can conclude that RFNN and RFNN-GA are superior to SWAT, and that RFNN-GA outperforms RFNN. However, SWAT is a physical-based model based on climate, soil, land use and water resource data. If we have enough and exact data, SWAT is able to simulate and predict river runoff quite well. In Vietnam, because we lack present and future data for the SWAT model, it is hard to predict Srepok runoff with SWAT as expected; however, it is simpler with RFNN
or RFNN-GA. Figure 4.13 shows the results of prediction in years 2015 to 2018 with the previous RFNN model. In this case, the climate data were acquired from SEA-START\(^1\).

### 4.3 Summary

In this chapter, we presented the applications of RFNN and some hybrids of RFFN to predict the Srepok runoff. We divided the prediction into two scenarios: short-term and long-term. For short-term prediction, the experimental results found that the mixture of RFNNs that utilizes DBSCAN and DTW for clustering and distance measuring, respectively, is the superior combination. The performance of RFNN-DB-DTW encourages a further deployment in practice. For short-term prediction, SWAT, RFNN and the hybrid of RFNN and Genetic Algorithm are used. According to the experimental results, RFNN and RFNN-GA clearly outperform SWAT; among the three methods, RFNN-GA is the superior method. As with RFNN-DB-DTW, RFNN-GA can definitely be applied for real deployment. In Section 7, a proposed further deployment of an information system, which plugs in the prediction function, is presented.

---

\(^1\)http://startcc.iwlearn.org
Chapter 3 first introduces the context of the problems of boiler efficiency optimization and two particular problems: boiler efficiency simulation and MSA real-time boiler efficiency forecasting. Then we present some concepts involving the first problem (boiler efficiency simulation). The experimental results of the first problem are also presented. Then, we introduce the second problem (MSA real-time boiler efficiency forecasting). We also present in detail some heuristic algorithms that particularly solve the second problem. Finally, we conclude by describing some findings achieved by the experiments.

Contents

5.1 Context ................................................. 61

5.2 Simulation of Boiler Efficiency ...................... 62
  5.2.1 Background ........................................... 63
  5.2.2 Soft Sensor .......................................... 63
  5.2.3 Simulating Boiler Efficiency by RFNN .......... 64
  5.2.4 Experimental Results ............................... 65

5.3 Forecasting Multi-Step-Ahead Real-Time Boiler Efficiency 67
  5.3.1 MSA Forecasting Strategies ....................... 69
  5.3.2 Hybrid of Stochastic Exploration and RFNN .... 69
  5.3.3 A Reinforcement Learning Algorithm for RFNN .. 71
  5.3.4 Experimental Results ............................... 72

5.4 Summary .................................................. 77

5.1 Context

Critically fierce competition in the modern industrial economy is forcing companies to seek strategies to reduce cost, increase productivity, and improve production efficiency. An effective strategy will maintain company growth, improve business efficiency, and increase profits. When a large quantity of goods are produced, even an increase of just one percent in annual productivity can bring considerable profits.
Chapter 5. RFNN and Boiler Efficiency Optimization

Because Vietnam is an agricultural country, fertilizer plants play an important role in agricultural production. Among many in the country, Phu My Fertilizer Plant\(^1\) is the biggest one. It was established March 28, 2003 and officially began operation on January 19, 2004. The functions of Phu My Fertilizer Plant are to produce and trade urea fertilizer, liquid ammoniac, industrial gas and other chemical products. At the present, Phu My Fertilizer Plant production fulfills roughly 50% of the total urea domestic demand (the total is 2 million tons per year) in Vietnam.

In a fertilizer plant such as Phu My Fertilizer Plant, a boiler called MACCHI is the most important component. Plant managers constantly pay close attention to improving boiler performance or at least keeping boiler performance stable. In this research, we use RFNN and its hybrid methods to optimize the performance of the boiler in Phu My Fertilizer Plant. First, RFNN is applied to simulate boiler efficiency. Second, RFNN and its hybrid methods are applied to forecast multi-step-ahead real-time boiler efficiency.

5.2 Simulation of Boiler Efficiency

Regarding production process optimization, the optimal approaches found in literature can be divided into three categories [Kusiak 2006]:

- Analyzing production and operating models based on thermodynamics and chemistry.
- Applying soft computing methods such as fuzzy systems, evolutionary computing, artificial neural networks, etc., to seek optimal solutions.
- Combining the approaches mentioned above.

The drawback of approaches belonging to the first category is the lack of automatically applying analysis tools to solve complex mathematical formulas, considering many sensitive parameters in a constantly changing environment. Particularly regarding the production-process optimization at Phu My Fertilizer Plant, the construction of precise computational and comprehensive combustion models is sophisticated due to many complex expressions that change under ambient conditions [Li 2004]. Moreover, such constructed models lack consideration of certain characteristics, e.g., input parameters, to address a combustion process that changes over time. As the result, the optimization process based on those constructed models incurs increasing errors in the long run.

The approaches belonging to the second or third categories attempt to develop soft sensors to optimize the production process. A soft sensor, which may be called a virtual sensor, is computer software that collects multiple values of parameters correlated with each other in a particular technological process. Then the soft sensor will mine the correlations between these parameters to derive knowledge that is then exploited to optimize industrial processes as well as forecast a particular problem.

\(^1\) http://www.dpm.vn/en
In this research, we apply many data mining techniques to build a soft sensor that optimizes the performance of the boiler in Phu My Fertilizer Plant. First, we need to introduce some basic concepts relevant to the boiler before we discuss how to use RFNN to simulate that boiler.

5.2.1 Background

**Boiler.** Boilers are machines that transfer the heat of combustion into water to boil the water and turn it into steam. The boilers are vital to plants, because they provide energy for most of the operations. In Phu My Fertilizer Plant, the boiler called MACCHI is a type of "Titanium M" (140 tons/hour) and is suitable for fuel gas combustion. The boiler is associated with a control system having high-reliability measuring instruments in order to monitor automatic operations of the boiler.

**Boiler efficiency.** Precisely calculating the efficiency of the boiler is very important in optimization processes. The efficiency of a boiler is defined as "the percentage of (heat) input energy is used efficiently to generate steam". In literature, there are two methods of assessing boiler efficiency [UNEP 2006]:

- **Direct Method:** As part of the energy obtained from steam compared to the energy of the fuel in the boiler.
- **Indirect Method:** Efficiency is the difference between loss and energy input.

**Boiler operational data.** The control system of the boiler continuously reads and stores the values of input parameters such as fan speed, pressure, steam temperature, load capacity, and so on from the boiler. The operational data consists of parameters’ values, which control and monitor the boiler. Among these parameters, several ones are controllable, e.g., water temperature, air flow, etc., the others are uncontrollable, e.g., ambient temperature and humidity of the air, etc. Some mathematical expressions in which these parameters are variables are defined to calculate the boiler efficiency. The mathematical expressions are very sophisticated but contain a slight margin of error. Another solution that is able to approximate the correlation of the parameters and boiler efficiency is a artificial neural network model, e.g., RFNN.

5.2.2 Soft Sensor

In this research, a soft sensor is developed to collect data automatically from the real boiler and then optimize the boiler based on the knowledge mined from the collected data. The main task of the soft sensor is to keep the boiler efficiency at a stable status. That means the soft sensor continuously detects downtrends of the boiler efficiency and will immediately adjust control parameters to increase its efficiency. To develop some complex modules involving knowledge mining, we employed RFNN
and clustering algorithms. Figure 5.1 presents the architecture of the soft sensor. The soft sensor is named Boiler Efficiency Optimization (BEO) and its working process is described in detail in Appendix A. Among several sophisticated modules of BEO, the Boiler Efficiency Simulation Module is very important; the module is developed by RFNN.

![Figure 5.1: Architecture of BEO](image)

### 5.2.3 Simulating Boiler Efficiency by RFNN

Combustion is considered as a process of time-oriented technology, and the equation of state combustion efficiency is a complex nonlinear equation in which coefficients are not fixed. It is difficult to find exact coefficients of that nonlinear equation. As a result, we need an approximate solution. Neural networks seem to be a simple and effective approximation scheme for this kind of problem. Regarding the BEO soft
5.2. Simulation of Boiler Efficiency

sensor, the boiler with internal reaction equations is monitored as a black box, with input parameters and a corresponding output called the boiler efficiency. Therefore, we need a boiler simulator for simulating the real boiler by knowledge obtained from the collected dataset of the real boiler. The boiler simulator determines the correlation between the input parameters and the boiler efficiency. The collected input parameters consist of air flow, air pressure, water flow, and so on. This boiler is simulated by a multi-variable equation \( y = f(x_1, x_2, ..., x_n) \) in which \( x_i \) are the input parameters, \( y \) is the boiler efficiency, and \( f(\cdot) \) is the suggested model of neural networks. This modeling helps us build a boiler simulation with technological features just like a real boiler. Among many advanced techniques that can excellently approximate the boiler, such as fuzzy systems, support vector machines, etc., RFNN is chosen due to its straightforward idea and easy deployment.

5.2.4 Experimental Results

To simulate the real boiler (MACCHI) with RFNN, we need to provide a complete training dataset to learn the internal reaction equations of the real boiler. Therefore, the boiler’s operational data are collected. Particularly, in each duration of 60s, the BEO soft sensor collects a record consisting of the control parameters and the corresponding boiler efficiency. The dataset is quite big because in one year, we have \( 60 \times 24 \times 365 = 525,600 \) records. Thus over a few years, the data is considerable. For the experiments, we use the dataset collected over the latest six months, since periodic maintenance of the real boiler occurs every six months. In the soft sensor, clustering algorithms such as K-means and Fuzzy C-means are utilized to reduce the size of data and thus speed up the BEO process. Moreover, we also prune out some redundant input parameters by using correlation expressions. The number of input parameters decreases from 44 to 10, each of which has high correlation with the boiler efficiency.

The results of RFNN are assessed by using various standard, statistical performance evaluation criteria. The considered statistical measures are Pearson’s correlation coefficient (R), root mean squared error (RMSE), and mean absolute relative error (MARE).

\[
R = \frac{\sum_{i=1}^{n}((O_i - \overline{O})(P_i - \overline{P}))}{\sqrt{\sum_{i=1}^{n}(O_i - \overline{O})^2} \sqrt{\sum_{i=1}^{n}(P_i - \overline{P})^2}}, \quad (5.1)
\]

\[
RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (O_i - P_i)^2}, \quad (5.2)
\]

\[
MARE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{O_i - P_i}{O_i} \right|, \quad (5.3)
\]

where \( O_i \) is the observed boiler efficiency at time \( i \); \( \overline{O} \) is the average observed boiler efficiency; \( P_i \) is the simulated boiler efficiency at time \( i \); \( \overline{P} \) is the average simulated boiler efficiency and \( n \) is the number of observed data.
Table 5.1 presents the performance of RFNN in the training and testing phase for the entire dataset. In the testing phase, with $R=0.926$, MARE = 2.97E-3 and RMSE = 0.304, RFNN is quite suitable for simulating the real boiler at Phu My Fertilizer Plant. Figure 5.2 and 5.3 illustrate the simulation of boiler efficiency in the last 1000 seconds.

Table 5.1: Performance of RFNN during training and testing phases

<table>
<thead>
<tr>
<th></th>
<th>MARE</th>
<th>R</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>2.04E-03</td>
<td>0.946</td>
<td>0.206</td>
</tr>
<tr>
<td>Testing</td>
<td>2.97E-03</td>
<td>0.926</td>
<td>0.304</td>
</tr>
</tbody>
</table>

Figure 5.2: Observed boiler efficiency and simulated efficiency in testing phase

Figure 5.3: The degree of correlation between observed boiler efficiency and simulated efficiency in testing phase

**Estimated benefits by year of applying BEO.** The benefits of BEO is presented in detail in Appendix A. In this chapter, we only summarize the benefits.
Total steam production of the boiler at Phu My Fertilizer Plant was approximately 600,000 tons in 2013. The cost of the energy to produce a ton of steam is 2.75 MMBTU/T on average (according to the factory’s statistical data). The average of 0.52% in improved boiler efficiency is explained in Table A.3 (refer to Appendix A for detailed explanations).

Table 5.2: Estimated Benefit of BEO

<table>
<thead>
<tr>
<th>Year</th>
<th>Energy to produce one ton of steam without BEO (MMBTU/h)</th>
<th>Improvement %</th>
<th>Energy decreasing (MMBTU/h)</th>
<th>Energy to produce one ton of steam with BEO (MMBTU/h)</th>
<th>Fuel cost (USD/MMBTU) per ton</th>
<th>Benefits per ton (USD)</th>
<th>Total benefits per years (USD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013</td>
<td>2.75</td>
<td>0.52</td>
<td>0.0143</td>
<td>2.7357</td>
<td>6.56</td>
<td>0.09381</td>
<td>56,286.00</td>
</tr>
<tr>
<td>2014</td>
<td>2.75</td>
<td>0.52</td>
<td>0.0143</td>
<td>2.7357</td>
<td>6.69</td>
<td>0.09567</td>
<td>57,402.00</td>
</tr>
</tbody>
</table>

5.3 Forecasting Multi-Step-Ahead Real-Time Boiler Efficiency

Note that the deployment of the soft sensor (BEO) has been divided into two phases: 2013-2014 and 2015-2016. In the first phase, we did not implement the two modules of Anomaly Detection and Multi-Step-Ahead Real-Time Forecasting, as seen in Figure 5.1. In Table A.3, we present the impressive benefits that BEO brought to Phu My Fertilizer Plant in 2013 and 2014, even without the two modules. The benefits confirm that BEO is really effective and provides motivation to continue analyzing its disadvantages. Observing the boiler efficiency, we realize that it is a non-stationary time series dataset that continuously varies. Therefore, it is difficult to detect whether the boiler efficiency is really decreasing in some periods (refer to Appendix A to fully understand the BEO working process). While observing experiments in the first phase, we realized that this problem is a main factor impacting the performance of the soft sensor. When the soft sensor detects that the boiler efficiency is decreasing, it adjusts the control parameters to prevent this. However, if the detection is false, the adjustment is redundant. The problem implies there are some time lags in the detections. In Figure 5.4, we show two different cases of detection of a downtrend in boiler efficiency. In Figure 5.4 a), it is a correct detection; in Fig 5.4 b), it is an incorrect detection. In Fig 5.4 b), after detecting the downtrend, the soft sensor adjusts the control parameters to increase the boiler efficiency at time $t + 1$ and makes the boiler efficiency at time $t + 1$ higher than the boiler efficiency at time $t$. However, if the soft sensor does not adjust the control parameters, the actual boiler efficiency will be higher than this adjustment. In Figure 5.5, we illustrate the improvement due to a MSA forecasting solution. With the
support of this solution, we easily realize which downtrend, such as Figure 5.5 a), is correct.

Moreover, if we apply 2-step-ahead forecasting as seen in Figure 5.5 b), we can avoid many "increasing traps" such as the event happening at time $t + 1$. In Figure 5.5 b), if we observe the duration time from $t - 2$ to $t + 2$, it really is a global downtrend. But if we just observe the duration time from $t - 1$ to $t + 1$, we do not recognize this global downtrend.

![Figure 5.4: Illustration of detection of boiler efficiency's downtrend for two cases a) correct detection and b) incorrect detection](image)

![Figure 5.5: Illustration of detection improvement that is supported by MSA forecasting a) 1-step-ahead forecasting and b) 2-step-ahead forecasting](image)

This practical issue raises the challenge of how to improve real-time MSA forecasting of boiler efficiency. This task involves time series data, so we can employ any traditional forecasting methods such as linear regressions, artificial neural networks (ANNs) and so on. In Section 5.2.4, we proved that RFNN is suitable for simulating the operation of the real boiler, so we can utilize RFNN to solve this challenge. Moreover, the latest observed data also contribute much to the "intelligence" of the
forecasting engine. We can employ a reinforcement learning method to improve the power of RFNN.

5.3.1 MSA Forecasting Strategies

Among many different strategies of MSA forecasting, iterated forecasting and direct forecasting are quite common [Hamzacebia 2009, Sorjamaa 2007]. For the iterated forecasting, one-step-ahead is repeated $n$ times to accomplish the task of $n$-step-ahead forecasting, whereas the direct method tries to forecast directly at time $t + n$. However, both strategies retain a few drawbacks. Due to iterative forecasting with no reinforced learning or supervision in surrounding environments, the iterated forecasting gradually provides more incorrect results when we compare the simulated data with observed data. In other words, after each iteration, the result of forecasting has an error. The error increases exponentially over time; thus the larger $n$ is, the worse the error. In contrast, in direct forecasting, the relationship between the data at time $t$ and $t + n$ is vague and the obscure relationship is explored by employing a regression method such as ANNs. For many practical applications, people usually encounter the problem of how to choose the most appropriate $n$; most solutions are based on experimentation. In [Qian-Li 2008, Shang 2005, Benmouiza 2013], the authors applied some chaotic formulas to enrich the temporal information of observed data. That approach appeared to be a significant improvement addressing the drawbacks of the direct forecasting. However, the disadvantage of the chaotic solution is that the optimal $n$ value is nearly unique for a specific dataset. Due to the goal of MSA forecasting, $n$ must be flexibly varied and the chaotic approach becomes inappropriate.

In spite of the drawbacks of these two strategies, they have been employed to solve many problems of time series prediction and forecasting. For the problem of MSA real-time boiler efficiency forecasting, we try to discover the correlation of control parameters at time $t$ with the boiler efficiency at time $t$ or $t + n$. Hence we are able to utilize the above strategies and combine them with other theories to compose some new methods. Particularly, in this research, we propose two methods - namely, SE-RFNN and RTL-RFNN - based on the idea of direct forecasting. To assess two proposed methods, we compare the performance of the two methods with RFNN. In fact, the two proposed methods are constructed on the basis of the direct forecasting strategy.

5.3.2 Hybrid of Stochastic Exploration and RFNN

In Section 5.2.3, through experiments, we proved that there is a close correlation between control parameters and corresponding boiler efficiency at the same certain time $t$. But it is vague for correlations between the control parameters at time $t$ and the boiler efficiency at time $t + n$. While utilizing the correlation between the control parameters and the boiler efficiency simultaneously to forecast real-time MSA boiler efficiency, we encounter a challenge of forecasting the control parameters at time $t + n$.
from their observation at time $t$. After observing our dataset, we realize that most control parameters have normal contributions (e.g., in Figure 5.6) and they vary in time continuously with no definite temporal rule. Due to the characteristic of the control parameters, we propose a real-time MSA forecasting method, namely SE-RFNN, that is based on stochastic exploration and RFNN. The process of SE-RFNN is presented as follows.

1. Gather a dataset namely $D$ including several $X(t); X(t) = \{x_1(t), x_2(t), ..., x_n(t), y(t)\}$, where $x_i(t)$ is a control parameter of boiler and $y(t)$ is the corresponding efficiency of boiler at time $t$.

2. Divide dataset $D$ into two datasets denoted $D_1$ and $D_2$ that are responsible for two tasks: training task and testing task.

3. Repeat training task for a RFNN with $D_1$ and testing task for this RFNN with $D_2$ until the RFNN satisfies stopping criteria. For training the RFNN, we employ the back-propagation algorithm.

4. At time $t$, to forecast the boiler efficiency at time $t + n$, first we forecast the control parameters with a pseudo-random number sampling method called Box-Muller Transform [Box 1958] based on their contributions. To calculate the mean $\mu$ and the standard deviation $\sigma$ of each control parameter distribution, we do not use total dataset $D$. Depending on $n$ value, a duration $\tau$ whose data is used to calculate $\mu$ and $\sigma$ are different. In the research, $\tau = n * \tau_p$, where $\tau_p$ is a parameter that is set by end users.

5. Apply the random values of control parameters obtained in step 4 for the RFNN to produce the boiler efficiency at time $t + n$.

When testing the method, the $\tau_p$ value obviously impacts the performance of SE-RFNN. To determine the most appropriate value of $\tau_p$, we use experiments and identify $\tau_p = 60$. 

Figure 5.6: Distribution of in-use $N_2$ (%mol)
5.3.3 A Reinforcement Learning Algorithm for RFNN

Based on the R-RTRL NN proposed in [Chen 2013], we would like to introduce RTRL-RFNN derived from replacing recurrent neural networks with recurrent fuzzy neural networks. The principles of the two algorithms are similar. RTRL-RFNN is presented as Algorithm 6 in which the RFNN input is prepared as seen in the steps 1-3 of SE-RFNN above. However, there is a slight change in the dataset $D$ which includes several $X(t); X(t) = \{x_1(t), x_2(t), ..., x_n(t), y(t+n)\}$, where $x_i(t)$ is a control parameter of boiler at time $t$ and $y(t)$ is the efficiency of boiler at time $t+n$. As mentioned above, RTRL-RFNN is based on the principle of the direct forecasting method. Thus, the change in the dataset $D$ aims to explore correlations between the control parameters at time $t$ and the efficiency of boiler at time $t + n$.

**Algorithm 6: Pseudo-code of RTRL-RFNN**

```plaintext
input: number of forecasting steps $n$, one learned RFNN, one its copy denoted $RFNN_{tmp}$ and a data set $D$ that contains observed data until time $t$

output: Output of forecasting at time $t + n$ and the $RFNN$ and $RFNN_{tmp}$ after reinforced learning with observed data at time $t$

for $i \leftarrow (t - n)$ to $t$ do
    $Y(i+n)$ is the output of $X(i)$ produced by $RFNN$;
    if $i = (t - n)$ then
        $E(t) \leftarrow (Y^{(d)}(t) - Y(t));$
        // $Y^{(d)}(t)$ is the observed value at time $t$
        update $RFNN_{tmp}$ by the back-propagation algorithm that is employed for RFNN in training phases with the error $E(t)$;
    else
        $\hat{Y}(i+n)$ is the output of $X(i)$ produced by $RFNN_{tmp}$;
        $\hat{E}(t) \leftarrow (\hat{Y}(i+n) - Y(i+n))$;
        if $i < t$ then
            update $RFNN_{tmp}$ by the back-propagation algorithm with the error $\hat{E}(t)$;
        else
            update $RFNN$ by the back-propagation algorithm with the error $\hat{E}(t)$;
        $Y(t+n)$ is the output of $X(t)$ produced by $RFNN$;
        // the MSA forecasting of $X(t)$ is $Y(t+n)$ that we are looking for
    end
end
```

end
5.3.4 Experimental Results

In Section 5.2.3, we collected data over six months to build up a knowledge base that serves for many different purposes of the soft sensor. Each record of the data consists of many fields that correspond to many control parameters of the boiler such as fan speed, pressure, steam temperature, load capacity, etc. and a corresponding boiler efficiency. Particularly, in each duration of 60s, our system collects a record. In Section 5.2.3, we employed two clustering algorithms, K-means and Fuzzy C-means, as two options to reduce the size of data and speed up our system responsiveness. Due to the short-time forecasting objective, in this research, we only collect data from the most recent week and use it to train RFNN. We do so without compressing by utilizing any clustering algorithms as in the previous research.

Table 5.3: The performance of RFNN, SE-RFNN and RTRL-RFNN during training and testing phases for 1-step-ahead forecasting

<table>
<thead>
<tr>
<th></th>
<th>RFNN</th>
<th>SE-RFNN</th>
<th>RTRL-RFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MARE</td>
<td>R</td>
<td>RMSE</td>
</tr>
<tr>
<td>Training</td>
<td>1.48E-03</td>
<td>0.968</td>
<td>0.156</td>
</tr>
<tr>
<td>Testing</td>
<td>2.91E-03</td>
<td>0.951</td>
<td>0.306</td>
</tr>
</tbody>
</table>

Table 5.4: The performance of RFNN, SE-RFNN and RTRL-RFNN during training and testing phases for 2-step-ahead forecasting

<table>
<thead>
<tr>
<th></th>
<th>RFNN</th>
<th>SE-RFNN</th>
<th>RTRL-RFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MARE</td>
<td>R</td>
<td>RMSE</td>
</tr>
<tr>
<td>Training</td>
<td>1.71E-03</td>
<td>0.958</td>
<td>0.181</td>
</tr>
<tr>
<td>Testing</td>
<td>3.45E-03</td>
<td>0.922</td>
<td>0.351</td>
</tr>
</tbody>
</table>

The results of three methods developed in this research are assessed by using various standard statistical performance evaluation criteria. The considered statistical measures are Pearson’s correlation coefficient (Equation 5.1), root mean squared error (Equation 5.2), and mean absolute relative error (Equation 5.3).

The biggest disadvantage of RFNN is that it is a black box for end users. That means it is hard to choose the most suitable combination of values of RFNN’s coefficients, such as the number of fuzzy rules \( M \), the learning rate \( \eta \), or the momentum value \( \beta \). In the experiment, we use the strategy of "trial and error" which repeats the training phase many times with many different sets of coefficients. Ultimately, we choose the best one in which we set fuzzy rules \( M = 10 \), \( \eta = 0.005 \) and \( \beta = 0.001 \).
5.3. Forecasting Multi-Step-Ahead Real-Time Boiler Efficiency

Figure 5.7: Correlation between observed values and 1SA forecasted values in testing phase by a) RFNN, b) SE-RFNN and c) RTRL-RFNN

Figure 5.8: Correlation between observed values and 2SA forecasted values in testing phase by a) RFNN, b) SE-RFNN and c) RTRL-RFNN

Figure 5.9: Correlation between observed values and 4SA forecasted value in testing phase by a) RFNN, b) SE-RFNN and c) RTRL-RFNN
Table 5.5: The performance of RFNN, SE-RFNN and RT-RL-RFNN during training and testing phases for 4-step-ahead forecasting

<table>
<thead>
<tr>
<th></th>
<th>RFNN</th>
<th>SE-RFNN</th>
<th>RT-RL-RFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARE</td>
<td>2.04E-03</td>
<td>0.936</td>
<td>0.224</td>
</tr>
<tr>
<td>RMSE</td>
<td>0.936</td>
<td>0.224</td>
<td>2.04E-03</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>RFNN</th>
<th>SE-RFNN</th>
<th>RT-RL-RFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MARE</td>
<td>3.68E-03</td>
<td>0.871</td>
<td>0.739</td>
</tr>
<tr>
<td>RMSE</td>
<td>3.13E-03</td>
<td>0.919</td>
<td>0.324</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>RFNN</th>
<th>SE-RFNN</th>
<th>RT-RL-RFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Testing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MARE</td>
<td>3.46E-03</td>
<td>0.937</td>
<td>0.336</td>
</tr>
<tr>
<td>RMSE</td>
<td>3.46E-03</td>
<td>0.937</td>
<td>0.336</td>
</tr>
</tbody>
</table>

Table 5.6: The performance of RFNN, SE-RFNN and RT-RL-RFNN during testing phases for 1SA, 2SA and 4SA forecasting

<table>
<thead>
<tr>
<th></th>
<th>RFNN</th>
<th>SE-RFNN</th>
<th>RT-RL-RFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARE</td>
<td>2.91E-03</td>
<td>3.07E-03</td>
<td>3.38E-03</td>
</tr>
<tr>
<td>R</td>
<td>0.951</td>
<td>0.924</td>
<td>0.950</td>
</tr>
<tr>
<td>RMSE</td>
<td>0.924</td>
<td>0.950</td>
<td>0.306</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>RFNN</th>
<th>SE-RFNN</th>
<th>RT-RL-RFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARE</td>
<td>3.45E-03</td>
<td>3.16E-03</td>
<td>3.39E-03</td>
</tr>
<tr>
<td>R</td>
<td>0.922</td>
<td>0.924</td>
<td>0.949</td>
</tr>
<tr>
<td>RMSE</td>
<td>0.924</td>
<td>0.949</td>
<td>0.351</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>RFNN</th>
<th>SE-RFNN</th>
<th>RT-RL-RFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MARE</td>
<td>3.68E-03</td>
<td>3.13E-03</td>
<td>3.46E-03</td>
</tr>
<tr>
<td>R</td>
<td>0.871</td>
<td>0.919</td>
<td>0.937</td>
</tr>
<tr>
<td>RMSE</td>
<td>0.739</td>
<td>0.937</td>
<td>0.324</td>
</tr>
</tbody>
</table>

The summarized results are presented in Tables 5.3, 5.4, and 5.5. Because instances of RFNN method and RT-RL-RFNN method are the same, all evaluation criteria of both are the same during training phases. The results indicate that three methods can produce acceptable MARE, RMSE and R values and they are nearly equal. That means three methods are appropriate for forecasting real-time MSA boiler efficiency. In testing phases, the observed values and the forecasted values of three methods at 1SA, 2SA and 4SA forecasting are presented in Figures 5.7, 5.8, and 5.9. Figures 5.10, 5.11, and 5.12 visualize the correlation between the observed values and simulated values at 1SA, 2SA and 4SA forecasting.

Observing the evaluation criteria, we easily realize that the forecasting results become poorer when increasing forecasting steps n. In Table 5.6 we have different analyses regarding the performance of three methods with 1SA, 2SA and 4SA forecasting in testing phases. We try to compare the performance of the three methods via the analysis. Table 5.6 shows that all evaluation criteria of SE-RFNN and RT-RL-RFNN are better than RFNN when forecasting step is bigger than 2. For illustrating, the relationships between Pearson’s correlation coefficient (R) and forecasting steps of three methods are presented in Fig. 5.13. It shows that the three methods have different error-rising rates (slopes) when the forecasting step increases from 1SA to 4SA. Figure 5.13 also shows that RFNN has the deepest slope when the forecasting step n increases, whereas SE-RFNN and RT-RL-RFNN are less sen-
sitive to $n$. This can be explained in that the control parameters only have a close correlation with the boiler efficiency at the same time, and with some of the nearest neighbors of this boiler efficiency.

![Graph](image)

Figure 5.10: 1SA boiler efficiency forecasting in testing phase by a) RFNN, b) SE-RFNN and c) RTRL-RFNN

![Graph](image)

Figure 5.11: 2SA boiler efficiency forecasting in testing phase by a) RFNN, b) SE-RFNN and c) RTRL-RFNN
Figure 5.12: 4SA boiler efficiency forecasting in testing phase by a) RFNN, b) SE-RFNN and c) RTRL-RFNN

Figure 5.13: Relationship between Pearson’s correlation coefficient (R) and forecasting steps of three methods

Although MARE and RMSE of SE-RFNN perform RTRL-RFNN, some forecasted values seem to be anomalies because of a few incorrect random values of
control parameters; R values of SE-RFNN are somewhat poorer than RTRL-RFNN. So we can conclude that SE-RFNN and RTRL-RFNN have the same performance if evaluating two methods is based on total evaluation criteria.

5.4 Summary

In this chapter, we present the applications of RFNN and some hybrids of RFFN to build a soft sensor called BEO for Phu My Fertilizer Plant. We propose using RFNN and its hybrid methods, such as RFNN-SE and RTRL-RFNN to implement two important modules of the soft sensor: Boiler Efficiency Simulation and Multi-Step-Ahead Real-Time Forecasting. We deployed the soft sensor without MSA Real-Time Forecasting Module in 2013-2014; the benefits brought by the soft sensor to Phu My Fertilizer Plant was approximately 55,000 USD per year. At the present time, considering the remarkable experimental results of the MSA Real-Time Forecasting Module, the new version of BEO is a marked improvement. However, it is necessary to verify these new and improved benefits of BEO by deploying it at Phu My Fertilizer Plant. Because of the strict policy of the plant, we are waiting for a suitable time to deploy and assess BEO.
In Chapter 6, we describe a few recent works relating to river runoff prediction and boiler efficiency optimization.

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>River Runoff Prediction</td>
<td>79</td>
</tr>
<tr>
<td>6.2</td>
<td>Boiler Efficiency Optimization</td>
<td>82</td>
</tr>
</tbody>
</table>

### 6.1 River Runoff Prediction

For any country in the world, river runoff prediction is very important for water resources planning and management. In the past few decades, river runoff prediction has been studied by a large number of scientists in many fields, especially computer science [Maier 2010, Maier 2000]. Generally, models that are used to predict river runoff can be classed into the two main groups: physical-based models and data-driven models. Typically, the first approach has a complex structure, and requires rather substantial mathematical knowledge and diverse calibration data. In the environmental field, people often use these kind of models in modeling water resources, including runoff of rivers. In [Quyen 2013, Quyen 2014, Panhalkar 2014, Demirela 2009, Santhi 2001, Zhang 2014] the authors proposed methods using Soil and Water Assessment Tool (SWAT) and GIS techniques to model the current and future changes in water resources. The main disadvantage of these methods is that SWAT requires diverse kinds of data, ranging from climate and water resource data to soil map data. As a result, employing SWAT is costly and time-consuming. Moreover, in Vietnam, much environmental and natural data are not available or are unreliable. Consequently, the use of SWAT for modeling river runoff is only barely acceptable [Quyen 2013, Quyen 2014].

In 2000, Bart Nijssen et al. introduced their experimental results of predicting the discharge from global rivers [Nijssen 2010]. They used Macroscale Hydrological Models (MHMs) to predict runoff from several large rivers in the world. The authors pointed out that the calibration of MHMs is a time-consuming process and thus unfeasible for modeling large basins. The experimental results of MHMs were also poor, with an average relative (monthly) RMSE of 70%. In addition, in [Hopson 2010], Thomas M. Hopson and Peter J. Webster utilized a hydrologic model initialized by NASA and NOAA to forecast 1-to-10-day horizons for major
river basins of Bangladesh. The authors concluded that the accuracy of forecasted results depends on three important factors: the quality of weather predictions, the availability of a range of satellite precipitation products to initialize the modeled watershed states and the incorporation of all data sources used in the model.

In short, most physical-based models require a significant amount of calibration data, and a high degree of expertise and experience; thus it is difficult to utilize these models to accurately predict and forecast river runoff.

In contrast, data-driven models require minimal information, are easy to develop, and have been found to be accurate in various applications of hydrologic prediction [He 2014, Erdal 2013, Piotrowski 2013b, Reggiani 2009, Wu 2008, Maier 2010, Maier 2000]. To date, several advanced data-driven models have been proposed to predict and forecast river runoff, e.g., support vector machines [He 2014, Piotrowski 2013b, Wu 2008], CART models [Erdal 2013], Bayesian models [Erdal 2013] and neural networks [Maier 2000, Maier 2010]. Among those data-driven models, neural network model has proved its strength in simulating the sophisticated relationships of many different kinds of complex data, especially hydrologic data.

Although researchers have introduced several novel improvements of neural networks for some specific cases of river runoff prediction [Chen 2013, Razavi 2009, Burchard-Levine 2014, Asadi 2013, Piotrowski 2011, Chau 2006, Piotrowski 2012], it is hard to identify a single neural network model that achieves optimal results on the overall problem domain. In other words, depending on a specific problem, authors will experimentally explore the approximations of the optimal models. For example, in 2010, a study of Lance E. Besaw et al. [Besaw 2010] demonstrated that the use of neural networks could effectively predict stream-flow of the Winoceski River in northern Vermont, U.S.A. Lance E. Besaw et al. pointed out that predictions based on hourly data are more efficient than using daily data because important relationships between climate and the runoff were lost in the latter case.

Moreover, in 2009, Saman Razavi et al. [Razavi 2009] also showed that using temporal neural networks to predict flows in the Karoon basin in south-western Iran proved more effective than traditional methods such as ARMA. In other studies [Alcazar 2008] and [Asadi 2013], the neural network model and its hybrid models were used to simulate and predict runoff in the Ebro River Basin in Spain and the Aghchah River in Iran, respectively. The authors noted that their proposed methods were appropriate for these case studies. In the review of [Maier 2010], the authors highlighted a robustly increasing trend of utilizing neural networks for the prediction of water resources. The authors also noted that in the period of 1999-2010, among 210 high-impact papers focusing on the prediction and forecasting of water resources, 165 papers dealt with river runoff prediction. The case study areas in these papers are located throughout the world, in regions such as Vietnam, the UK, the U.S., Italy, Spain, China, Brazil, etc.

A great many researchers have attempted to tackle challenge of the river runoff prediction by utilizing neural networks. Some efforts were directed towards the comparisons of neural networks with other advanced methods, or hybrid models
of neural networks combined with various theories. In [He 2014], Zhibin He et al. studied and compared three models: ANN, adaptive neuro fuzzy inference system (ANFIS) and support vector machine (SVM) and used them to predict the Heihe River runoff in China. The authors concluded that all three models can be applied successfully to establish accurate and reliable river runoff prediction. In 2014, Muhammad Shoaib, Asaad Y. Shamseldin, and Bruce W. Melville studied wavelet-based neural network models for rainfall-runoff modeling [Shoaib 2014]. Two neural network models, consisting of Multilayer Perceptron Neural Network (MLPNN) and Radial Basis Function Neural Network (RBFNN), were combined with continuous wavelet and discrete wavelet transformation types. Experiments were conducted on the data from the Brosna catchment in Ireland showed that the wavelet transformation can significantly increase the performance of MLPNN and RBFNN. In other studies, Adam P. Piotrowski and Jaroslaw J. Napiorkowski attempted to model the Annapolis River runoff and studied the comparison of methods to avoid over-fitting in neural networks training [Piotrowski 2013a]. Roohollah Noori et al. compared the performance of ANN and principal-component analysis of multivariate linear regression models to predict the Sofichay River runoff [Noori 2000].

A few years ago, one interesting approach was to combine neural networks with evolutionary algorithms such as Genetic Algorithm (GA) [Burchard-Levine 2014], Particle Swarm Optimization (PSO) [Piotrowski 2011, Chau 2006, Piotrowski 2012]. The approach was utilized to successfully tackle various problems, and quickly proved appropriate for river runoff prediction. Most of the experimental results of these publications showed that the hybrids of evolutionary algorithms and neural networks outperform traditional neural networks in predicting and forecasting river runoff. In addition, another interesting technique is chaotic expressions, which are responsible for highlighting temporal characteristics of time series data generally and river runoff data particularly. In [Qian-Li 2008, Shang 2005, Benmouiza 2013], the authors applied chaotic expressions to reconstruct a new kind of chaotic data, namely phase space, from the original time series. Then they used a neural network model and its hybrid models to explore within the phase space.

In summary, to meet the challenges of river runoff prediction, we have a large number of candidate methods: linear regressions, nonlinear regressions, support vector machines, fuzzy systems, neural networks and so on. Although neural networks have been used widely, we cannot conclude that neural network model is the most superior method for the overall problem domain. Depending on a specific problem (with a corresponding specific dataset), we must analyze (manually, empirically, etc.) the main characteristics of the dataset to decide which methods are most suitable. Then we verify this by experiments. In the reviewed works, we realize that most case studies involve rivers with sloping terrain. These rivers often produce runoff data that varies greatly over time, especially seasonally. Furthermore, most of these rivers locate in mountains; consequently, sometimes natural disasters such as storms, droughts, and landslides occur. As a result, the river runoff data have large margins and contain a few anomalies; it is difficult to predict and forecast these noisy data. The Srepok River, which is chosen as a case study in this thesis, also
has a sloping terrain. Therefore, the neural network model and associated hybrid models seem to be suitable methods to predict the Srepok runoff.

**A short survey of Vietnamese research.** In Vietnam, most research which has utilized techniques of computer science to address problems of climate change and hydrology are barely acceptable - not only in theory, but in practice. So far, there have been few significant publications addressing combinations of computer science and hydrology, and most of them are local publications. In 2007, Pham Thi Hoang Nhung, Quang Thuy Ha studied how to use Artificial Neural Network in predicting the runoff of Hoa Binh Lake within 10 days [Nhung 2007]. The authors employed genetic algorithms for learning an ANN to predict the runoff. Nguyen Thanh Son et al. studied the effect of climate change on the water resources of the Nhue River basin [Son 2011]. The authors utilized climate change scenarios combined with the North American Mesoscale (NAM) model. They also conducted some simulations of climate change effects on water resources and drew several significant assessments. In [Xuan 2011], Tran Thanh Xuan studied the impacts of climate change on river runoff and indicated which ones have maximum influence. In [Tinh 2008], the author employed ANNs to predict rainfall and river runoff and thus minimize drought within the river basins in the Central Highland of Vietnam. The results showed that their proposed method was effective in predicting river runoff.

### 6.2 Boiler Efficiency Optimization

Regarding production process optimization at a fertilizer plant such as Phu My, several optimal approaches in literature can be divided into three categories [Kusiak 2006, Song 2007]:

1. Analyzing production and operating models based on thermodynamics and chemistry.

2. Applying soft computing methods such as fuzzy logic, evolutionary computing, artificial neural network, etc to seek optimal solutions.

3. Combining the approaches mentioned above.

The drawback of approaches belonging to the first category is the lack of automatically applying analysis tools to solve complex mathematical formulas with many sensitive parameters in a constantly changing environment. Particularly regarding production process optimization at Phu My Fertilizer Plant, the construction of computational and comprehensive combustion models should be sophisticated due to many complex expressions that change under ambient conditions [Li 2004]. Moreover, such constructed models lack characteristics, considered as input parameters, of a combustion process changing over time. As the result, the optimization process based on those constructed models produce increasing errors over the long term.
6.2. Boiler Efficiency Optimization

Many efforts have been invested into approaches belonging to the second and third categories. Some applications offering functions such as soft sensors have been developed in the literature. A soft sensor, which may be called a virtual sensor, consists of computer software collecting multiple values of parameters correlated with each other in a particular technological process. One can mine the correlation between those parameters to derive knowledge; that knowledge can be exploited to optimize industrial processes as well as forecasting particular problems. Some notable applications including soft sensors are listed as follows.

- The first one was developed by Yokogawa Corporation and has four soft sensors for advanced process control. One of those sensors, namely RQE, applies artificial neural networks in modeling [Japan Cooperation Center 2008].

- The second one was developed by Emerson Corporation, namely DeltaV, and includes virtual sensors employing artificial neural networks (ANN) [Emerson 2009].

- The third was developed by Aspen Technology Inc. The software suite AspenONE Advanced Process Control for Chemicals includes many soft sensors, e.g., Aspen Inferential Qualities (Aspen IQ) applying Partial Least Square PLS, Fuzzy PLS, ANN, and a hybrid of PLS and ANN for data analysis and modeling [Aspen 2008].

In general, one can see a soft sensor as a data mining application combined with specific industrial knowledge to solve forecasting or estimating problems. Data mining techniques, especially neural networks and clustering algorithms, are used relatively frequently. The applications mentioned above have been commercialized and deployed for many years; they have proven their efficiency in terms of economic profits and productivity. Moreover, most of the commercial applications contain the functions of observing and optimizing boiler efficiency (so-called combustion efficiency).

Although the challenges of production process optimization in general, and boiler efficiency optimization in particular, are interesting, there have been few publications utilizing computer science to deal with the challenges. In [Barroso 2003], a simple computer software program was developed to perform simultaneous optimization of the stack and hot air temperatures. The authors showed that the stoichiometric ratio and steam power significantly impact overall boiler efficiency.

In [Kusiak 2006], Andrew Kusiak and Zhe Song presented an application of data mining techniques to optimize boiler efficiency. The authors applied a K-means algorithm to cluster the historical data into a set of control signatures, called a knowledge base. To deal with the real-time optimization of boiler efficiency, they utilized a temporal linear regression model that explores the knowledge base to identify an optimal control setting that can improve the boiler efficiency. A neural network model was used to simulate a real boiler and verify the effectiveness of the
suggested optimal control setting. Without performing live testing, which is expensive and time-consuming, the authors concluded that their approach was effective and significantly increased boiler efficiency. To improve the performance of this approach, in [Song 2007], Andrew Kusiak and Zhe Song applied some methods of variable selection to refine the control parameters that take part in the approach as well as assigning weights to the variables. They also concluded that both proposed methods performed well in terms of improving boiler efficiency.

The temporal linear regression model used in [Kusiak 2006] and [Song 2007] has disadvantages in that it only gives good results for short-term simulation and 1-step-ahead estimation. That means the model can only estimate the maximal boiler efficiency at time $t+1$ from the boiler efficiency at time $t$ and the estimated control parameters at time $t+1$. It is not able to provide a multi-step-ahead (MSA) estimate of the boiler efficiency during run-time. Moreover, the coefficients of this linear regression model are inflexibly updated during run-time, whereas the boiler efficiency is nonlinear and non-stationary. Therefore, to improve the boiler efficiency, it is necessary to build an engine that can not only model the nonlinear and non-stationary characteristics of boiler efficiency, but also estimate (forecast) MSA boiler efficiency in run-time.

When building a forecasting engine, people are often unsure how to choose a suitable strategy of MSA forecasting. Among many different strategies, iterated forecasting and direct forecasting are quite common [Hamzacebia 2009, Sorjamaa 2007]. For iterated forecasting, one-step-ahead forecasting is repeated $n$ times to accomplish the task of $n$-step-ahead forecasting. The direct method attempts to forecast directly at time $t+n$. However, both strategies retain a few drawbacks. Iterative forecasting with no reinforced learning or supervision in the surrounding environments, thus it gradually provides more inaccurate results when we compare the simulated data with observed data. In other words, after each iteration, the forecasting result has an error. The error increases exponentially over time; the larger $n$ is, the worse the error. In contrast, for direct forecasting, the relationship between the data at $t$ and $t+n$ is vague and it is difficult to explore the obscure relationship by employing a regression method such as a neural network model. In many practical applications, people usually encounter a problem of how to choose the most appropriate $n$, and most solutions are based on experimentation. So in [Qian-Li 2008, Shang 2005, Benmouiza 2013], the authors applied some chaotic formulas to enrich the temporal information of observed data; that approach seems to offer a significant improvement regarding the drawback of direct forecasting. However, the major disadvantage of the chaotic solution is that the optimal $n$ value is nearly unique for a specific data. Due to the goal of MSA forecasting, $n$ must be flexibly varied; the chaotic approach becomes inappropriate.

Reinforcement learning (RL) has attracted several researchers and has been applied widely to tackle problems of run-time simulation during the past few decades. In particular, RL has been widely applied to learning artificial neural networks [Chen 2013, Lin 1995, Stanley 2002]. In fact, RL is a broad class of optimal controlling methods based on estimating value functions from experience, simulation,
or search [Sutton 1996]. RL enhances the adaption of control systems based on the latest change in the surrounding environment. Generally, RL can be classified into three categories: dynamic programming, Monte Carlo methods and temporal difference learning that is a combination of the two previous categories [Sutton 1998]. Each method has its advantages and disadvantages. Depending on a specific problem, we can choose the most appropriate method. However, applying RL for ANNs seems to be more suitable for solving control system problems such as robot controlling, vehicle controlling, game programming, etc. than for solving problems of time series forecasting [Busoniu 2008, Gosavi 2008]. In [Chen 2013], the authors introduced a different kind of reinforcement learning for ANNs, namely, a multi-step-ahead (MSA) reinforced real-time recurrent learning algorithm for recurrent neural networks (R-RTRL NN). Through experiments, the authors applied R-RTRL NN for MSA flood forecasting, which is a type of time series forecasting. The authors also noted that the method outperformed two other kinds of ANNs.
7.1 Conclusion

In Vietnam, agriculture is one of the major industries. It recently contributed approximately 15-20% to the national GDP. Rice exports contributed about 1.8 billion USD in 2015. Therefore, problems involving agriculture attract plenty of attention from scientists, managers, and even the government in Vietnam. However, scant significant research, especially regarding applications of computer science to hydrology and fertilizer production, have been deployed successfully into practice during the past few years. Water resources and fertilizer are the most important elements influencing the productivity of rice plants. Thus it is necessary to promote research involving water resources and fertilizer production and apply the results to practice.

Response to this practical demand, in this thesis we study artificial neural networks and related hybrid methods. Then we apply the studies to practical and urgent problems affecting Vietnamese agriculture: river runoff prediction and boiler efficiency optimization. River runoff prediction belongs to the hydrology field, whereas boiler efficiency optimization involves fertilizer production. We attempt to solve these two completely different problems not only in theory but also in practice. One of our solutions has been deployed successfully. Among several viable methods, we choose artificial neural networks as the key one because of the straightforward idea and easy deployment. We addressed some drawbacks of artificial neural networks by combining them with fuzzy systems, evolutionary algorithms (genetic algorithm), chaotic expressions, and clustering algorithms. Depending on the different objectives of sub-problems, various hybrid methods are used. Table 7.1 shows our hybrid methods, their corresponding applications and publications in this thesis.

River Runoff Prediction. We divide the task of river runoff prediction into two cases: short-term and long-term. For short-term prediction, the experimental results show that a mixture of RFNNs that utilizes DBSCAN and DTW for clustering and distance-measuring, respectively, is the best combination. The performance of RFNN-DB-DTW encourages further practical deployment. For short-term prediction, SWAT, RFNN and a hybrid of RFNN and Genetic Algorithm are used. Based on the experimental results, RFNN and RFNN-GA clearly outperformed SWAT; among the three methods, RFNN-GA is the best method. Like RFNN-DB-DTW, RFNN-GA can definitely be applied for practical deployment.
Table 7.1: Statistic of proposed methods, corresponding problems and publications

<table>
<thead>
<tr>
<th>Problems</th>
<th>Sub-Problems</th>
<th>Methods</th>
<th>Publications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Srepok runoff prediction</td>
<td>Short-term prediction</td>
<td>RFNN, RFNN-KM-Euclid</td>
<td>[Duong 2015],</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RFNN-KM-DTW</td>
<td>[Tri 2016],</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RFNN-DB-DTW</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Long-term prediction</td>
<td>RFNN, RFNN-GA, SWAT</td>
<td>[Duong 2014b],</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Duong 2016c],</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Duong 2016a]</td>
</tr>
<tr>
<td>Boiler efficiency optimization</td>
<td>Boiler efficiency</td>
<td>RFNN</td>
<td>[Duong 2014a]</td>
</tr>
<tr>
<td></td>
<td>simulation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MSA real time</td>
<td>RFNN, SE-RFNN, RTRL-RFNN</td>
<td>[Duong 2016b]</td>
</tr>
<tr>
<td></td>
<td>boiler efficiency</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>forecasting</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In Vietnam, there are many large rivers and some of them play a central role in people’s livelihoods and in production, e.g., the MeKong River in southern Vietnam, the Srepok River in the Central Highland of Vietnam, and the Hong River in northern Vietnam. Due to the sloping terrain, the Srepok runoff has large margins; it is very high in the rainy season and very low (almost out of water) in the sunny season. Moreover, there are several natural abnormalities that often occur in the Srepok basin, e.g., storms, droughts, landslides. Therefore, the Srepok runoff contains several anomalies. In contrast, the MeKong basin and the Hong basin are plains (flat terrain) and have few natural abnormalities. Thus Srepok runoff prediction is more difficult than MeKong or Hong runoff prediction.

However, the experimental results of the Srepok runoff prediction indicate that we can solve this problem. In fact, the proposed methods to predict the Srepok runoff can be applied to other rivers such as the MeKong River or the Hong River.

**Boiler Efficiency Optimization.** We used RFNN and some hybrids of RFFN to build a soft sensor called BEO for Phu My Fertilizer Plant. RFNN and associated methods such as RFNN-SE and RTRL-RFNN are proposed to implement two important modules of the soft sensor: Boiler Efficiency Simulation and Multi-Step-Ahead Real-Time Boiler Efficiency Forecasting. We deployed the soft sensor with the MSA Real-Time Boiler Efficiency Forecasting Module in 2013-2014; the soft sensor brought a benefit to Phu My Fertilizer Plant of approximately 55,000 USD per year. The experimental results of the MSA Real-Time Boiler Efficiency Forecasting Module were remarkable, and this module will be plugged into the new version of BEO. However, it is necessary to verify this new benefit of BEO by de-
ploying it at Phu My Fertilizer Plant. Because of the strict policy of the plant, we are waiting for a suitable time to deploy and assess BEO.

7.2 Perspectives

Climate Change and River Runoff Prediction. River runoff prediction does not significant benefit if it is stand-alone. In [Quyen 2013], we proposed an information system for integrating, storing, and analyzing many kinds of data involving climate change in the Srepok basin, e.g., climate data, water resources, soil resources, etc. The data schema of the information system called SRClim is illustrated in Figure 7.1. SRClim was created in 2013 and has been developing ever since. The objective for SRClim is that it must integrate all necessary data involving climate change of the Srepok basin. Furthermore, SRClim must ensure high levels of the data’s availability, security, consistency, analysis, visualization, etc. Therefore, we will plug the function of river runoff prediction into SRClim.

Figure 7.1: The data schema of SRClim

Moreover, SRClim will be extended to connect with all hydrology stations and climate stations via a virtual private network (VPN) that will permit SRClim to collect data automatically and in real time from these stations. In this context, the function of multi-step-ahead real-time river runoff forecasting is also necessary for SRClim; we can utilize SE-RFNN or RTRL-RFNN to implement the function.

In addition, we will verify our proposed methods for other rivers such as the MeKong River or the Hong River. We collected Hong runoff data from 1960 to 2006. Furthermore, we will also research other advanced methods such as deep
learning. Utilizing deep learning, particularly deep belief networks, is appropriate for the task of river runoff prediction (the results were published in [Tri 2016]). In further research, we will conduct more experiments with many settings of the deep learning model, and with many different datasets such as those of the Srepok runoff and the Hong runoff.

**Boiler Efficiency Optimization.** As mentioned above, it is necessary to verify the improved benefit of the new version of BEO by deploying it at Phu My Fertilizer Plant. Because of the strict policy of the plant, we are waiting for a suitable time to deploy and assess BEO. Based on theoretical analysis, we have a strong chance of success. In addition, we need to develop the function of anomaly detection that detects and removes noise (anomalies). It is an important function that we first focused on at the beginning of the project. Due to a dearth of computer science knowledge and experience with boilers, the function was not deployed successfully. Although the anomalies rarely appear, they do impact the overall performance of the soft sensor.

### 7.3 Publications

In conclusion, this thesis addressed some practical and urgent problems in Vietnam by proposing some methods that improve upon artificial neural networks. The experimental results prove that our proposed methods are appropriate for tackling the problems and can be deployed in practice. The proposed methods and their experimental results have been presented and published in high-quality international conferences and journals. The publications are listed as follows.


7.3. Publications


[Quyen 2014] Nguyen Thi Ngoc Quyen, Nguyen Duy Liem and Nguyen Kim Loi. Effect of land use change on water discharge in Srepok watershed, Central


Appendix A

The Soft Sensor - BEO

A.1 Architecture of BEO

Figure A.1 shows our system architecture, namely BEO - Boiler Efficiency Optimization. The system includes several complex modules. We present these modules as follows.
A.1.1 Real-Time Monitoring

This module is an OPC (OLE for Process Control) Client collecting automatically data from OPC Server via a local network. OPC is a software interface standard that allows Windows programs to communicate with industrial hardware devices\(^1\). In every duration, the tool gets parameter's values from OPC Server and the value of duration is defined by the user, usually 60s.

A.1.2 Data Pre-Processing

The module has two main functions: (i) Reading raw operational data from several separate historical text files or receiving real-time data from the real-time monitoring module; then analyzing the data structure and storing the data into a unit database (SQL Server DBMS); (ii) Cleaning the database to make sure it has no errors, outliers, and noisy data. Each record in the database consists of control parameters, load of the boiler, and a corresponding boiler efficiency.

A.1.3 Data Clustering

Since the system works in real-time, the operational data is enormous. Reducing the size of the operational data is significant in speeding up the system to meet its real-time requirement. Data clustering is to group similar records into the same cluster, and derives a knowledge base consisting of the centers of those clusters.

A.1.4 Anomaly Detection

This module is responsible for detecting anomalies of the real boiler during operating. The module is integrated in Pre-Processing data Module to remove outliers. Moreover, the module detects some abnormalities caused by unknown reasons and warns the operators to determine timely solutions.

A.1.5 Efficiency Calculator

The boiler efficiency can be calculated by the boiler simulator module. Typically, boiler simulator only works on M important parameters that are chosen by experts. However, each tuple of the operating data is a set of N parameters and N is usually larger than M. Several formulas can calculate the boiler efficiency from full \(N\) parameters. Unfortunately, these formulas are very complex. Two methods are used in the BEO soft sensor to calculate the boiler efficiency [UNEP 2006].

- **Direct Method**: The boiler efficiency is the ratio of the energy obtained from steam and the energy of the fuel in the boiler.
- **Indirect Method**: The boiler efficiency is the difference between energy loss and energy input.

\(^1\)http://www.opcdatablock.com/WhatIsOPC.html
A.1.6 Boiler Efficiency Simulation

Combustion is considered as a process of time-oriented technology, and the equation of state combustion efficiency is a complex nonlinear equation where coefficients are not fixed. It is difficult to exactly find coefficients of that nonlinear equation. As a result, we need an approximate solution. Neural networks seem to be a simple and effective approximation scheme for this kind of problem. In BEO soft sensor, the boiler with the internal reaction equations is monitored as a black box with control parameters and an corresponding output called the boiler efficiency. Therefore, we need a boiler simulator for simulating the real boiler from the operational data. The boiler simulator determines the correlation between the control parameters and the boiler efficiency. The control parameters consisting of air flow, air pressure, water flow, etc. to operate the boiler. The boiler is simulated by a multi-variable equation $y = f(x_1, x_2, ..., x_n)$, where $x_i$ is a control parameter, $y$ is a boiler efficiency, and $f(.)$ is a suggested model, e.g., neural networks. This modeling of the boiler helps us to build a boiler simulation with technological features like a real boiler. Among many advanced techniques that can approximate the real boiler, such as fuzzy systems, support vector machines, etc., RFNN is chosen due to its straightforward idea and easy deployment.

As presented in detail in Section 5, the Boiler Efficiency Simulation Module is implemented by RFNN with mean absolute relative error (MARE) approximately $2.04E-03$ and $2.97E-03$, in the training phase and the testing phase, respectively.

A.1.7 Boiler Efficiency Optimization

At the time of the boiler efficiency appears downtrends, this module detects in knowledge base and finds a tuple of control parameters that gives higher efficiency than current efficiency. The controllable parameters will be changed by the new values in the found tuple. After suggesting the new parameter values, the Boiler Efficiency Simulation Module predicts the boiler efficiency according to these new values. In the case of the predicting efficiency is lower than the current efficiency; the suggestion will be ignored. Conversely, this module will apply the new found parameters for the boiler with expecting an improved boiler efficiency. Process working of the module is illustrated in Figure A.2, and it has several steps as follows.

- **Step 1:** Reading data from OPC Server via the Real-Tiem Monitoring Module.

- **Step 2:** Finding some similar tuples with the current control parameters that have higher efficiency than the current and have the same load.

- **Step 3:** Checking whether the change of load is higher than a delta value. That the change is not higher a delta value indicates the load is stable and the module can adjust parameters to improve boiler efficiency. Conversely, that the change is higher a delta value indicates the load is not stable, the module ignores and continue monitoring.
• **Step 4:** In the case of stable load, if the new similar tuples can give higher efficiency that is predicted by the Boiler Efficiency Simulation Module, these tuples will be applied to improve the boiler efficiency.

![Flowchart diagram](image)

**Figure A.2:** Real-time optimization work-flow

### A.1.8 Boiler Controller

The same as the Real-Time Monitoring Module, the Boiler Controller Module is an OPC Client controlling the real boiler through OPC Server. According to some adjustments recommended by the Real-Time Boiler Efficiency Optimization Module,
the Boiler Controller Module adjusts the control parameters of the real boiler to improve its efficiency.

**A.1.9 Multi-Step-Ahead Real-Time Forecasting**

As presented in 5, this module is responsible for forecasting the downtrends of boiler efficiency. In the case of down-trend appearances, this module will inform to the Real-Time Boiler Efficiency Optimization Module to proceed the adjustment of control parameters.

**A.2 Benefit of BEO**

We employed a statistical method called statistical inference for two samples [Montgomery 2009] to assess the efficiency of the BEO soft sensor to Phu My Fertilizer Plant. Note that, to calculate confidently, this method requires that the size of samples must be large so that Student’s t-distribution comes close to a normal distribution. We collected the operational data with the duration of 1 sample/60s.

Data used for assessment was collected in 17 days with two separate periods: (i) 1st period: from November 02, 2013 to November 07, 2013, the size of samples is 5892 and the boiler load distributes from 76 ton/h to 83 ton/h. (ii) 2nd period: from November 08, 2013 to November 18, 2013, the size of samples is 8834 and boiler load distributes from 72 ton/h to 84 ton/h. We defined the minimum size of samples is 30 for each boiler load. During the time of collecting data, the boiler operation was in auto-mode and the boiler load is continuous. Therefore, we must rounded many values of boiler load to get one rounded value. For example, all boiler loads from 68.5 ton/h to 69.49 ton/h are rounded to 69 ton/h. After collecting data, we employed the statistical inference method to prove that the BEO soft sensor improve the performance of power consumption.

As mentioned in [Montgomery 2009], Student’s t-distribution is one of normal distribution families. In Student’s t-distribution, mean of $n$ observed data is estimated as below:

$$ \bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \quad (A.1) $$

and standard deviation $\sigma$:

$$ \sigma = \sqrt{\frac{\sum_{i=1}^{n} (x - \bar{x})}{n - 1}} \quad (A.2) $$

We define some factors that are used in Tables A.1 and A.2:

- During running the boiler with BEO, the mean of power consumption is $\bar{x}^{BEO}$.
- During running the boiler with BEO, the standard deviation of power consumption is $\sigma^{BEO}$. 
• During running the boiler without BEO, the mean of power consumption is $\bar{x}_\text{noBEO}$.

• During running the boiler without BEO, the standard deviation of power consumption is $\sigma_x^{\text{noBEO}}$.

Table A.1: Data of improvement of power consumption from November 02, 2013 to November 07, 2013

<table>
<thead>
<tr>
<th>Load</th>
<th>With BEO</th>
<th>Without BEO</th>
<th>$x_BEO$</th>
<th>$x^{\text{noBEO}}$</th>
<th>$\sigma_x^{\text{BE}O}$</th>
<th>$\sigma_x^{\text{noBEO}}$</th>
<th>Confidence %</th>
<th>Quantity of Improvement %</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>2.73603</td>
<td>2.83275</td>
<td>0.00683</td>
<td>0.00685</td>
<td>100.00</td>
<td>3.41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>2.73761</td>
<td>2.77537</td>
<td>0.00695</td>
<td>0.00695</td>
<td>100.00</td>
<td>1.36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>2.72608</td>
<td>2.74504</td>
<td>0.00213</td>
<td>0.00213</td>
<td>99.99</td>
<td>0.66</td>
<td></td>
<td></td>
</tr>
<tr>
<td>79</td>
<td>2.73282</td>
<td>2.78055</td>
<td>0.00101</td>
<td>0.00101</td>
<td>99.98</td>
<td>0.23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>2.73673</td>
<td>2.74136</td>
<td>0.00066</td>
<td>0.00066</td>
<td>100.00</td>
<td>0.17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>2.73244</td>
<td>2.74662</td>
<td>0.00077</td>
<td>0.00077</td>
<td>100.00</td>
<td>0.44</td>
<td></td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>2.73281</td>
<td>2.74793</td>
<td>0.00126</td>
<td>0.00126</td>
<td>100.00</td>
<td>0.55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>2.73722</td>
<td>2.74594</td>
<td>0.00240</td>
<td>0.00240</td>
<td>99.59</td>
<td>0.32</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table A.2: Data of improvement of power consumption from November 08, 2013 to November 18, 2013

<table>
<thead>
<tr>
<th>Load</th>
<th>Without BEO</th>
<th>No BEO</th>
<th>$x_BEO$</th>
<th>$x^{\text{noBEO}}$</th>
<th>$\sigma_x^{\text{BE}O}$</th>
<th>$\sigma_x^{\text{noBEO}}$</th>
<th>Confidence %</th>
<th>Quantity of Improvement %</th>
</tr>
</thead>
<tbody>
<tr>
<td>72</td>
<td>2.71527</td>
<td>2.76663</td>
<td>0.00263</td>
<td>0.00263</td>
<td>100.00</td>
<td>1.93</td>
<td></td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>2.72383</td>
<td>2.75074</td>
<td>0.00181</td>
<td>0.00181</td>
<td>100.00</td>
<td>1.16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>2.72477</td>
<td>2.74937</td>
<td>0.00116</td>
<td>0.00116</td>
<td>100.00</td>
<td>0.89</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>2.72900</td>
<td>2.75293</td>
<td>0.00084</td>
<td>0.00084</td>
<td>100.00</td>
<td>0.77</td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>2.73108</td>
<td>2.75340</td>
<td>0.00077</td>
<td>0.00077</td>
<td>100.00</td>
<td>0.81</td>
<td></td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>2.73184</td>
<td>2.74041</td>
<td>0.00083</td>
<td>0.00083</td>
<td>100.00</td>
<td>0.64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>2.73470</td>
<td>2.75310</td>
<td>0.00084</td>
<td>0.00084</td>
<td>100.00</td>
<td>0.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>79</td>
<td>2.74096</td>
<td>2.75391</td>
<td>0.00120</td>
<td>0.00120</td>
<td>100.00</td>
<td>0.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>2.74455</td>
<td>2.74789</td>
<td>0.00177</td>
<td>0.00177</td>
<td>94.18</td>
<td>0.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>2.73922</td>
<td>2.74053</td>
<td>0.00242</td>
<td>0.00242</td>
<td>99.99</td>
<td>0.38</td>
<td></td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>2.73677</td>
<td>2.75183</td>
<td>0.00210</td>
<td>0.00210</td>
<td>100.00</td>
<td>0.55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>2.74342</td>
<td>2.75553</td>
<td>0.00197</td>
<td>0.00197</td>
<td>100.00</td>
<td>0.44</td>
<td></td>
<td></td>
</tr>
<tr>
<td>84</td>
<td>2.74808</td>
<td>2.76777</td>
<td>0.00368</td>
<td>0.00368</td>
<td>100.00</td>
<td>0.71</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure A.3: Plot of improvement of power consumption from November 02, 2013 to November 07, 2013

Figure A.4: Plot of improvement of power consumption from November 08, 2013 to November 18, 2013

**Improvement of power consumption by boiler load.** Table A.1 & Figure A.3 show the improvement of power consumption with BEO and without BEO in the first period. Table A.2 & Figure A.4 show the improvement of power consumption with BEO and without BEO in the second period. The experimental results show that BEO improved the performance of power consumption with confidence larger 95% at nearly all load. At 80 ton/h boiler load, BEO achieves the improvement of the performance of power consumption with confidence 94.18%. The Figures A.3 & A.4 show that BEO achieves good improvement in power consumption for boiler load below 78 ton/h and over 82 ton/h. Total improvement of power consumption in observed time (17 days) can be summarized as follows.

- The power consumption in total with BEO = 23167.32 MMBTU (one million
British Thermal Unit)

- The power consumption in total with BEO (equivalent calculation) = 23288.63 MMBTU

The experimental results show that the power consumption reduced in total. It means that boiler with the support of BEO improved the performance of power consumption approximately 0.52%.

**Estimated benefits by year of applying BEO.** Total steam production of boiler at Phu My Fertilizer Plant is approximately 600,000 tons in 2013. The cost of the energy to produce a ton of steam is 2.75 MMBTU/T in average (according to the statistical data of the factory). The average of benefit for 0.52% improved boiler efficiency can be explained as Table ??.

**Table A.3: Estimated Benefit of BEO**

<table>
<thead>
<tr>
<th>Year</th>
<th>Energy to produce one ton of steam without BEO (MMBTU/h)</th>
<th>Improvement %</th>
<th>Energy decreasing (MMBTU/h)</th>
<th>Energy to produce one ton of steam with BEO (MMBTU/h)</th>
<th>Fuel cost (USD/MMBTU)</th>
<th>Benefits per ton (USD)</th>
<th>Total benefits per years (USD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013</td>
<td>2.75</td>
<td>0.52</td>
<td>0.0143</td>
<td>2.7357</td>
<td>6.56</td>
<td>0.09381</td>
<td>56,286.00</td>
</tr>
<tr>
<td>2014</td>
<td>2.75</td>
<td>0.52</td>
<td>0.0143</td>
<td>2.7357</td>
<td>6.69</td>
<td>0.09567</td>
<td>57,402.00</td>
</tr>
</tbody>
</table>

In conclusion, in reality, the average natural gas price is about 6.56 USD/MMBTU in 2013 and about 6.69 USD/MMBTU in 2014, the total benefit estimated is about 57,000.00 USD per year. The benefits achieved prove the BEO soft sensor is effective for the boiler operation.