TENNIS WINNER PREDICTION BASED ON TIME-SERIES HISTORY WITH NEURAL MODELING

Mr. Amornchai Somboonphokkaphan

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science and Information
Department of Mathematics
Faculty of Science
Chulalongkorn University
Academic Year 2008

การพยากรณ์ผู้ชนะการแข่งขันเทนนิสบนพื้นฐานของเหตุการณ์ในอดีตแบบอนุกรมเวลาโดยใช้ตัวแบบทางประสาท

นายอมรชัย สมบูรณ์โภคภัณฑ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาวิทยาการคอมพิวเตอร์และสารสนเทศ ภาควิชาคณิตศาสตร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2551

Thesis Title	TENNIS WINNER PREDICTION BASED ON TIME-SERIES HISTORY WITH NEURAL MODELING
By	Mr. Amornchai Somboonphokkaphan
Field of Study	Computer Science
Advisor	Suphakant Phimoltares, Ph.D.
Co-Advisor	Professor Chidchanok Lursinsap, Ph.D.

Accepted by the Faculty of Science, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

.... Dean of the Faculty of Science

(Professor Supot Hannongbua, Ph.D.)

THESIS COMMITTEE

.... Chairman

(Assistant Professor Saranya Maneeroj, Ph.D.)

.... Advisor

(Suphakant Phimoltares, Ph.D.)

.... Co-Advisor

(Professor Chidchanok Lursinsap, Ph.D.)

.... External Examiner

(Jakkarin Suksawatchon, Ph.D.)

อมรชัย สมบูรณ์โภคภัณฑ์ : การพยากรณ์ผู้ชนะการแข่งขันเทนนิสบนพื้นฐานของเหตุการณ์ในอดีตแบบอนุกรมเวลาโดยใช้ตัวแบบทางประสาท. (TENNIS WINNER PREDICTION BASED ON TIME-SERIES HISTORY WITH NEURAL MODELING) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ดร. ศุภกานต์ พิมลธเรศ, อ.ที่ปรึกษาวิทยานิพนธ์ร่วม : ศ. ดร. ชิดชนก เหลือสินทรัพย์, 53 หน้า.

เทนนิสเป็นกีฬาที่ได้รับความนิยมสูงเป็นอันดับต้นของโลก มีนักวิจัยจำนวนมากทำการศึกษาแบบจำลองของการทำนายผลการแข่งขันโดยใช้ข้อมูลทางสถิติของนักเทนนิส งานวิจัยนี้ได้นำเสนอวิธีที่มีสมรรถภาพสูงอย่าง Neural Network ในการทำนายผลการแข่งขัน ที่มีการใช้ข้อมูลทางสถิติและข้อมูลสภาพแวดล้อมรวมเข้าด้วยกัน อีกทั้งยังได้เพิ่มข้อมูลประสบการณ์ของนักเทนนิสในช่วงเวลาหนึ่งเข้าไปด้วย ทำให้การทำนายผลการแข่งขันมีความถูกต้องแม่นยำมากยิ่งขึ้น

ภาควิชา คณิตศาสตร์	ลายมือชื่อนิสิต อมรชัย สมบูรณ์โภคภัณฑ์
สาขาวิชาวิทยาการคอมพิวเตอร์และสารสนเทศ	ลายมือชื่ออ.ที่ปรึกษาวิทยานิพนธ์หลัก
ปีการศึกษา 2551	ลายมือชื่ออ.ที่ปรึกษาวิทยานิพนธ์ร่วม

507 36190 23 : MAJOR COMPUTER SCIENCE

KEYWORDS : TENNIS / NEURAL NETWORK / MULTI-LAYER PERCEPTRON / BACK-PROPRAGATION LEARNING ALGORITHM / TIME-SERIES

AMORNCHAI SOMBOONPHOKKAPHAN : TENNIS WINNER PREDICTION BASED ON TIME-SERIES HISTORY WITH NEURAL MODELING. ADVISOR : SUPHAKANT PHIMOLTARES, Ph.D., CO-ADVISOR : PROF. CHIDCHANOK LURSINSAP, Ph.D., 53 pp.

Tennis is one of the most popular sports in the world. Many researchers have studied in tennis model to find out whose player will be the winner of the match by using the statistical data. This paper proposes a powerful technique to predict the winner of the tennis match. The proposed method provides more accurate prediction results by using both of the statistical data and environmental data and concerning experience of players based on Multi-Layer Perceptron (MLP) with back-propagation learning algorithm.

Department : Mathematics	Student's Signature อมรชัย สมบูรณ์โภคภัณฑ์
Field of Study : Computer Science and Information	Advisor's Signature
Academic Year : 2008	Co-Advisor's Signature

ACKNOWLEDGEMENTS

I would like to acknowledge my advisor, Dr. Suphakant Phimoltares and my co-advisor, Professor Chidchanok Lursinsup, at The Advanced Virtual and Intelligent Computing (AVIC) Research Center, for all their great support and patience that tremendously helped me accomplish this thesis. They also suggest the solution for solving many problems occurred while doing an experiment. I would like to thank all my friends, and most importantly, my family for everything that they have supported me.

I dedicate this work to all the people as I mentioned above. Without them, this work will never be done. Throughout this thesis, I had encountered many problems, but they were trivial, compared to all the supports given by these people. The encouragement from them was so great and those impressions will always be remembered.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Nowadays, tennis is one of the most popular sports in the world. In every year, there are four major Grand Slam tennis events which are Australian Open, French Open, US Open and Wimbledon. These four grand slam tournaments are considered to be the most famous tennis tournament in the world. According to the four major grand slams, court surfaces of these tournaments are different; Australian and US Open is played on hard court, French Open is played on clay and Wimbledon is played on grass. Each court surface has its own characteristics and makes difference in speed and bounce of the ball. Clay court has a slower paced ball and a fairly true bounce with more spin. Hard court has a faster paced ball and very true bounce. Grass court has a faster paced ball and more erratic bounce. Moreover, the scoring system of Grand Slam tournament is also different. Typically for both men's and women's matches, the first player with two-sets winning wins the match. Unlikely to the general match, in the Grand Slam Tournaments, the first player who wins three sets wins the match.

Due to the growth of sport betting, predictions are widely used in many kinds of sports, especially tennis. The tennis prediction model is created to evaluate the chance of winning and the expected length of the match that players will face. Most people believe that the first serve person in the set has more advantage than another because most of the games often go like that so the first serve affect to the games' score [1][2]. Additionally, lots of players always make fault in the first serve and do better in the second serve so second serve might affect to the games' score too. Nevertheless, the first serve and the second serve affect to the games' score but there is another thing that might be refuting an advantage of serves, it is strongly returns of serve. Moreover, the surface characteristics also affect to the players, e.g., some players perform better on grass but they may get worse on clay.

The major purpose of this thesis is to perform an advanced tennis model which provides more accurate prediction results by using the statistical data and environmental data based on Multi-Layer Perceptron. In this paper, back-propagation algorithm, a standard algorithm for supervised learning pattern, is used. In order to build the good tennis prediction model, the appropriate input features, which are based on two main types of data: statistical data and environmental data, are selected for the model.

1.1 Problem Formulation

While gathering the data from sources (OnCourt software [3] and internet [4] [5] [6] [7] [8]), it can concluded that the statistical data are separated in two groups. First, the statistical data of all players that announced on the first week of each year. Second, the statistical data of match which announced after match played. But this thesis doesn't use the first statistical data because it may be not complete because it's up-to-date only the beginning of the year. So, this thesis also uses only the second statistical data.

From study the many research papers, the problems are concluded

1.1.1 Out-of-date data problem

Many research papers are interested in the statistical data which is announced by the organizers on the first week of the tournament starts. These data can be used to predict the tennis matches. However, it will be out-of-date except the first round of the tournament because when predict matches in the second round, it didn't include the data in the first round. This problem is one of the factors that make the results of prediction not good as it should be.

1.1.2 Lacking environmental data problem

From the introduction above, each tournament has different court surface and each court has its own characteristics that affect to the match. Hence, it can conclude that the environment is important for the prediction but many research papers didn't mention it.

1.1.3 Lacking trend of player problem

One of statistical data, called Matchfacts, is the rank position of each player that judge by using the results of match played. The matchfacts is the current rank position of player not representing the trend of the player. For example, player A is defeat player B even though the matchfacts of player A is less than player B. This event can happen if player A has a good performance in this year but player B is falling down or injured. This problem may be one of the important factors that will be increasing the prediction results.

1.2 Objectives

This thesis focuses to create a tennis model to enhance the accuracy of tennis prediction, so the objectives are as follows:

1. To solve *out-of-date data problem* by using our solution to summing the statistical data from previous to almost current match like the experience of players.
2. To solve *lacking environmental data problem* by combining the environmental data such as the court surface.
3. To solve *lacking trend of player problem* by incorporate Time-Series to include the statistical data in the short period of time.

To solve *out-of-date data problem* and *lacking environmental data problems*, this thesis uses a process; Managing data, which is run only the first time and stores the results in database. After that, put the results as the input features to progress the prediction.

1.3 Scope of Work

1. Researchers gathering the data only on four grand slam tournaments which are Australian Open, French Open, Wimbledon and US Open.
2. Researchers focusing on the statistical data and environment data.
3. Researchers implement process, called “Managing Data”, to manipulate data before used it as input features.
4. This thesis would like to perform a tennis prediction model which provides more accurate prediction result. The Multi-Layer Perceptron and Time-Series are used in the model.

1.4 Research Methodology

In order to achieve the above objectives, the following tasks will be carried out by means of appropriate theoretical work described below:

1. Study concepts of related technologies
2. Define the statement of the problem
3. Derive a MLP to crate the tennis model
4. Evaluate the proposed model
5. Write the thesis

Table 1.1 : Research methodology time table

No	Tasks	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	Study concepts of related technologies																		
2	Define the statement of the problem																		
3	Derive a MLP to crate the tennis model																		
4	Evaluate the proposed model																		
5	Write the thesis																		

CHAPTER 2

LITERATURE REVIEW

The first tennis model was proposed by Kemeny and Snell [9] which has only one parameter; probability of each player winning a point. Furthermore, Barnett and Clarke [10] proposed the prediction of a match played at the Australian Open 2003 by using Markov chain model set up in Microsoft Excel which has the probability of player A winning a point if player A is serving and the probability of player B winning a point if player B is serving as inputs.

Many research papers are based on the statistics on winning percentage of players on both serving and receiving. To use the statistical data, there are three problems associated with using these statistics as inputs to predict the tennis match. First, some researchers use the statistics to predict all matches in tournament which make the data slightly out of date except the first round. This problem is called out-of-date-data problem. The second problem is lacking environmental data. This paper is called without-environment-data problem. In this paper, to overcome this shortcoming, the individual player's statistics data will combine with a type of given surface. Therefore, Barnett and Clarke [11] covered the first problem by updating the statistics as tournament progress and giving more weights to more recent matches to cover the out of date data problem and manipulating statistics only the percentage of points won on serve and return of serve for each player. There are some recent papers are used TENNISPROB program and Markov Chain Model that will be explained afterwards.

2.1 TENNISPROB

This program calculates the probabilities exactly (not by simulation). This program considers one match between two players A and B. They assume the point which are independent and identically distributed (depends on only who serves). Then, modeling a tennis match between A and B depends on only two parameters: the probability p_a that A win a point on service, and the probability p_b that B win a point on service.

Given these two (fixed) probabilities, given the rules of the tournament, given the score and who serves the current point. It can calculate the probability of winning the current game, the current set and the match. [12]

2.2 Markov Chain Model

A Markov chain is a random process evolving in time in accordance with the transition probabilities of the Markov chain. It has to be made aware of the time element in a Markov chain. Some pictorial representations or diagrams may be helpful to students. Only two visual displays will be discussed in this paper. These visual displays are sample path diagram and transition graph.

A sample path diagram is similar to a tree diagram that is usually taught in an introductory probability course. In this diagram, starting from an initial state all the possible sample paths of the Markov chain are drawn for a small value of the time parameter, n.Usually, n is taken to be four or five. This sample path diagram displays the possible progression of the Markov chain for n steps starting from an initial state. This display may help to clarify to the students the dependent nature of the Markov chain. Students may use this sample path diagram to evaluate the probabilities of occurrence of a particular sample path. Some state and class properties may be apparent from the sample path diagrams. Single and longer string of sample path may be simulated and shown to the students. Another useful visual display is transition graph. This graph is based on the one-step transition probabilities of the Markov chain that are usually displayed as a transition probability matrix. In this type of graph, states are numbered and each state is written, with a small circle around it, on a piece of paper. The states are spread out on the paper so that they are distinguishable and lines may be drawn to link them. Positive transition probability between two states is indicated by a line joining the two states. An arrow is used to indicate the direction of positive transition probability. Once all the lines and arrows are drawn, students may use the graph to help them to partition the state space into equivalence classes, to identify absorbing states, transient states and the periodic behavior of certain state or equivalence class.

CHAPTER 3

THEORETICAL BACKGROUND

3.1 Large Amount of Data

In the past century, human is extracting data by manually. As the information technology in to human life, the information has grown up and the volume of data in modern times has been increasing in size and complexity. It is quite hard for human to extract the data by himself so the technology becomes helpful for collecting, processing, managing and storing the data which is called data mining.

3.2 Data Mining

Data Mining is a process of analyzing the data from large amount of data that different aspect and summarizing the data to be useful information. Technically, Data Mining is also a process of finding correlations or patterns among various fields in the database. Data mining software analyzes correlations and patterns in stored data based on open-ended user queries. Several types of analytical software are available: statistical, machine learning, and neural networks.

3.2.1 Major element of Data Mining

Data mining consists of five major elements:

a) Extracting, transforming, and loading data onto the data warehouse system.
b) Storing and managing the data in a multidimensional database system.
c) Providing the data access to business analysts and information technology professionals.
d) Analyzing the data by application software.
e) Presenting the data in a useful format, such as a graph or table.

3.2.2 Level of analysis

Different levels of analysis are available:

a) Artificial Neural Network: It is a non-linear predictive model which learns from training and resembles biological neural networks in structure which we will explain in section 3.3.

b) Genetic Algorithm: It is an optimization technique that uses process such as natural selection, genetic combination and mutation based on the concepts of natural evolution.

c) Decision Tree: It is a tree structure that represents sets of decisions. These decisions generate rules for the classification of any dataset.

d) Nearest Neighbor method: A technique that classifies each record in a dataset based on a combination of the classes of the *k* record(s) most similar to it in a historical dataset.

e) Rule Induction: The extraction of useful if-then rules from data based on statistical significance.

f) Data Visualization: The visual interpretation of complex relationships in multidimensional data. Graphics tools are used to illustrate data relationships.

3.3 Artificial Neural Network

An Artificial Neural Network (ANN) is machine that is designed to model the way in which the brain perform. The key element of this paradigm is the novel structure of the information processing system. To achieve good performance, it is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems.

A brain has great structure and ability to build up its own rules through what we usually refer to as experience. Indeed, experience is built up over time. Artificial neuron networks seem to be people which learn from the examples. Artificial Neural Network is an adaptive system for which its structure can be changed by using external and internal information owing through the network during the learning process. For the learning models, there are three major types of learning: supervised learning, unsupervised learning, and reinforcement learning. An artificial neural network is configured for a specific application through a learning process such as pattern recognition and classification.

3.4 Advantages of Neural Networks

Neural networks have ability to derive the meaning of complicated or imprecise data. It can be used to extract patterns and detect trends which are too complex to be noticed by either humans or other computer techniques. A trained neural network can be thought of as an "expert" in the category of information and it has been given to analyze. This expert can be used to provide projections that have given new situations of interest and could answer "what if" question[13].

There are many advantages include:

1. Nonlinearity: An ANN can be linear or nonlinear.

2. Input-Output Mapping: A popular paradigm of learning called supervised learning which involves modification of the synaptic weights of a neural network by applying a set of training samples. Each example consists of a unique input signal and a corresponding

desired response. Thus, the network learns from the examples by constructing an input-output mapping for the problem at hand.

3. Adaptivity: An ANN has an ability to learn how to do any tasks based on the data given for training or initial experience. In particular, an ANN can be easily retrained to deal with minor changed in the operating environmental conditions.

4. Self-Organization: An ANN can create its own organization or representation of the information that receives during learning time.

5. Real Time Operation: ANN computations may be carried out in parallel, and special hardware devices are being designed and manufactured which take advantage of this capability.

6. Fault Tolerance via Redundant Information Coding: Partial destruction of a network leads to the corresponding degradation of performance. However, some network capabilities may be retained even with major network damage.

3.5 Neural networks against conventional computers

Neural networks take more different approach to solve problems than conventional computers. Conventional computers use an algorithmic approach for example; in order to solve a problem the computer follows the set of instructions. Unless the specific steps for the computer to be followed, the computer cannot solve the problem which restricts the capability to solve problem of conventional computers. Furthermore, Neuron Networks would be so much more useful because they could do things that we don't exactly know how to do [13].

Neural networks process is similarity to the human brain does. The network is composed of a large number of highly interconnected processing elements (neurons) working in parallel to solve a specific problem. Neural networks learn from the examples which must be selected carefully otherwise useful time is wasted or even worse the network might be functioning incorrectly. The disadvantage is that because the network finds out how to solve the problem by itself, its operation can be unpredictable [13].

On the other hand, conventional computers use a cognitive approach to problem solving; the way to solve problem must be known and stated in small unambiguous instructions. These instructions are converted to a high level language program and then into machine code which the computer can understand. These machines are totally predictable; if anything goes wrong is due to a software or hardware fault.

Neural networks and conventional algorithmic computers are not in competition but complement each other. There are tasks that more suitable to an algorithmic approach like arithmetic operations and more suitable to neural networks. Even more, a large number of tasks, require systems that use a combination of the two approaches (normally a conventional computer is used to supervise the neural network) in order to perform at maximum efficiency.

3.6 Similarities of Human and Artificial Neurons

3.6.1 Human Brain Learning

In the human brain, a neuron collects signals from others through a host of fine structures called *dendrites.* The neuron sends out spikes of electrical activity through a long, thin stand known as an *axon*, which splits into thousands of branches as shown in Figure 3.1. At the end of each branch, a structure called a *synapse as shown in Figure 3.2,* converts the activity from the axon into electrical effects that inhibit or excite activity from the axon into electrical effects that inhibit or excite activity in the connected neurons. When a neuron receives excitatory input that is sufficiently large compared with its inhibitory input, it sends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes [13].

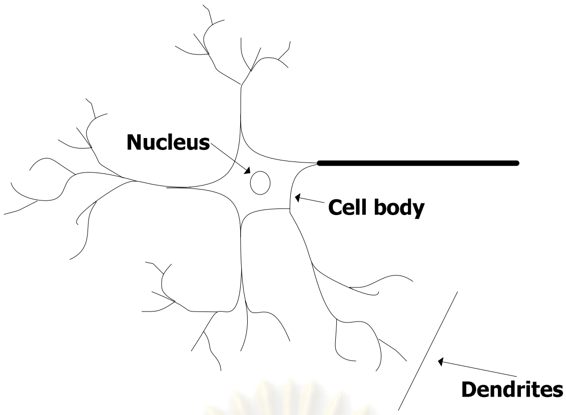

Figure 3.1: Components of a neuron

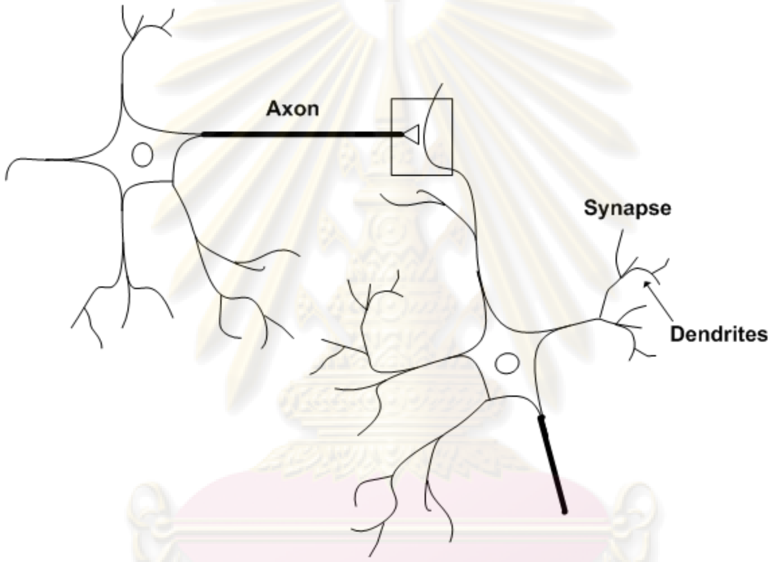

Figure 3.2: The synapse

3.6.2 Artificial Neural

Neural networks are conducted by first trying to deduce the essential features of neurons and their interconnections. We then typically program a computer to simulate these features. However, because our knowledge of neurons is incomplete and our computing power is limited, our models are necessarily gross idealizations of real networks of neurons.

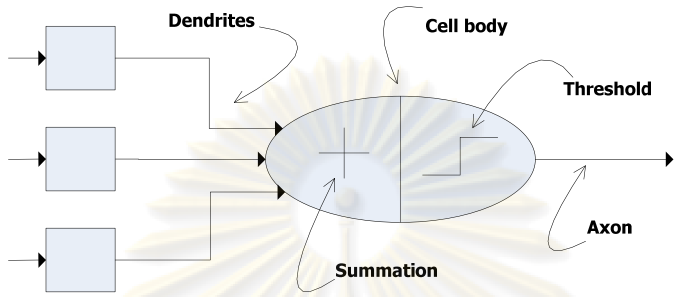

Figure 3.3: The neuron model

3.7 Model of a Neuron

There are three basic elements of the neural model:

1. A set of synapses or connecting links, each of which is characterized by a weight or strength of its own
2. An adder for summing the input signals, weighted by the respective synapses of the neuron; the operations described here constitutes a linear combiner.
3. An activation function for limiting the amplitude of the output of neuron.

3.8 Network Architectures

3.8.1. Single-Layer Feed-forward networks

Feed-forward ANNs allow signals to travel one way only; from input to output. There is no feedback (loops) i.e. the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straight forward networks that associate inputs with outputs. It is illustrated in Figure 3.4 for the case of three nodes in both the input and output layers. Such a network is called a single-layer network, with the designation "single-layer" referring to the output layer of computation nodes. We do not count the input layer of source nodes because no computation is perform there [13].

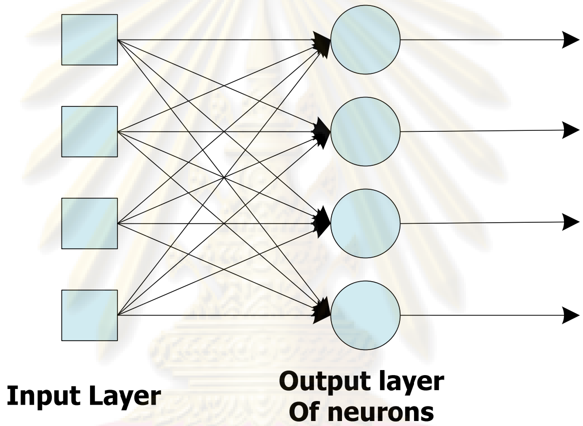

Figure 3.4: The Single-Layer Feed-forward networks

3.8.2. Multi-Layer Feed-forward networks

This feed-forward neural network distinguishes itself by the presence of one or more hidden layers whose computation nodes are correspondingly hidden neurons. The function of hidden neurons is to intervene between the external input and the network output in some useful manner. The neural network in Figure 3.5 is called fully connected in the sense that every node in each layer of the network is connected to every other node in the adjacent forward layer [13].

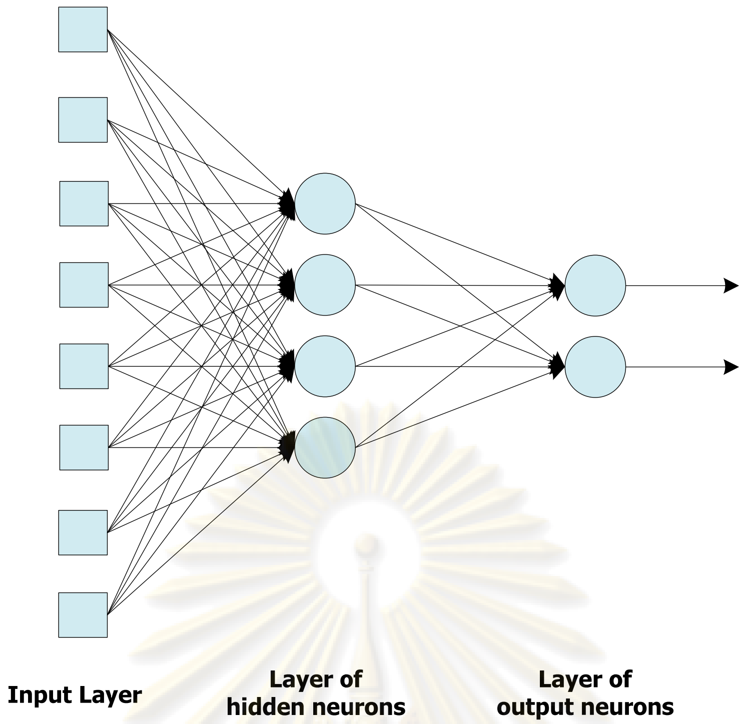

Figure 3.5: The Multi-Layer Feed-forward networks

3.8.3. Recurrent networks

A recurrent neural network distinguishes itself from a feed-forward neural network in that it has at least one feedback loop [13]. A recurrent network may consist of a single layer of neurons with each neuron feeding its output signal back to the inputs of all the other neurons as shown in Figure 3.6.

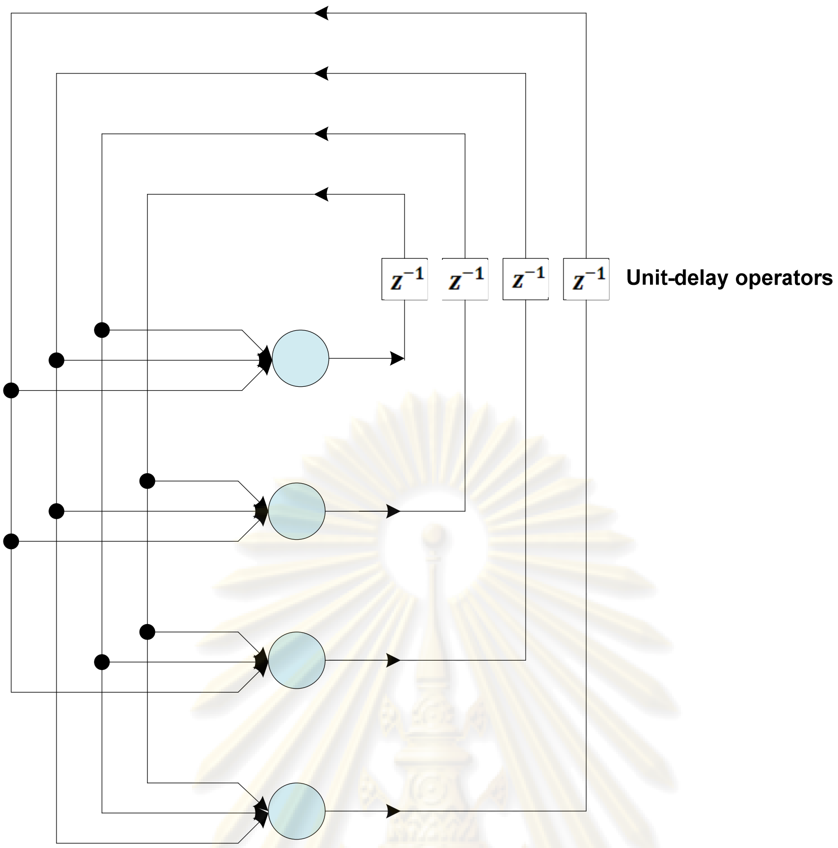

Figure 3.6: The Recurrent networks.

3.9 Perceptrons

The perceptron is the simplest form of a neural network which used linear separable to classify the patterns.

3.9.1 Single-Layer Perceptrons

Single-layer perceptrons compose of a single neuron which adjusts synaptic weights and bias. Single-Layer perceptrons consists of the input layer and an output layer.

3.9.2 Multi-Layer Perceptrons

Multi-Layer perceptrons have been created for solve some difficult and various problems by training in a supervised manner with the error back-propagation algorithm. Multi-Layer perceptrons consists of the input layer, one or more hidden layers of computation nodes, and an output layer.

CHAPTER 4

TENNIS PREDICTION MODEL

4.1 The Multi-Layer Perceptron Neural Network Model

Artificial Neuron Network has three kinds of layers which are input layer, hidden layer, and output layer. As shown in figure 4.1, there are an input layer on the top, hidden layer in the middle and an output layer at the bottom.

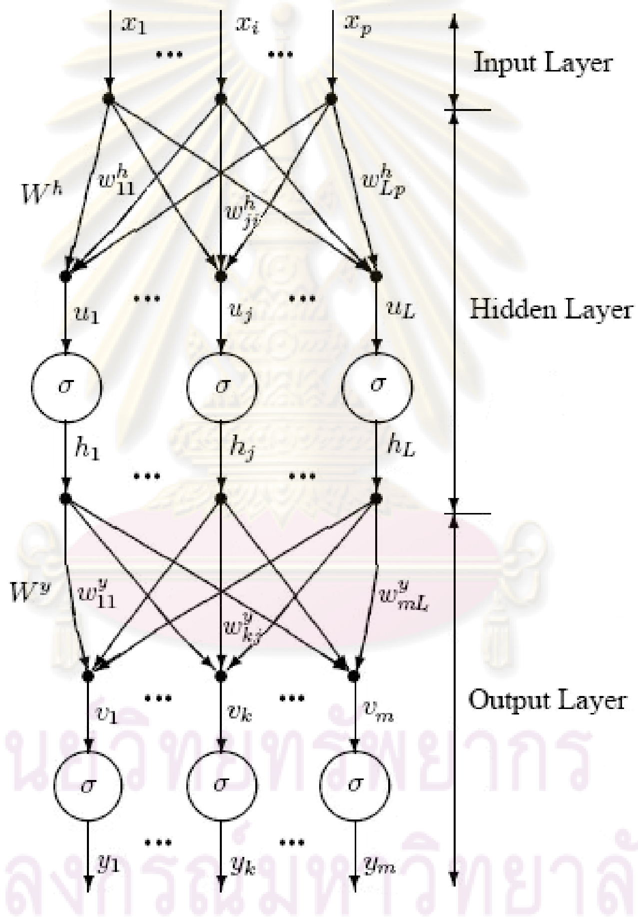

Figure 4.1 : Multi-Layer Perceptron

4.1.1 Input Layer

Input layer is a vector of predictor variable values. Its characteristic is $(x_1 \ \dots \ x_p)$. The input layer (or processing before the input layer) standardizes these values so that the range of each variable is -1 to 1. The input layer distributes the values to each of the neurons in the hidden layer. In addition to the predictor variables, there is a *bias* which is a constant input of 1.0 that is fed to each of the node in hidden layers. The bias is multiplied by a weight and added to the sum going into the neuron.

4.1.2 Hidden Layer

In hidden layer, the value from each input neuron is multiplied by a weight (w_{ji}), and the resulting weighted values are summed together producing a combined value u_j. The weighted sum (u_j) is fed into a transfer function, σ, which outputs a value h_j. The outputs from the hidden layer are distributed to the output layer.

4.1.3 Output Layer

In output layer, the value from each hidden layer neuron is multiplied by a weight (w_{kj}), and the resulting weighted values are added together producing a combined value v_j. The weighted sum (v_j) is fed into a transfer function, σ, which outputs a value v_k. The *y* values are the outputs of the network.

If a regression analysis is being performed with a continuous target variable, there is a single neuron in the output layer, and then it generates a single *y* value. For classification problems with categorical target variables, there are *N* neurons in the output layer producing *N* values, one for each of the *N* categories of the target variable.

4.2 Multi-Layer Perceptron Architecture

As shown in figure 4.1, there is a full-connected neuron network which has three layers. All neural networks have an input layer and an output layer, but the number of hidden layers may vary as figure 4.2.

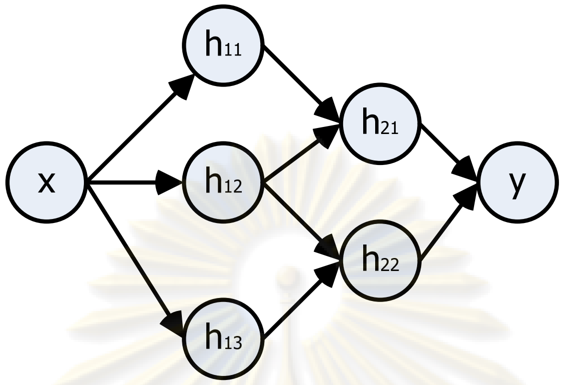

Figure 4.2 : A perceptron network with two hidden layers and four total layers

For multi layer perceptron, there is more than one hidden layer, the output from one hidden layer is fed into the next hidden layer and separate weights are applied to the sum going into each layer.

4.3 Training Multi-Layer Perceptron Networks

The goal of the training process is to find the set of weight values that will cause the output from the neural network that matches the actual target values as closely as possible. There are several issues involved in designing and training a multi-Layer perceptron network:

- Selecting the number of hidden layers to use in the network.
- Deciding the number of neurons to use in each hidden layer.
- Finding a globally optimal solution to avoids local minima.
- Converging to an optimal solution in a reasonable period of time.
- Validating the neural network to test for over fitting.

4.3.1 Selecting the Number of Hidden Layers

Most of problems, the use of one hidden layer is sufficient, at least two hidden layers are required for modeling data with discontinuities such as a saw tooth wave pattern. The use of two hidden layers rarely improves the model, and there is no theoretical reason for using more than two hidden layers.

4.3.2 Deciding the number of neurons to use in the hidden layers

One of the most important characteristics of a perceptron network is the number of neurons in the hidden layer(s). If an inadequate number of neurons are used, the network will be unable to model complex data, and the resulting fit will be poor.

If too many neurons are used, the training time may become excessively long, and might be worse, the network may over fit the data. When over fitting occurs, the network will begin to model random noise in the data. The result is that the model fits the training data extremely well, but it generalizes poorly to new, unseen data. Validation must be used to test for this.

4.3.3 Finding a globally optimal solution

Typically, neural network might have a couple of hundred weighs and the values must be found to produce an optimal solution. If neural networks were linear models like linear regression, it would be a breeze to find the optimal set of weights. But the output of a neural network as a function of the inputs is often highly nonlinear which makes the optimization process complex.

If you plotted the error as a function of the weights, you would likely see a rough surface with many local minima as shown in figure 4.3.

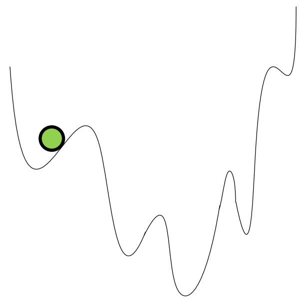

Figure 4.3 : Graph of error.

This figure is highly simplified because it represents only a single weight value (on the horizontal axis). With a typical neural network, you would have a 200-dimension, rough surface with many local valleys.

Optimization methods such as steepest descent and conjugate gradient are highly susceptible to finding local minima if they begin the search in a valley near a local minimum. They have no ability to see the big picture and find the global minimum.

Several methods have been tried to avoid local minima. The simplest way is just to try a number of random starting points and then use the one with the best value. A more sophisticated technique called simulated annealing improves on this by trying widely separated random values and then gradually reducing ("cooling") the random jumps in the hope that the location is getting closer to the global minimum.

4.3.4 Converging to the Optimal Solution — Conjugate Gradient

Most training algorithms follow this cycle to refine the weight values:

a) Running the set of predictor variable values through the network using a tentative set of weights.

b) Computing the difference between the predicted target value and the actual target value.

c) Averaging the error information over the entire set of training cases,

d) Propagating the error backward through the network and compute the gradient (vector of derivatives) of the change in error with respect to changes in weight values,

e) Making adjustments to the weights for reducing the error. Each cycle is called an epoch.

Because error information is propagated backward through the network, this type of training method is called backward propagation.

The back-propagation training algorithm was first described by Rumelhart and McClelland in 1986; it was the first practical method for training neural networks. The original procedure used the gradient descent algorithm to adjust the weights toward convergence using the gradient. Because of this history, the term “back-propagation” or “backprop” often is used to denote a neural network training algorithm using gradient descent as the core algorithm. That is somewhat unfortunate since backward propagation of error information through the network is used by nearly all training algorithms, some of which are much better than gradient descent.

The back-propagation is using gradient descent often converges very slowly or not at all. On large-scale problems its success depends on user-specified learning rate and momentum parameters. There is no automatic way to select these parameters, and if incorrect values are specified the convergence may be exceedingly slow, or it may not converge at all. While back-propagation with gradient descent is still used in many neural network programs, it is no longer considered to be the best or fastest algorithm.

The traditional conjugate gradient algorithm uses the gradient to compute a search direction. It then uses a line search algorithm such as Brent’s Method to find the optimal step size along a line in the search direction. The line search avoids the need to compute the Hessian matrix of second derivatives, but it requires computing the error at multiple points along the line. The conjugate gradient algorithm with line search (CGL) has been used successfully in many neural network programs, and is considered one of the best methods yet invented.

The scaled conjugate gradient algorithm uses a numerical approximation for the second derivatives (Hessian matrix), but it avoids instability by combining the model-trust region approach from the Levenberg-Marquardt algorithm with the conjugate gradient approach. This allows scaled conjugate gradient to compute the optimal step size in the search direction without having to perform the computationally expensive line search used by the traditional conjugate gradient algorithm. Of course, there is a cost involved in estimating the second derivatives.

Tests performed by Moller show the scaled conjugate gradient algorithm converging up to twice as fast as traditional conjugate gradient and up to 20 times as fast as back-propagation using gradient descent. Moller's tests also showed that scaled conjugate gradient failed to converge less often than traditional conjugate gradient or back-propagation using gradient descent.

4.4 Input Features

Most of the researchers concentrate only on the statistical data such as percentage of first serve, winning percentage on the first serve, winning percentage on the second serve which directly affect to the match result.

To reduce the problem of lacking environmental data, the court surface is selected to be one of input features. According to the court surface, it produces an effect to the individual statistic of the player. For example, some players do a better job on grass but some players do not.

In this paper, both statistical data and environmental data are used. The selected statistical features consist of winning percentage on the first serve, winning percentage on the second serve, winning percentage on return serve, winning percentage on break point and total point win. For the environmental data, the court surface is selected to be one of the input features. All input features used as input vector of the MLP can be shown as follows:

1. Winning percentage on the first serve: This feature represents a chance of the player to get point on the first serve and it can be calculated by the following equation.

$$Winning\ \%\ on\ 1^{st}\ Serve = \frac{1^{st}\ \text{Serve Win}}{\text{Total } 1^{st}\ \text{Serve}} \qquad (4.1)$$

2. Winning percentage on the second serve: This feature represents a chance of the player to get point on the second serve. Equation 4.2 shows how to get this feature.

$$Winning\ \%\ on\ 2^{nd}\ Serve = \frac{2^{nd}\ Serve\ Win}{Total\ 2^{nd}\ Serve} \qquad (4.2)$$

3. Winning percentage on return serve: This feature represents a chance of the player to gets point on receiving from opponent's serve as shown in equation 4.3.

$$Winning\ \%\ on\ Return\ Serve = \frac{\text{Return Serve Win}}{\text{Total Return Serve}} \qquad (4.3)$$

4. Winning percentage on break point: This feature represents a chance of the player to get point when he faces the break point game as depicted in equation 4.4.

$$Winning\ \%\ on\ Break\ Point = \frac{\text{Break Point Win}}{\text{Total Break Point}} \qquad (4.4)$$

5. Total point wins: This feature represents an average of wining point per match as illustrated in equation 4.5.

$$Total\ Point\ Win = \frac{\text{Point Win}}{\text{Number of Matches}} \qquad (4.5)$$

6. First serve: This feature represents an average of number of serve per match as shown in equation 4.6.

$$First\ Serve = \frac{\text{Total First Serve}}{\text{Number of Matches}} \qquad (4.6)$$

7. Rank: This feature represents a rank of player in the tournament as illustrated in equation 4.7.

$$Rank = \frac{\text{Rank}}{\text{Max Rank}} \tag{4.7}$$

8. Aces: This feature represents number of serves that receiver can't touch the ball as illustrated in equation 4.8.

$$Aces = \frac{\text{Total Aces}}{\text{Number of Matches}} \tag{4.8}$$

9. Double faults: This feature represents number of serves that lands outside the service box or hits the net as illustrated in equation 4.9.

$$Double\ Faults = \frac{\text{Total Double Faults}}{\text{Number of Matches}} \tag{4.9}$$

10. Hard court: This feature represents the match that play on hard court.

11. Clay court: This feature represents the match that play on clay court.

12. Grass court: This feature represents the match that play on grass court.

In the tennis match, it is played between two players (singles) so the input data in 1–9 is needed to have two sets; the data set of player 1 and the data set of player 2 so the input vector consists of 21 parameters.

CHAPTER 5

EXPERIMENTS AND EVALUATION

To evaluate the proposed method, the high performance computer with the specification of Pentium Core2Duo 2.53 GHz and 2 GB of RAM is used for training the MLP. Next subsection describes data managing for our model.

As mention about the use of MLP, it is realized that the tennis prediction results will be more accurate than the use of Markov Chain Model. In order to prove the effectiveness of the proposed method, there are seven experiments were taken.

For the experiment I, this experiment is able to answer the following question. Could the proposed system (using the MLP) which is called StatEnv model, provides more accurate prediction results than Barnett and Clarke model which uses Markov Chain Model?

For the experiment II, this experiment is able to answer the following question. Could the proposed system which is using statistical data and environmental data which is called AdvancedStatEnv model, provides more accurate prediction results than the StatEnv model which is using only the statistical data?

For the experiment III, this experiment is able to answer the following questions. Could the proposed system (using Time series) which is called TimeSeries model, provides better prediction results than AdvancedStatEnv model?

For the experiment IV, this experiment is able to answer the following question. Which input features for the MLP can provide better results?

For the experiment V, this experiment is able to answer the following question. Does the court which is an environmental data, affect to the tennis predict result?

For the experiment VI, this experiment is able to answer the following question. How long of the past experience of players that more suitable for TimeSeries Model?

For the experiment VII, this experiment is able to answer the following question. Would the model provide better result if selects the appropriate input features and the past one year of experience?

5.1 MLP Modeling

This paper proposed three models of MLP which are StatEnv Model, AdvancedStatEnv Model and TimeSeries Model. These three models have different input features and parameters. Table 5.1 shows the input features which each model used and Table 5.2 shows the parameters of each model.

Table 5.1: The input features of each model.

Model	Input Features	Input Vector
StatEnv Model	1. Winning percentage on the first serve 2. Winning percentage on the second serve 6. First serve	6 nodes
AdvancedStatEnv Model	1. Winning percentage on the first serve 2. Winning percentage on the second serve 3. Winning percentage on return serve 4. Winning percentage on break point 5. Total point wins 6. First serve 7. Rank 8. Aces 9. Double faults 10. Hard court 11. Clay court 12. Grass court	21 nodes
TimeSeries Model	1. Winning percentage on the first serve 2. Winning percentage on the second serve 3. Winning percentage on return serve 4. Winning percentage on break point 5. Total point wins 8. Aces 9. Double faults 10. Hard court 11. Clay court 12. Grass court	31 nodes

5.2 Parameters

To find the suitable MLP model, the learning parameters are adjusted until the error is reduced into acceptable value. The appropriate value of each parameter is shown in the table 5.2.

Table 5.2: The appropriate value of parameters in MLP models.

Model	Hidden Node	Learning Rate	Momentum
StatEnv Model	20	0.3	0.2
AdvancedStatEnv Model	50	0.3	0.2
TimeSeries Model	150	0.3	0.2

The StatEnv Model has 6 input nodes but AdvancedStatEnv Model has 21 input nodes so the hidden node of AdvancedStatEnv Model should be increase from 20 nodes to 50 nodes to get the acceptable value of error. Therefore, the TimeSeries Model which has 31 input nodes, use 150 hidden nodes. All the numbers of hidden nodes that show in the table come from the experimental.

5.3 Data Managing

Clarke and Norton [14] show the way to collect the statistical data which release after the end of the match played so most of tournaments use their techniques to collect the data [15]. This thesis uses the statistical data which collected by the tournaments and manipulate it before because the data are enlarging every day. This process is called "Managing Data".

As shown in figure 5.1, The Managing Data process is a preprocessing which get data and manipulate it before store back to the database. After that, use the updated data to predict the tennis match. Moreover, when the data are increasing, this process will manipulate only the new data.

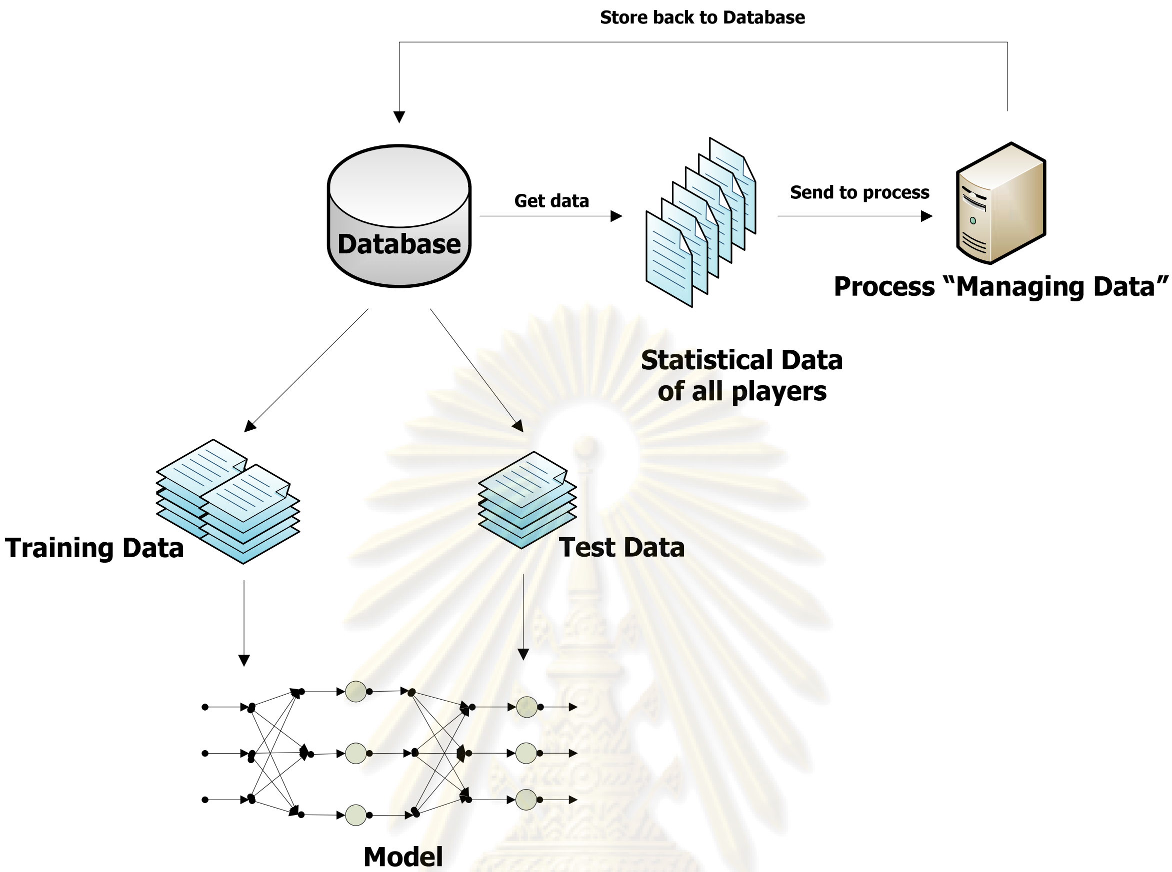

Figure 5.1 : The Managing Data Process.

This example shows how to manipulate the statistical data. Assume that Roger Federer and Novak Djokovic have been played only one tournament in the past at the French Open 2008 so the collected data from the tournament could be representing in Table 5.3 and Table 5.4.

Table 5.3: The statistical data of Roger Federer at French Open 2008.

Player1	Player2	Round	1^{st} Serve Win	Total 1^{st} Serve
Roger Federer	Diego Hartfield	1^{st}	39	47
Roger Federer	Fabrice Santoro	2^{nd}	32	42
Roger Federer	Janko Tipsarevie	3^{th}	95	107
Roger Federer	Tomas Berdych	4^{th}	44	59
Roger Federer	James Blake	Quarter-Final	49	65
			259	320

Table 5.4: The statistical data of Novak Djokovic at French Open 2008.

Player1	Player2	Round	1^{st} Serve Win	Total 1^{st} Serve
Benjamin Beeker	Novak Djokovic	1^{st}	42	50
Simone Bolelli	Novak Djokovic	2^{nd}	34	45
Samuel Querrey	Novak Djokovic	3^{th}	36	48
Lleyton Hewitt	Novak Djokovic	4^{th}	45	62
David Ferrer	Novak Djokovic	Quarter-Final	44	58
			201	263

To manipulate these collected data, there are three steps below;

- Selects all the historical data of each player. In this step, the data in table 5.3 and table 5.4 are the historical data of Roger Federer and Novak Djokovic.
- The value in the 1^{st} serve win column and total 1^{st} serve column are summarized.
- The winning percentage on 1^{st} serve is calculated by equation (4.1).

For example, the summation of 1^{st} serve win of Roger Federer is 259 and the summation of total 1^{st} serve of Roger Federer is 320. Then, the winning percentage of 1^{st} serve of Roger Federer is 259 /320 = 0.81. For Novak Djokovic, the summation of 1^{st} serve win of Novak Djokovic is 201 and the summation of total 1^{st} serve of Novak Djokovic is 263. Then, the winning percentage of 1^{st} serve of Novak Djokovic is 201/263 = 0.76. Therefore, other input features are calculated by using the equations above (equation (4.2) - equation (4.9)).

5.4 Training data

The statistical data and environmental data of match played obtained from OnCourt System and some websites which are ATP World Tour [4], Australian Open [5], French Open [6], Wimbledon [7] and US Open [8].

For the schedule of events in Grand Slam Tournament, the Australian Open is the first event in the year, second event is French Open, third event is Wimbledon and then US Open is the last event. For the StatEnv Model and AdvancedStatEnv Model, the training set is collected from the year 2003 until the year of prediction. For example, if the prediction is Australian Open 2006, the training set is collected from the beginning of the year 2003 to the end of year 2005. TimeSeries Model uses the same data as AdvancedStatEnv Model and also uses the collected data only in the past one year to be the input data (365 days before prediction).

5.5 Evaluation Results

Experiment I: Comparison between the existing model and the proposed StatEnv model.

The objective of this experiment is to compare the proposed system in the first version, namely, StatEnv model with the Barnett-and-Clarke model.

Table 5.5 : Comparison between Barnett-and-Clarke model and StatEnv Model.

Tournament	Barnett and Clarke Model	StatEnv Model
Australian Open 2007	74.8031 %	77.9528 %
French Open 2007	75.5906 %	72.4409 %
Wimbledon 2007	66.9291 %	68.2540 %
US Open 2007	70.8661 %	74.0157 %
Grand Slam 2007	**72.0472 %**	**73.1659 %**
Australian Open 2008	70.0787 %	70.8661 %
French Open 2008	60.6299 %	68.5039 %
Wimbledon 2008	74.8031 %	68.5039 %
US Open 2008	71.6535 %	73.2283 %
Grand Slam 2008	**69.2913 %**	**70.27756 %**

These two models use the same input features and test on the same dataset to compare the StatEnv Model which based on MLP with Barnett and Clarke model which based on Markov Chain Model.

As result in table 5.5, it shows that StatEnv Model provide more accurate results than Barnett and Clarke model. It can conclude that the MLP-based model will provide more accurate result than Markov Chain model. Therefore, this thesis has focused on tennis prediction model based on MLP.

Experiment II: Comparison between the StatEnv model and the AdvancedStatEnv model.

The objective of this experiment is to compare the proposed system in the second version, namely, AdvancedStatEnv model with the first version, namely, StatEnv model.

Table 5.6 : The accuracy of StatEnv Model and AdvancedStatEnv Model.

Tournament	StatEnv Model	AdvancedStatEnv Model
Australian Open 2007	77.9528 %	78.7402 %
French Open 2007	72.4409 %	70.8661 %
Wimbledon 2007	68.2540 %	78.7402 %
US Open 2007	74.0157 %	76.3780 %
Grand Slam 2007	**73.1659 %**	**76.1811 %**
Australian Open 2008	70.8661 %	79.5276 %
French Open 2008	68.5039 %	68.5039 %
Wimbledon 2008	68.5039 %	75.5906 %
US Open 2008	73.2283 %	71.6535 %
Grand Slam 2008	**70.27756 %**	**73.8189 %**

As the result from table 5.6, it can conclude that the appropriate input features in AdvancedStatEnv Model which are provide the more accurate prediction result than the input feature in StatEnv Model.

Experiment III: Comparison between the AdvancedStatEnv model and the TimeSeries model.

The objective of this experiment is to compare the proposed system in the second version, namely, AdvancedStatEnv model with the TimeSeries model.

Table 5.7 : The accuracy of AdvancedStatEnv Model and TimeSeries Model.

Tournament	AdvancedStatEnv Model	TimeSeries Model
Australian Open 2007	78.7402 %	81.1024 %
French Open 2007	70.8661 %	78.7402 %
Wimbledon 2007	78.7402 %	80.3150 %
US Open 2007	76.3780 %	73.2283 %
Grand Slam 2007	**76.1811 %**	**78.3465 %**
Australian Open 2008	79.5276 %	80.3150 %
French Open 2008	68.5039 %	70.8661 %
Wimbledon 2008	75.5906 %	73.2283 %
US Open 2008	71.6535 %	77.1654 %
Grand Slam 2008	**73.8189 %**	**75.3937 %**

To compare TimeSeries Model with AdvancedStatEnv Model, there are two tennis events that can be compared which are Australian Open 2007 and Australian Open 2008. As result in table 5.7, the accuracy of TimeSeries Model is more than the AdvancedStatEnv Model. This can conclude that the experience of the players in the past one year directly affect to the prediction results.

Experiment IV: Comparison between the original AdvancedStatEnv model and AdvancedStatEnv model after adding new features.

The objective of this experiment is to compare the original AdvancedStatEnv model and AdvancedStatEnv model after adding new features.

Table 5.8 : The accuracy of each AdvancedStatEnv Model which include new feature(1).

Tournament	Add new features (AdvancedStatEnv Model)		
	Original	First Serve	Rank
Australian Open 2007	78.7402 %	76.3780 %	78.7402 %
French Open 2007	70.8661 %	68.5039 %	73.2283 %
Wimbledon 2007	78.7402 %	79.5276 %	75.5906 %
US Open 2007	76.3780 %	74.8031 %	74.8031 %
Grand Slam 2007	**76.1811 %**	**74.8032 %**	**75.5906 %**
Australian Open 2008	79.5276 %	76.3780 %	76.3780 %
French Open 2008	68.5039 %	67.7165 %	63.7795 %
Wimbledon 2008	75.5906 %	77.1654 %	77.1654 %
US Open 2008	71.6535 %	70.8661 %	71.6535 %
Grand Slam 2008	**73.8189 %**	**73.0315 %**	**72.2441 %**

Table 5.9 : The accuracy of each AdvancedStatEnv Model which includes new feature(2).

Tournament	Add new features (AdvancedStatEnv Model)			
	Original	Aces	Double Faults	Aces + Double Faults
Australian Open 2007	78.7402 %	78.7402 %	81.1024 %	82.6772 %
French Open 2007	70.8661 %	75.5906 %	72.4409 %	71.6535 %
Wimbledon 2007	78.7402 %	81.1024 %	79.5276 %	81.1024 %
US Open 2007	76.3780 %	73.2283 %	77.1654 %	76.3780 %
Grand Slam 2007	**76.1811 %**	**77.1654 %**	**77.5591 %**	**77.9528 %**
Australian Open 2008	79.5276 %	77.9528 %	80.3150 %	75.5906 %
French Open 2008	68.5039 %	67.7165 %	69.2913 %	71.6535 %
Wimbledon 2008	75.5906 %	75.5906 %	77.1654 %	76.8031 %
US Open 2008	71.6535 %	74.0157 %	70.0787 %	74.8031 %
Grand Slam 2008	**73.8189 %**	**73.8189 %**	**74.2126 %**	**74.7126 %**

To see the effective of the input features to the prediction results, this research adds the input features into the tennis prediction model which are First serve (the number of first serve), Rank, Aces และ Double Faults

From table 5.8, there are two input features which are First Serve and Rank that decrease the accuracy of prediction results.

From table 5.9, there are two input features which are Aces and Double Faults that affect to the prediction results and provide more accurate prediction result. After add each features, the results is more accurate than the original one. Then, this experiment tried to add both of Aces and Double Faults to see how it works. As the result in table 5.9, it can conclude that both of Aces and Double Faults affect to the prediction results.

Experiment V: Comparison between the original AdvancedStatEnv model and AdvancedStatEnv model after remove court feature.

The objective of this experiment is to compare the original AdvancedStatEnv model and AdvancedStatEnv model after remove court feature.

Table 5.10 : The accuracy of AdvancedStatEnv Model which remove court feature.

Tournament	AdvancedStatEnv Model	
	Original	Remove Court feature
Australian Open 2007	78.7402 %	76.3780 %
French Open 2007	70.8661 %	70.0787 %
Wimbledon 2007	78.7402 %	77.9528 %
US Open 2007	76.3780 %	69.2913 %
Grand Slam 2007	**76.1811 %**	**73.4252 %**
Australian Open 2008	79.5276 %	75.5906 %
French Open 2008	68.5039 %	70.8661 %
Wimbledon 2008	75.5906 %	72.4409 %
US Open 2008	71.6535 %	70.0787 %
Grand Slam 2008	**73.8189 %**	**72.2441 %**

As the result in table 5.10, the original one which has court gives better tennis prediction result. It can conclude that court which is an environmental data, affect to the tennis prediction model.

Experiment VI: Comparison TimeSeries model in difference experience time.

The objective of this experiment is to compare the TimeSeries model in different experience time.

Table 5.11: The accuracy of each TimeSeries Model in different of experience time (1).

Tournament	TimeSeries Model			
	12 Months	9 Months	6 Months	3 Months
Australian Open 2007	81.1024 %	81.8898 %	78.7402 %	77.9528 %
French Open 2007	78.7402 %	75.5906 %	73.2283 %	81.8898 %
Wimbledon 2007	80.3150 %	67.7165 %	78.7402 %	76.3780 %
US Open 2007	73.2283 %	75.5906 %	71.6535 %	72.4409 %
Grand Slam 2007	**78.3465 %**	**75.1969 %**	**75.5906 %**	**77.1654 %**
Australian Open 2008	80.3150 %	78.7402 %	73.2283 %	74.0157 %
French Open 2008	70.8661 %	67.7165 %	65.3543 %	72.4409 %
Wimbledon 2008	73.2283 %	67.7165 %	66.1417 %	71.6535 %
US Open 2008	77.1654 %	77.9528 %	75.5906 %	70.8661 %
Grand Slam 2008	**75.3937 %**	**73.0315 %**	**70.0787 %**	**72.2441 %**

Table 5.12: The accuracy of each TimeSeries Model in different of experience time (2).

Tournament	TimeSeries Model (Cont.)		
	12 Months	15 Months	18 Months
Australian Open 2007	81.1024 %	81.8898 %	77.5906 %
French Open 2007	78.7402 %	75.5906 %	65.3543 %
Wimbledon 2007	80.3150 %	76.3780 %	76.3780 %
US Open 2007	73.2283 %	70.8661 %	70.0787 %
Grand Slam 2007	**78.3465 %**	**76.1811 %**	**72.3504 %**
Australian Open 2008	80.3150 %	73.2283 %	78.7402 %
French Open 2008	70.8661 %	70.8661 %	66.9291 %
Wimbledon 2008	73.2283 %	73.2283 %	74.0157 %
US Open 2008	77.1654 %	70.0787 %	69.2913 %
Grand Slam 2008	**75.3937 %**	**71.8504 %**	**72.2441 %**

To concentrate on the experience of the tennis player, this research tried to find out how long of experience time that affect to the prediction results by using the Time-Series model. There are four period of times which are three months, six months, nine months and twelve months. As shown in the table 5.11 and table 5.12, the twelve months is the best period of time for the model to increase the accuracy of prediction model.

Experiment VII: Combine all experiments to make the better model.

The objective of this experiment is to compare the original TimeSeries model and combining of appropriate input features with past one year of experience model.

Table 5.13 : The accuracy of the original TimeSeries Model and the best model.

Tournament	TimeSeries Model (12 Months)	
	Original	Add Features (Aces + Double Faults)
Australian Open 2007	81.1024 %	83.4646 %
French Open 2007	78.7402 %	77.1654 %
Wimbledon 2007	80.3150 %	81.1024 %
US Open 2007	73.2283 %	79.5276 %
Grand Slam 2007	**78.3465 %**	**80.3150 %**
Australian Open 2008	80.3150 %	82.6772 %
French Open 2008	70.8661 %	72.4409 %
Wimbledon 2008	73.2283 %	78.7402 %
US Open 2008	77.1654 %	75.4094 %
Grand Slam 2008	**75.3937 %**	**77.3169 %**

As the results that show in experiment IV and V, this thesis add the Aces and Double Faults as input features into the TimeSeries Model which use the past one year of players' experience to build a final model.

As the results in table 5.13, the prediction results from the final model (adding Aces and Double Faults) gives better result than the original one. It can be conclude that not only the past one year of players' experience but Aces and Double Faults are also affect to the tennis prediction model.

CHAPTER 6

CONCLUSION

6.1 Discussion

Most of research paper concentrated on Australian Open but this thesis has worked on all tournament of Grand Slam which is Australian Open, French Open, Wimbledon and US Open. As the results show in experiments above, the result from Australian Open could present how the model work out step by step, in another hand, the result from the rest tournament is not stable. Then, this thesis averages the prediction result of four tournaments to see how the models work.

The data that use in this thesis is a statistical data which collected after the tennis match's done since the year of 2003. This thesis focused on the year 2007-2008 because there is enough data for the tennis prediction model.

As the experiments I in chapter 5, it shows that the use of MLP provides more accurate result than using Markov Chain model. Experiment II shows that adding some input features is increasing the accuracy of the prediction result. Then, the experiment III shows that the experience of players is also provide more accurate prediction result by using MLP based on Time-Series model.

Moreover, it has been already proved that more input features provide more accurate results but what feature affects to the tennis prediction model. This can be found out in experiment IV which shows that the appropriate input features could be increasing the prediction result. The good answers are "Double Faults" and "Aces" input features.

Furthermore, the use of MLP based on Time-Series shows that the experience of players affect to the prediction results but how long of experience should be taken into the model? As the result from the experiment V, the past one year of players' experience will provide the most accurate prediction results.

In addition, it can be also concluded that the use of MLP based on appropriate input features and the experience of players in the past one year provide much more accuracy than Markov Chain (Barnett and Clarke) as shown in table 6.1.

Table 6.1: The accuracy of Barnett and Clarke Model and the final model (based on MLP)

Tournament	Barnett and Clarke Model	Final Model (Based on MLP)
Australian Open 2007	74.8031 %	83.4646 %
French Open 2007	75.5906 %	77.1654 %
Wimbledon 2007	66.9291 %	81.1024 %
US Open 2007	70.8661 %	79.5276 %
Grand Slam 2007	**72.0472 %**	**80.3150 %**
Australian Open 2008	70.0787 %	82.6772 %
French Open 2008	60.6299 %	72.4409 %
Wimbledon 2008	74.8031 %	78.7402 %
US Open 2008	71.6535 %	75.4094 %
Grand Slam 2008	**69.2913 %**	**77.3169 %**

Finally, the table 6.2 and figure 6.1 show the average result of the year 2007 and 2008 from the models which based on MLP but different of input features. It can conclude that not only an environmental data but the experiences of player are also affect to the tennis prediction model. The prediction results of all tournaments in Grand Slam in the year 2007 are shown in Table 6.3 and Figure 6.2 and the prediction results the year 2008 are shown in Table 6.4 and Figure 6.2.

Table 6.2 : The average prediction results from MLP models in the year 2007-2008

Tournament	StatEnv Model	AdvancedStatEnv Model	TimeSeries Model
Grand Slam 2007	74.0158 %	77.9528 %	80.3150 %
Grand Slam 2008	72.6378 %	74.7126 %	77.3169 %

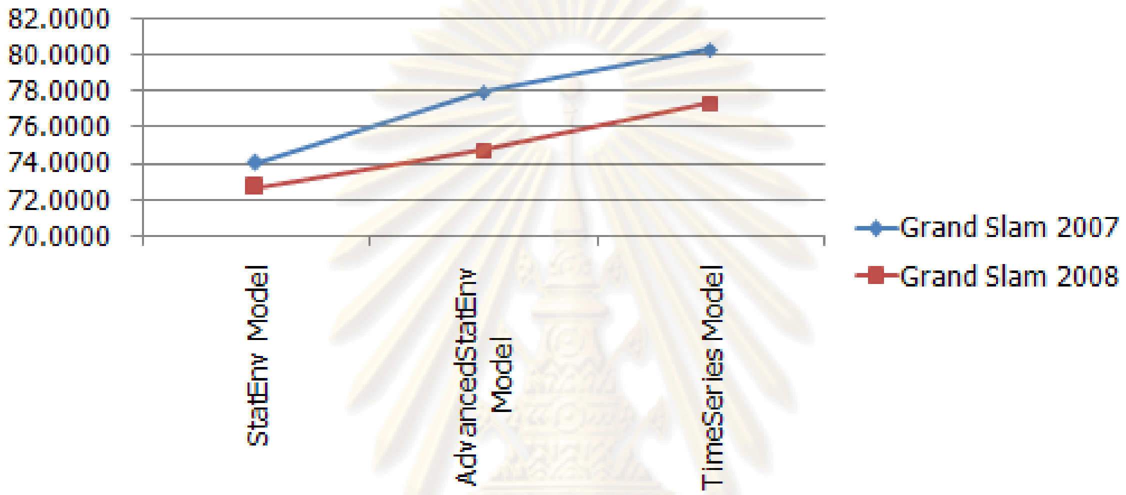

Figure 6.1 : The graph of average prediction results from MLP models in the year 2007-2008

Table 6.3 The prediction results from MLP model of all Grand Slam tournaments in year 2007

Tournament	StatEnv Model	AdvancedStatEnv Model	TimeSeries Model
Australian Open 2007	77.9528 %	82.6772 %	83.4646 %
French Open 2007	63.7795 %	71.6535 %	77.1654 %
Wimbledon 2007	77.1654 %	81.1024 %	81.1024 %
US Open 2007	77.1654 %	76.3780 %	79.5276 %

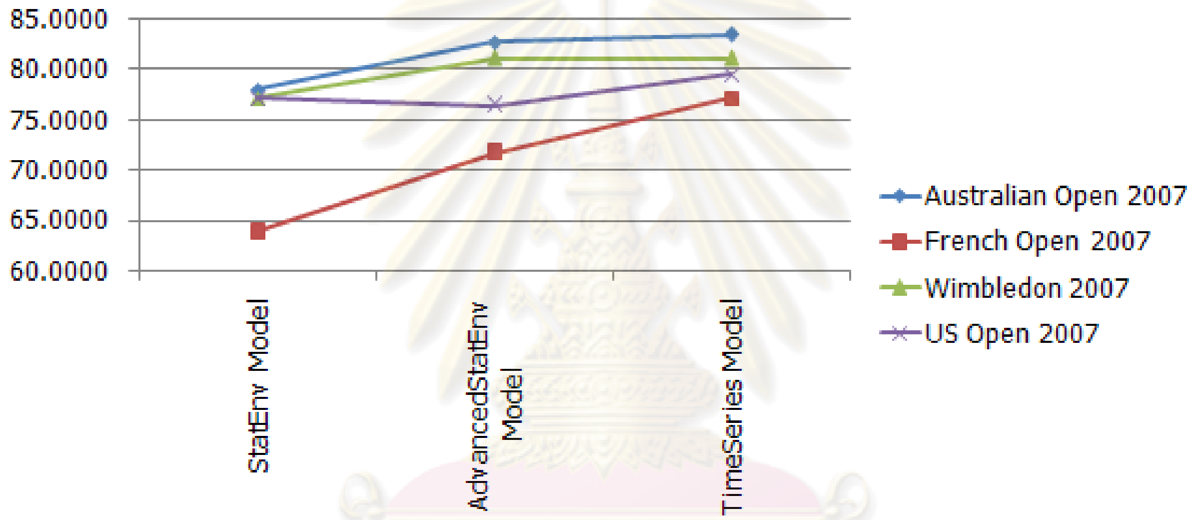

Figure 6.2 : The prediction results from MLP model of all Grand Slam tournaments in year 2007

Table 6.4 : The prediction results from MLP model of all Grand Slam tournaments in year 2008

Tournament	StatEnv Model	AdvancedStatEnv Model	TimeSeries Model
Australian Open 2008	72.4409 %	75.5906 %	82.6772 %
French Open 2008	68.5039 %	71.6535 %	72.4409 %
Wimbledon 2008	77.1654 %	76.8031 %	78.7402 %
US Open 2008	72.4409 %	74.8031 %	75.4094 %

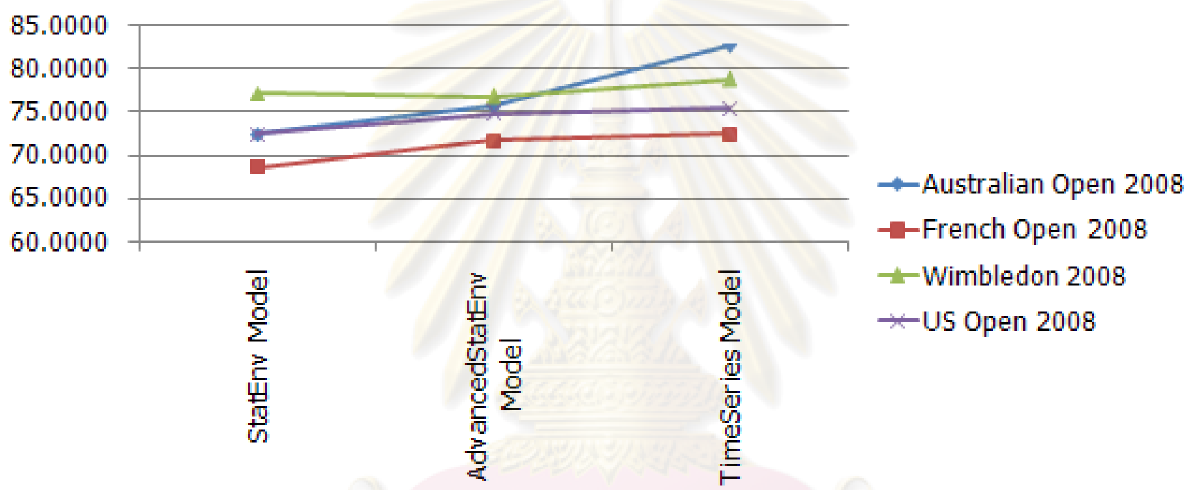

Figure 6.3 : The prediction results from MLP model of all Grand Slam tournaments in year 2008

Since, this thesis focused on input features as shown in experiment IV and V so it might say that the appropriate input features will provide more accurate result. For AdvancedStatEnv Model, researcher selects input features that other researchers always use in tennis prediction model.

To prove the input features, this thesis analyze each of input feature by calculating the weight values of input nodes and hidden nodes. Then, the input feature which has higher weight value is more important or appropriate than others. The weight values of each input feature of AdvancedStatEnv Model which predicts tennis Grand Slam 2008 are shown in table 6.5.

Table 6.5 : Weight of AdvancedStatEnv Model in Grand Slam 2008

Input Features	Player	Weight	Average Weight
Winning percentage on the first serve	1	-1.4956	(|-1.4956| + |-1.6361|) / 2
	2	-1.6361	= 1.5659
Winning percentage on the second serve	1	0.6701	(|0.6701| + |-0.5000|) / 2
	2	-0.5000	= 0.5851
Winning percentage on return serve	1	-0.1788	(|-0.1788| + |-0.4388|) / 2
	2	-0.4388	= 0.3088
Winning percentage on break point	1	-1.8964	(|-1.8964| + |-1.0534|) / 2
	2	-1.0534	= 1.4749
Total point wins	1	-0.7584	(|-0.7584| + |0.6907|) / 2
	2	0.6907	= 0.7246
Aces	1	1.2131	(|1.2131| + |1.0437|) / 2
	2	1.0437	= 1.1284
Double faults	1	1.7679	(|1.7679| + |-1.2312|) / 2
	2	-1.2312	= 1.4996
court	-	0.8806	(|0.8806| + |1.0301| +
	-	1.0301	|1.5152|) / 3
	-	1.5152	= 1.1420

As the weight values are shown in table 6.5, the average weight values of each input feature are calculated to see how important they are. Then, the appropriate input features are ordered in ascending as below.

1) Winning percentage on the first serve
2) Double faults
3) Winning percentage on break point
4) Court
5) Aces
6) Total point wins

7) Winning percentage on the second serve

8) Winning percentage on return serve

It can conclude that “Winning percentage on the first serve” is the most appropriate input feature that directly affect to the accuracy of tennis prediction model. On another hand, the “Winning percentage on return serve” is the lowest appropriate input feature for the model.

6.2 Conclusion

In this thesis, the new approach to create the tennis prediction model is shown. To get more accuracy than the current techniques, the Multi-Layer Perceptron is applied to predict the winner of the tennis matches. Three proposed models, which consist of different set of input parameters, are shown that the selection of appropriated parameters extremely affect to the prediction. From comparison among the models, the MLP Model, the appropriate input features, and concentration on the experience of players in the past one year provide more accuracy than the current tennis models. After combining the appropriate input features and the past one year of player’s experience, the prediction model event provides more accurate result.

6.3 Future work

As explained in section 1.5 that there are two fundamental research points of Tennis Prediction Model. The first one is to help the organizer to predict who the winner of the game is. Another research point is to help the organizer to manage the time for playing game (the length of match).

This thesis concentrates only on the winner of the game by improving the accuracy of tennis prediction results. The prediction of time for playing game would be future work. To predict the probable length of match, it may help the organizer to manage the time such as how long for broadcasting and how many advertisements during the broadcast time.

REFERENCES

[1] Pollard, G., and Barnett, T. (2006). Fairer service exchange mechanisms for tennis when some psychological factors exist. Proceedings of the 8nd Australian Conference on Mathematics and Computers in Sport : 189-198.

[2] Magnus, J.R. and Klaassen, F.J.G.M. (1999). On the advantage of serving first in a tennis set: four years at Wimbledon. The Statist 48(2): 247-256.

[3] KAN-soft. 2001. OnCourt[Software]. Available download from: www.oncourt.info.

[4] ATP World Tour. Website[online]. (n.d.). Available from: www.atpworldtour.com [2009, January 1]

[5] Australian Open. Website[online]. (n.d.). Available from: www.australianopen.com [2009, January 1]

[6] French Open. Website[online]. (n.d.). Available from: http://2008.rolandgarros.com [2009, January 1]

[7] Wimbledon. Website[online]. (n.d.). Available from: http://championships.wimbledon.org [2009, January 1]

[8] US Open. Website[online]. (n.d.). Available from: www.usopen.org [2009, January 1]

[9] Kemeny, J.G., and Snell, J.L. 1960. Finite Markov chains. Springer-Verlag, Princeton, New Jersey D. Van Nostrand.

[10] Barnett, T., and Clarke, S.R. (2002). Using Microsoft Excel to model a tennis match. Proceedings of the 6M&CS : 63-68.

[11] Barnett, T., and Clarke, S.R. 2005. Combining player statistics to predict outcomes of tennis matches. IMA Journal of Management Mathematics 16 (2): 113-120.

[12] Franc, J.G.M., Klaassen, and Jan, R., Magnus. 2003. Forecasting the winner of a tennis match. European Journal of Operational Research : 257-267.

[13] Simon Haykin 1999. Neural Network: A Comprehensive Foundation, 2nd Edition. Prentice Hall.

[14] Clarke, S.R., and Norton, P. (2002). Collecting statistics at the Australian Open tennis championship. Proceedings of the 6M&CS: 105-111.

[15] Barnett, T., Brown, A., and Clarke, S.R. (2006). Developing a tennis model that reflects outcomes of tennis matches. Proceedings of the 8^{th} Australian Conference on Mathematics and Computers in Sport: 178-188.

VITAE

Amornchai Somboonphokkaphan was born in 1984. I received a Bachelor Degree in Science (Majoring Computer Science) from Thammasart University in 2006. I am working as Senior Application Developer for Wealth Management System Limited, Bangkok and is also pursuing a Masters degree in computer science.