Personalized and Course-Specific Performance Forecasting Model in Cycling

By

Xiaoxing Qiu

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

in

Computer Science

May 14th, 2021

Nashville, Tennessee

Approved: Date:

_____ _____

Advisor: Jules White

_____ _____

Second Reader: Douglas Schmidt

ACKNOWLEDGMENTS

I have received a great amount of assistance and support throughout my thesis work.

Firstly, I would like to express my deep appreciation to my advisor, Prof. Jules White. He offered me insightful suggestions and feedback in clarifying and formatting the research questions and searching for appropriate solutions. Without his assistance and involvement in the research, this thesis could not be accomplished. I also would like to express my gratitude to my thesis reader, Prof. Douglas Schmidt, for his encouragement and comments on my research.

I would like to acknowledge members in the MAGNUM research group, in particular, Zhongwei Teng and Carlos Olea, who provided my lots of advice through my thesis work. I also would like to thank Gabriela Gresenz for her help in my research. I would not have successfully completed this thesis without their support.

Finally, I would like to thank my family for supporting and encouraging me with their love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Context

In elite cycling, to obtain desired training effects, it is necessary to predict performance of a cyclist on a specific course. There are two major components of a cyclist's performance: speed at each moment and exercise intensity. Speed over time shows how fast an athlete can ride at each time step but it alone is not sufficient to determine a cyclist's performance since it does not consider the exercise intensity. For example, a biker may need to climb up on a steep uphill road, which requires significantly slower speeds and greater intensity than a similar climb on smooth ground. Similarly, a high-speed downhill ride on smooth terrain will place less stress on a rider's core muscle groups compared to the same downhill speed across tree roots. The individual terrain features, turns, gradients, and other aspects of a course have a large impact on a rider's speed and intensity, but current models are mainly course-independent.

There are several candidate metrics for exercise intensity, namely heart rate, power output, $\dot{V}O_2max$ etc. According to Jeukendrup etc's study, heart rate is a reliable indicator of exercise intensity in cycling [1]. However, heart rate can be affected by terrain factors such as grade of the hill, which varies on different riding courses. Research has shown that a cyclist's heart rate drifts upwards after exercising for 20 to 60 minutes despite unchanged work loads [2, 3], which is called cardiac drift. This phenomenon is considered to be associated with an increase of core body temperature during exercise [4, 5], which may cause athletes to lower their speed in order to maintain their heart rate. On the other hand, higher speed will lead to a higher heart rate in a given condition [3]. Moreover, heart rate responses vary with individual factors of cyclists such as gender, age etc. [3] and their cadence on different parts of a given course. As a result, heart rate forecasting should

be personalized and course-specific. A personalized performance forecasting model for specific courses will be vital for cyclists to set up achievable speed goals at different places in advance and maximize training effects without over-training.

1.2 Performance Forecasting Model

Researchers have built various models to predict performance in cycling. Ankang Le et al. proposed a mathematical model to evaluate athletes' heart rate response under moderate exercise intensity based on physical and physiological principles [3]. Alejandro Lucia etc. analyzed the preferred cadence of professional cyclists and found that on flat stages they tend to adopt higher cadences (around 90 rpm) while on mountain ascents cadences are around 70 rpm [6]. However, their models focus on laboratory conditions. Course-specific factors such as the slope of the road in real courses are unfortunately not taken into consideration which have significant influence on cyclists' heart rate response.

Due to development of wearable devices, large amounts of data now can be collected and processed via mobile devices, which offers an opportunity for building personalized performance forecasting models. Farrokh Mohammadzadeh et al. utilized a support vector machine predictor to predict the breathing rate based on the 3-D accelerations, heart rate, body temperature, electrodermal activity, humidity etc. in a controlled lab environment [7]. Yuchi Ming, Feng Xiao and their colleagues used a feed forward neural network (FFNN) to investigate the relationship between heart rate and physical activity in daily life with the help of a wearable physical activity recorder which monitors the 3-D accelerations of the body [8, 9].

Recurrent neural networks (RNNs) are able to exhibit sequential correlation and can seamlessly model problems with multiple inputs. Therefore these models are widely used in natural language processing and time series prediction [10]. In athletic performance forecasting, RNN based models can take personalized factors such as blood pressure and running speed during exercise to make a heart rate predictions. Jianmo Ni etc. proposed

an LSTM based model to learn a user's heart rate profile during exercise and offer workout route recommendation and short term heart rate prediction [11]. Minmin Luo etc. also proposed an LSTM-based model to predict heat rate based on heart rate signal, gender, age, accelerations and mental state [12]. Mingyun Bian et al. tracked facial key points from each frame of facial videos to estimate heart rate [2].

In summary, although there have been models for performance forecasting, most studies focus on the heart rate profile collected either during daily activities or under laboratory conditions. There are few models that are course-specific and personalized to forecast a cyclist's heart rate and speed. However, such a model can be beneficial to both cyclists and coaches. For example, a cyclist needs a model to set up various speed goals at different parts of a course before a competition or predict athletes' heart rate based on given speed goals. A coach can use such a model to evaluate the heart rate of an athlete on a specific course and at a given speed to ensure the exercise intensity and avoid over-training. Or given specific heart rate goals, these models can be used to predict how fast a cyclist can ride at different parts of the course.

1.3 Thesis Contributions

In this thesis, personalized multivariate models are proposed to forecast the heart rate and speed of a cyclist on a specific course. These models take the course-specific factors at each part of the course, such as the grade of road and the altitude, as well as current rider details, such as the cadence, into consideration and forecast the heart rate and speed of the athlete based on them. Results of heart rate and speed forested by different models are compared with each other by mean squared error (MSE).

1.4 Thesis Organization

The organization of this thesis is as follows: in Chapter 2, we review the fundamental research on all models that are utilized in this thesis, including basic concepts of each

model and their advantages as well as drawbacks respectively. In Chapter 3, we build random forest (RF), artificial neural network (ANN), recursive neural network (RNN), and long short term memory (LSTM) models to forecast the heart rate of athletes on a specific route. The results of different models are then compared against each other. In Chapter 4, we focus on forecasting the athlete's speed using RF, ANN, RNN and LSTM models. Finally we discuss the limitations of these models and conclude in Chapter 5.

Chapter 2

Models and Methodology

For cycling performance forecasting, there are important athlete and course-specific factors that must be taken into account. It is natural to consider multivariate models for performance forecasting. Random forest (RF) and neural networks are popular machine learning algorithms because they can be utilized both for regression and classification and are able to take multiple inputs. According to the overview of Leijnen et al., there are mainly 13 major neural network architectures [13]. Among these architectures, feed forward neural networks (FFNN) [8, 9], basic recurrent neural networks (RNN) [14] and long short term memory (LSTM) [2, 11] are utilized by researchers in performance forecasting. Therefore, all these three neural networks and random forest are selected to build a personalized model to predict the performance of a cyclist on a specific routes. Since LSTM is a special type of RNN, in order to distinguish these two models, basic RNN models are called simple RNN models in this thesis.

In this chapter, we first introduce metrics to evaluate performance forecasting models and briefly discuss feature selection and the over-fitting problem in machine learning. Then, basic principles of RF, FFNN, RNN and LSTM are briefly reviewed and their characteristics, advantages, and disadvantages are discussed.

2.1 Model Training and Evaluation

Most machine learning problems can be categorized into two types: classification and regression. Classification is the problem of classifying an object to one of several categories. For example, given a picture of an animal, a classification problem can be identifying whether the picture shows a cat or not. Regression is the problem of approximating a function f mapping from input vector x to output vector y. For example, given size

and number of rooms as well as the floor of an apartment, a regression problem can be outputting the price of this apartment.

In the performance forecasting problem, the performance of an athlete on a give course is a function of personalized factors such as cadence and terrain factors such as the grade and altitude of the course. Therefore, it is a regression problem.

A machine learning model should not only offer accurate predictions on previously seen inputs but also perform well on new unobserved data [15]. Therefore, the original dataset should be split into two subsets: a training set and a test set. A common train-test split ratio is 80/20, which means the training set contains 80% of the original data and the test set contains the rest. The training set is utilized to train the machine learning model. During training, the test set is not used and only the training set is taken as input. After training, the test set is used as the input and the error between predicted values and the real value is measured. The error on the training set is called training error while the error on the test set is called test error.

Over-fitting is a major challenge in machine learning. It occurs when the test error is much larger than the training error. Figure 2.1 illustrates an overfit curve and a well-fit curve over a series of data points. The overfit curve passes exactly through all data points but it does not capture the general trend of the data. In performance forecasting, an overfit model may offer low error on the training set but very large error on the test set.

The error between predicted values and real values is measured via error metrics. Mean squared error (MSE) and mean absolute error (MAE) are two of the most popular error metrics and are both widely used in machine learning. Let $y_1, y_2, ..., y_n$ be the target values and $\hat{y}_1, \hat{y}_2, ..., \hat{y}_n$ be the predicted values. MSE and MAE metric can be described as Formula 2.1.

$$\begin{aligned} MSE &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \\ MAE &= \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \end{aligned} \tag{2.1}$$

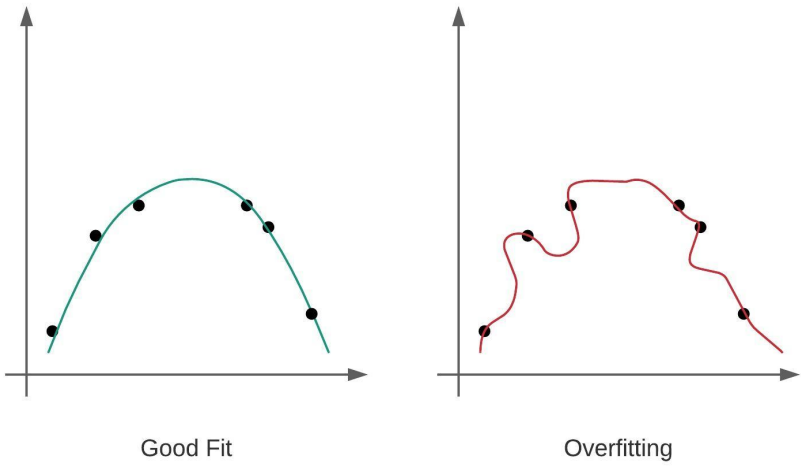


Figure 2.1: Over-fitting and Good Fit
[15]

By squaring the difference between the prediction and real values, MSE weight large errors more highly than the small errors. In performance forecasting, larger errors in heart rate and speed are particularly undesirable, as pushing an athlete past their limits even for short periods of time can dramatically reduce their performance later in a course. Therefore, the MSE metric is selected to evaluate the machine learning models built for cycling performance prediction.

2.2 Feature Selection

In the performance forecasting problem, an athlete's performance is affected by numerous features, such as temperature, the grade of course etc. However, not all factors should be provided as input to the machine learning models. Random forest models are sensitive to data variation, which means that a tiny difference in the data set can cause huge difference in the prediction. Pruning irrelevant features to an athlete's performance can reduce model over-fitting and improve forecasting accuracy. Neural networks have built-in mechanisms to mitigate over-fitting and carry out feature selection by assigning significant

features larger weights. However, this process consumes a large amount of time and requires a large amount of data. Removing less important factors can accelerate the training process of neural networks and reduce the amount of data needed. Therefore, in this thesis, feature selection is utilized for both random forest and neural networks.

In statistics, a correlation coefficient is utilized to characterize how strong a relationship is between two variables. A correlation coefficient is a real number between -1 and $+1$ and the absolute value of correlation coefficient shows the strength of a relationship. For example, assume there is a cyclist riding on a plane with constant velocity and direction. Assume there are two factors, one is the velocity of the cyclist, denoted as v, and the other is the color of the bike, denoted as c. Based on the property of uniform linear motion, the distance d can be described as the product of cyclist velocity v and time t. If one would build a distance forecasting model, v rather than c should be taken into consideration. The correlation coefficient between v and d is $\rho_{v,d} = 1$ while that between c and d is $\rho_{c,d} = 0$, which shows that there is a linear relationship between v and d while c and d are uncorrelated. Therefore, by calculating and comparing correlation coefficients between features and the target value, significant features can be extracted and irrelevant features can be filtered.

Two common correlation coefficients are widely used: the Pearson correlation coefficient and the Spearman's rank correlation coefficient. While Pearson's correlation coefficient assesses linear correlation [16], Spearman's rank correlation coefficient focuses on the monotonic relationship between two random variables [17]. According to the study by Bishara et al. [18], for non-normal data, calculating Spearman's correlation coefficient may be an optimal strategy when the data size is larger than 20. For performance forecasting, the heart rate and velocity of a cyclist show complex non-linear relationships with the factors such as cadence and the grade of the road etc. and the number of data is far more than 20. Spearman's rank correlation coefficient is more suitable for calculation of the correlation coefficient and therefore is selected to filter the personalized and terrain factors compared

with Pearson's correlation coefficient.

Compared with Pearson correlation, for Spearman's rank correlation coefficient, the raw data, such as heart rate, speed, the grade of course etc., needs to be converted to rank variables. Given a heart rate sequence HR and hr is a heart rate in this sequence, the rank variable r_{hr} of hr is obtained as follows [19]:

- Sort HR in ascending order, and denote the sorted heart rate sequence as HR_s
- The position where hr is in HR_s is the rank variable r_{hr} of hr

For example, given a heart rate sequence over 4 seconds are $\{100, 102, 95, 86\}$, the corresponding rank value for this heart rate sequence is $\{3, 4, 2, 1\}$.

Assume there are two data sequences, namely X and Y, and each of them has n examples. Based on aforementioned method, two ranked sequences rg_X and rg_Y can be obtained. The formula for Spearman's correlation coefficient is expressed in Formula 2.2 [19].

$$\rho_{rg_X, rg_Y} = \frac{cov(rg_X, rg_Y)}{\sigma_{rg_X} \sigma_{rg_Y}} \tag{2.2}$$

where $cov(rg_X, rg_Y)$ is the covariance of ranked sequences rg_X and rg_Y while σ_{rg_X} and σ_{rg_Y} are the standard deviations of the ranked sequences.

2.3 Random Forest

Random forest is a popular machine learning algorithm. Unlike neural networks, training a random forest does not require tuning the hyper parameters and therefore it is an easy-to-use and fast learning algorithm. In performance forecasting, random forest is utilized to take personalized factors, namely cadence, as well as terrain factors, such as the grade and altitude of a given course as input and forecast the heart rate and speed of the athlete.

Random forests are based on decision tree learning. A decision tree has a tree-like structure that captures a conditional sequence. Figure 2.2 shows a decision tree to decide

whether we go out to play soccer. A prediction process in this decision tree can be as follows: if there is an available soccer field and if it is not raining, then we will play soccer.

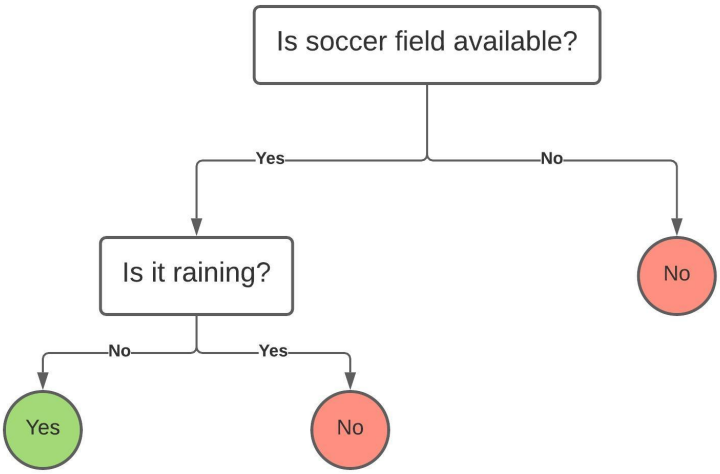


Figure 2.2: Decision Tree Example

While decision trees are simple to understand and interpret, the major disadvantage is that they are unstable. A small change in the data can dramatically influence the structure of the decision tree. In order to solve this, random forests utilize more than one decision tree and output the result using a bootstrap aggregation process. The bootstrap aggregation process is as follows [20]:

Given a training set $X = \{x_1, x_2, x_3, ..., x_m\}$ and $Y = \{y_1, y_2, y_3, ..., y_m\}$. For $d = 1, 2, ..., D$:

1. Sample X uniformly and repeatedly with replacement for m times, the sample sets are denoted as X_d and Y_d respectively

2. Train a regression tree t_d with X_d and Y_d

After training, these regression trees are utilized to predict an unseen sample u and prediction $\hat{p}(u)$ is obtained by averaging the predictions for all the individual regression trees:

$$\hat{p}(u) = \frac{1}{D} \sum_{d=1}^{D} t_d(u) \tag{2.3}$$

The variance can be calculated by

$$\sigma^2 = \frac{1}{D-1}\sum_{d=1}^{D}[t_d(u) - \hat{p}(u)]^2 \tag{2.4}$$

In random forest models, D is the number of individual decision trees. An optimal D can be found by cross-validation [21]. In this thesis, the scikit-learn library is utilized to build and train the random forest model.

2.4 Artificial Neural Networks

Random forest is less computationally expensive and can be easily implemented without tuning a large amount of hyperparameters. However, it also has a tendency to overfit to the training set. Moreover, it does not take temporal dependency of the data into consideration. In performance forecasting, the heart rate of an athlete at a specific moment depends on not only the cadence and velocity at that time but also the heart rate in the previous seconds. For example, due to cardiac drift, a chyclist's heart rate will rise after the exercise starts while the grade and speed remains almost the same. In order to handle these situations, artificial neural networks (ANNs) are introduced for performance forecasting.

ANNs are a series of computing systems with an inspiration from biological neural networks [22, 23, 24]. An ANN consists a set of computing units (neurons) which simulate how neurons and are interconnected with weights. While the behavior of each biological neuron is complicated, the process of passing a signal from one neuron to others can be simplified as the description of Simon Cross et. al [22], "when sufficient neurontransmitter has accumulated in the cell body, an action potential is generated". In ANN, the analogue of the neurontransmitter is the inner product of weights and the incoming signals. Given input and weights, the output of a neuron can be expressed as the Formula 2.5 [23].

$$y = \sigma(w^T x + b) \tag{2.5}$$

where $w = (w_1, w_2, ..., w_m)^T$ is the weight vector, $x = (x_1, x_2, ..., x_m)^T$ is the input vector and $b = (b_1, b_2, b_m)^T$ is the bias vector. σ is an element-wise activation function which defines the output behavior of the neuron.

Among all 13 major neural network architectures, feed forward neural network (FFNN) is the simplest one [13]. Figure 2.3 illustrates the structure of a four-layer FFNN. The input layer has three units. Then it is connected with two hidden layer with four and two neurons respectively. Finally there is an output layer with one unit. In this neural network, the input data is a three-dimensional vector and output is a scalar. However, the output can also be multi-dimensional by adding more neurons in the output layer.

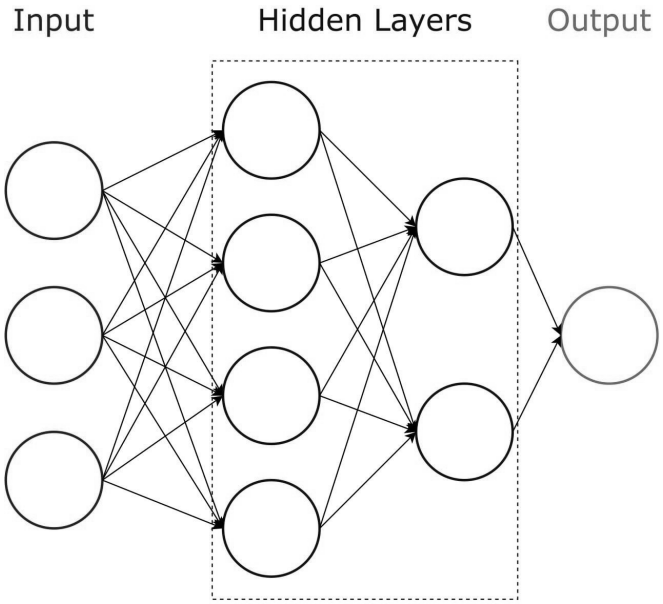


Figure 2.3: Structure of Neural Network

A neural network typically has multiple layers with multiple neurons per layer [23]. Since Formula 2.5 shows the output of a single neuron, the output of a layer can be derived by stacking the weight vectors to form a weight matrix. For the k-th layer, assume there are m neurons. Let $a^{[k-1]} = (a_1, a_2, ..., a_l)^T$ be the input of k-th layer, which is also the output of the $(k-1)$-th layer. $w_1, w_2, ..., w_m$ are weight vectors of m neurons. $b^{[k]} = (b_1, b_2, ..., b_m)^T$ is the bias vector. $\sigma(x)$ is an element-wise activation function. The output of k-th layer $a^{[k]}$

can be expressed as Formula 2.6.

$$a^{[k]} = \sigma(W^{[k]}a^{[k-1]} + b^{[k]}) = \sigma\left(\begin{pmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_m^T \end{pmatrix} a^{[k-1]} + b^{[k]}\right) \tag{2.6}$$

$W^{[k]}$ is the weight matrix of the k-th layer. To build a deep learning model, a critical process is to obtain a set of weight matrices of each layer so the model can have an accurate map from inputs to outputs. The training process involves obtaining weight matrices for each layer and determines whether the current weight matrices are sufficient to solve the problem.

There are various optimization algorithms to train an ANN model. The gradient descent algorithm is one of the most commonly used methods. The gradient descent algorithm contains both forward propagation and backward propagation. Forward propagation refers to the calculation from the input layer to the output layer. For layer k, Formula2.7 is the forward propagation of k-th layer [25].

$$\begin{aligned} Z^{[k]} &= W^{[k]}A^{[k-1]} + b^{[k]} \\ A^{[k]} &= \sigma^{[k]}(Z^{[k]}) \end{aligned} \tag{2.7}$$

The forward propagation will output a predicted vector $A^{[k]}$ which is either the final prediction or servers as the input for $(k+1)$-th layer. For the output layer, the output is compared against the real value y. The loss function, or cost function, is introduced to evaluate the deviation between the current output to real results. For regression problems, a common cost function is the mean square error loss function (also known as $L2$ loss). Given the m-dimensional fact vector y and prediction $\hat{y}$, the cost can be calculated with Formula 2.8 [25]. Therefore, finding a set of weight matrices for each layer in ANN is transformed into finding matrices W and b so that the cost function $J(W,b)$ is minimized.

$$J(W,b) = \frac{(\hat{y}-y)^T(\hat{y}-y)}{2m} \tag{2.8}$$

In order to minimize the cost function, the derivative of $J(W,b)$ over W and b should be calculated. The method traverses the ANN in reverse order and therefore is called backward propagation. The weight matrix and the bias vector at layer k is updated by Formula 2.9 [15].

$$\begin{aligned} W^{[k]} &= W^{[k]} - \alpha\frac{\partial J}{\partial W} \\ b^{[k]} &= b^{[k]} - \alpha\frac{\partial J}{\partial b} \end{aligned} \tag{2.9}$$

where α is the learning rate, which regulates the convergence speed of cost function. Figure 2.4 illustrates the behavior of loss function $J(\alpha)$ under different learning rates [26]. If the learning rate is too large, the model will converge fast but the cost function may "bounce around" the minimal value and result in divergent behavior. If the learning rate is too small, it will take a long time before reaching the minimal.

Leslie Smith proposed a systematical method to find optimal learning rates for neural networks [27]. By running a few epochs where the learning rate increases and plotting the cost function vs learning rate, the optimal learning rate can be obtained. In this thesis, an exponential decay function is utilized for searching for the optimal learning rate. The learning rate finder tool implemented by Pavel Surmenok et. al [28] is utilized to plot and find the optimal learning rate for neural network models.

2.5 Recurrent Neural Network

Recurrent neural networks (RNN) are a special class of ANN. Figure 2.5 shows the typical structure of an RNN [29]. Compared with a conventional ANN, neurons in an RNN are connected along a temporal sequence. For an arbitrary time t, an RNN unit not only takes the input x_t but also the information a_{t-1} from time $t-1$. The equation of an RNN

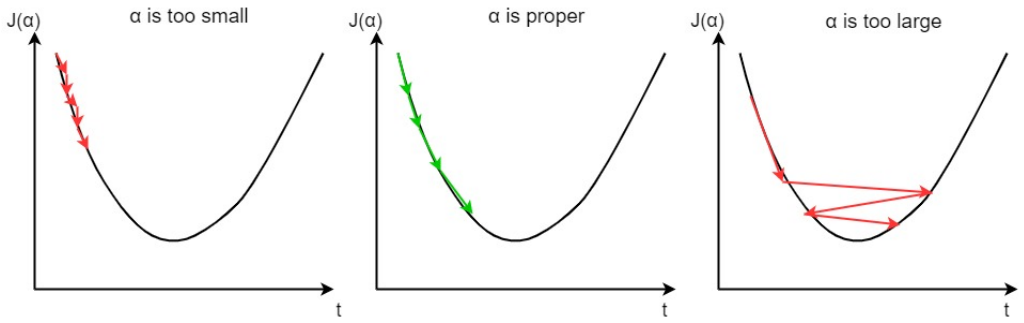


Figure 2.4: Comparison of Different Learning Rates During Training [26]

unit at time t is expressed in Formula 2.10 [30].

$$\begin{aligned} a_t &= \sigma(W_{aa}a_{t-1} + W_{ax}x_t + b_a) \\ o_t &= \sigma(W_{oa}a_t + b_o) \end{aligned} \tag{2.10}$$

To simplify the weight matrices in one RNN unit, as shown in in Formula 2.11, W_{aa} and W_{ax} are combined to form a weight matrix W_a. In this case, a_{t-1} and x_t should be stacked. W_o is abbreviated for $W_o a$.

$$\begin{aligned} W_a &= [W_{aa} \vdots W_a x] \\ [a_{t-1}, x_t] &= \begin{pmatrix} a_{t-1} \\ x_t \end{pmatrix} \end{aligned} \tag{2.11}$$

Then the equation of an RNN unit can be abbreviated as 2.12.

$$\begin{aligned} a_t &= \sigma(W_a[a_{t-1}, x_t] + b_a) \\ o_t &= \sigma(W_o a_t + b_o) \end{aligned} \tag{2.12}$$

There are some limitations of RNNs. If the input sequence x is very long, the computation will be very slow. Moreover, a basic RNN suffers from exploding and vanishing gradient problems when the length of the input sequence is very long. The former is easy to be spotted but the latter can be subtle. Considering the backward propagation process

of the gradient descent algorithm, if the input sequence x is very long, the derivative of the output layer can be hard to propagate back to affect the weight matrices in the earlier layers, which results in a vanishing gradient. Therefore, a basic RNN can not catch long-range dependencies between inputs and outputs.

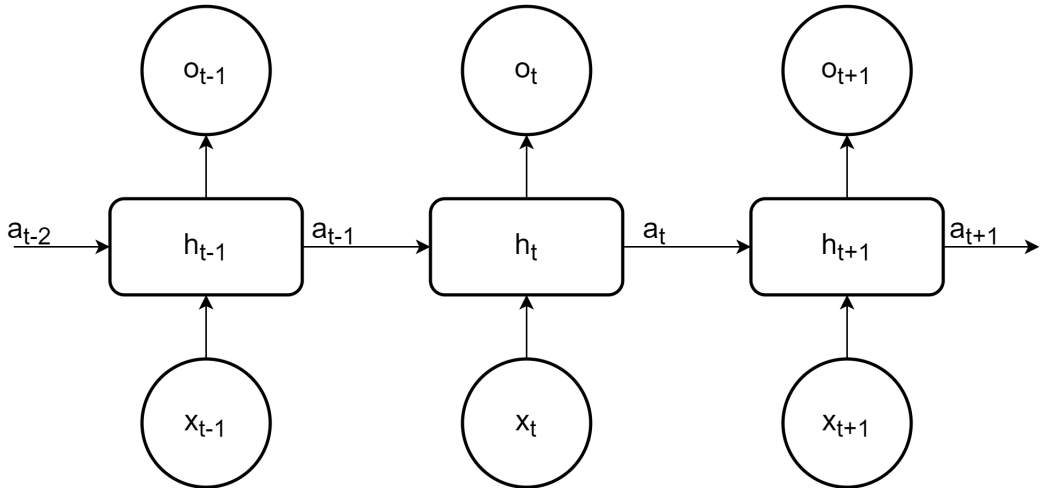


Figure 2.5: Structure of Recurrent Neural Network [29]

In order to solve this problem, the long short term memory (LSTM) architecture was proposed by Hochreiter and Schmidhuber [31]. The equation of the LSTM unit at time t is expressed in Formula 2.13 where $*$ is the Hadamard product of two matrices [31, 32, 33].

According to Formula 2.13, the structure of the LSTM is illustrated in Figure 2.6 [33]. Compared with a basic RNN, an LSTM has three gates to control the behavior of the underlying RNN, namely an input gate, output gate, and forget gate. By regulating these three gates based on the flow of data, the LSTM unit can remember values over arbitrary time intervals. Since speed and heart rate during exercises are time series data, an LSTM model is a potentially competitive candidate for performance forecasting.

$$
\begin{aligned}
\tilde{C}_t &= tanh(W_C[a_{t-1}, x_t] + b_c) \\
\gamma_u &= \sigma(W_u[a_{t-1}, x_t] + b_u) \qquad \textit{(update gate)} \\
\gamma_f &= \sigma(W_f[a_{t-1}, x_t] + b_f) \qquad \textit{(forget gate)} \\
\gamma_o &= \sigma(W_o[a_{t-1}, x_t] + b_o) \qquad \textit{(output gate)} \\
C_t &= \gamma_u * \tilde{C}_t + \gamma_f * C_{t-1} \\
a_t &= \gamma_o * tanh(C_t)
\end{aligned}
\tag{2.13}
$$

[33]

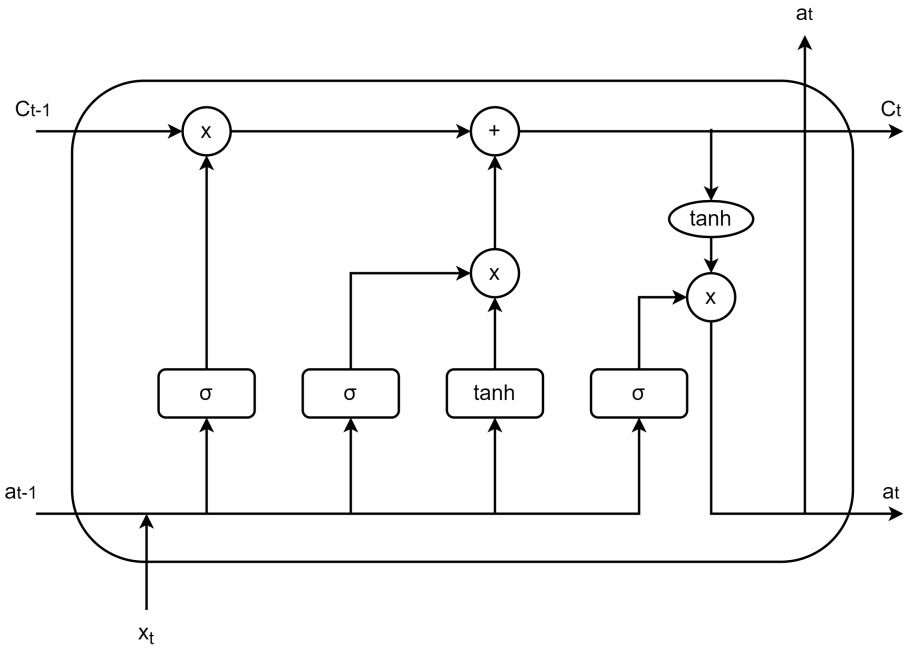


Figure 2.6: LSTM Architecture
[33]

Chapter 3

Heart Rate Forecasting

In this chapter, four learning algorithms, namely random forest, feed forward neural networks, simple recurrent neural networks, and long short term memory, are utilized to build heart rate forecasting models to predict an athlete's heart rate on a proposed course. In the first section, the dataset and feature selection are reviewed. Then the results of four heart rate forecasting models are discussed respectively. Moreover, the mean squared error (MSE) of four models are compared against each other. Finally, we will compare our models with Ni's model [11] and Luo's model [12] trained on our dataset.

3.1 The Dataset and Feature Selection

For heart rate forcasting, the dataset used in this thesis contains the grade of course, speed, heart rate, altitude, cadence, and distance at each second. Before selecting features, the significance of these sequences should be determined. Spearman's correlation coefficients (ρ) of these sequences are listed in Table 3.1.

According to Table 3.1, the grade of course, cadence, speed and altitude have a more significant influence on heart rate than other factors. Therefore, these four factors are selected as the features of the heart rate forecasting model.

Features	Spearman correlation coefficient
speed	0.1826
grade of course	0.2524
cadence	0.2389
distance	0.0466
altitude	0.2586

Table 3.1: Spearman correlation coefficient between factors and heart rate

The heart rate data is measured with an electrode sensor on a chest strap. During cycling, the sensor might situate far apart from the skin due to motion, which can cause error

on heart rate data. Therefore, in order to reduce the error and smooth the heart rate, the heart rate sequence is processed with a window average of 60 seconds. To be consistent with the heart rate data, the features, namely, the grade of course, cadence, speed and altitude, are also processed with the same window average process.

The data set is split between a training set and test set. By convention, a 80% vs 20% train-test split ratio is utilized to split data into the training set and test set. For the LSTM-based model, considering the order dependence in heart rate data, the training set is the first 80% of the data and the test set is the remaining 20%, rather than a random selection. In order to compare the results with LSTM-based models, the same train-test split process is also carried out for the random forest model.

3.2 Random Forest Heart Rate Model

The heart rate forecasting result and of the random forest model is in Figure 3.1. The predicted value follows the trend of the heart rate and shows the model learned some patterns in the heart rate sequence. However, the predicted heart rate shows a large error around time 8900s. Moreover, for the time range from 8600s to 8700s, the predicted heart rate remains stable while the real heart rate shows a sharp decreasing trend. The error in percentage between predicted heart rate and real hear rate is shown in Figure 3.2.

Data Set	Mean Squared Error
Training Set	6.1267
Test Set	215.8963

Table 3.2: Mean Squared Error in Training and Test Set

The source of error consists of three parts. The first part is that there are still some external factors that are not included in the data set such as the change of wind speed and directions during cycling, the traffic density at different parts of courses etc. The second part is some internal features such as the mental activities of the cyclist over time. For example, the cyclist would be anxious if he or she fell behind during the cycling and the

heart rate would also increase in this case. These features can have an influence on the heart rate but they are hard to measure and express quantitatively. The third part is that the random forest model may overfit the training set and provide erroneous predictions. In order to check for overfitting of the random forest model, the mean squared error (MSE) is calculated in Table 3.2. The MSE for the test set is approximately 35 times larger than that of training set which indicates that the random forest model severely overfit to the training set.

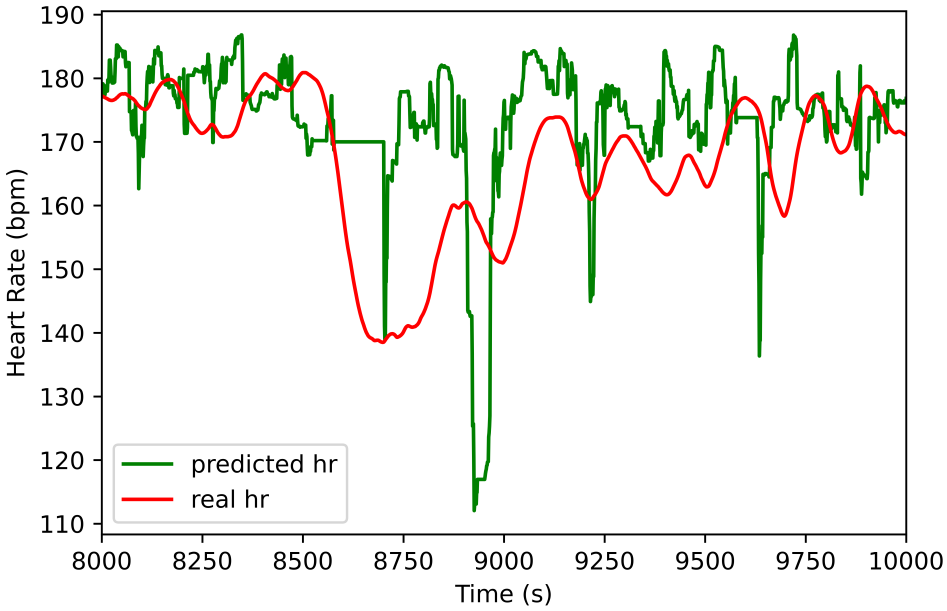


Figure 3.1: Heart Rate Prediction of Random Forest

3.3 FFNN heart Rate Model

Due to the overfitting problems of random forest, feed forward neural network (FFNN) models with different hyper-parameters were built. The hyper parameters of the FFNN models are listed in Table 3.3.

An important part for training a neural network is to find an optimal learning rate. To search the learning rate systematically, Leslie Smith's method was utilized [27]. By running

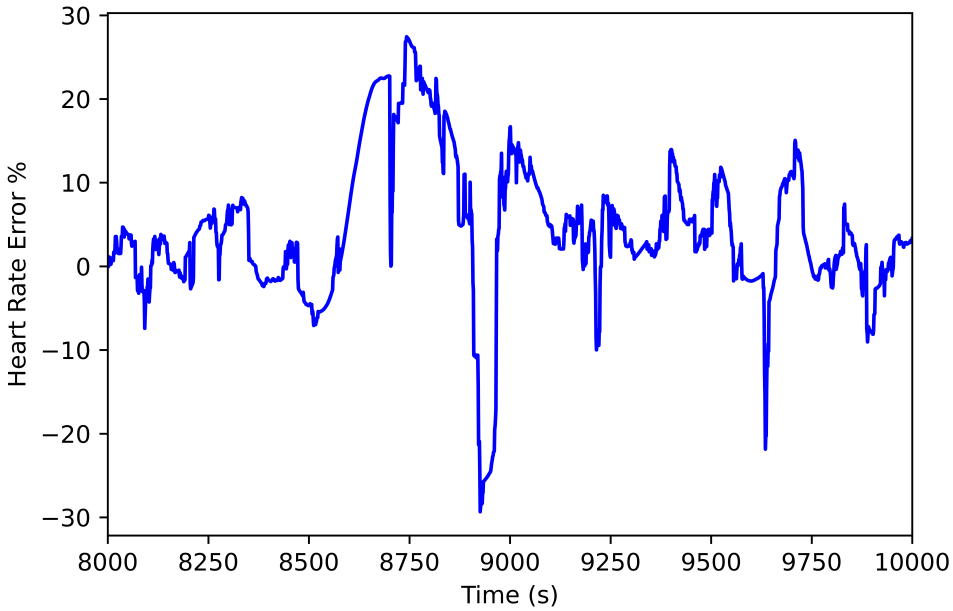
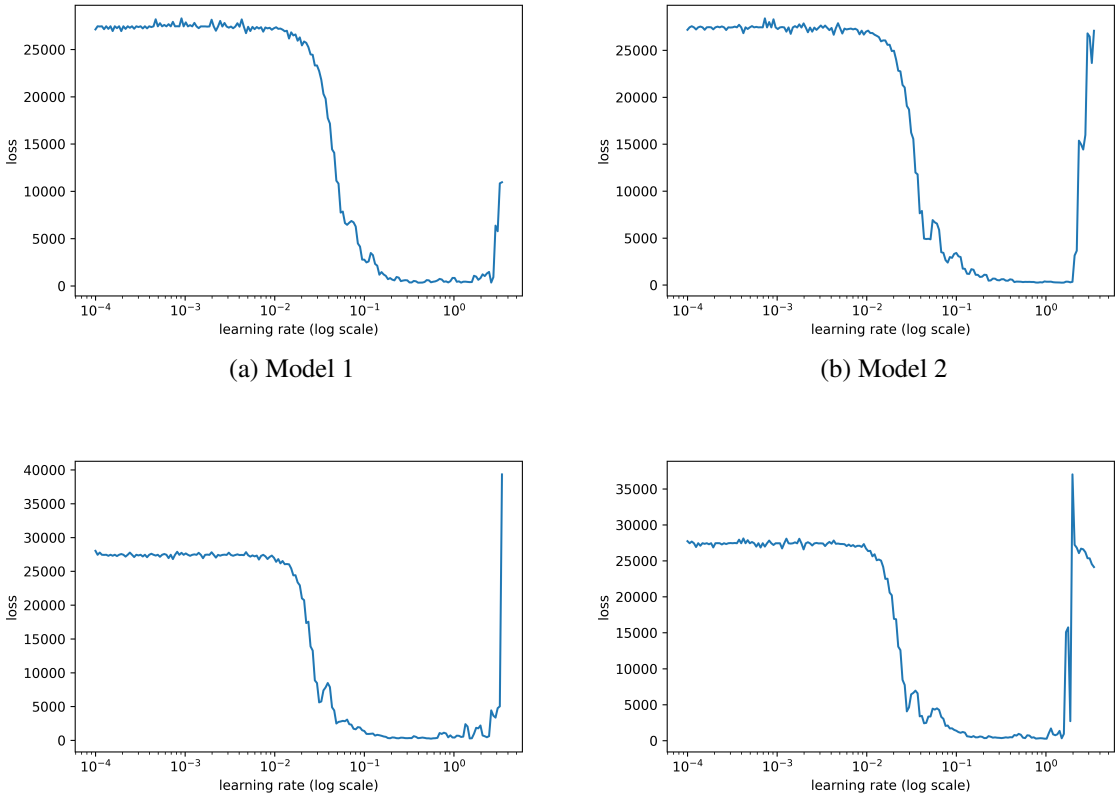


Figure 3.2: Heart Rate Prediction Error of Random Forest

Model Number	Number of layers	Number of Neurons in Each Layer
1	2	(5, 1)
2	2	(10, 1)
3	3	(5, 5, 1)
4	3	(10, 5, 1)

Table 3.3: Hyper-parameters of FFNN models

at each learning rate for 10 epochs, the loss vs learning rate curves are plotted as shown in Figure 3.3. According to the curves, the optimal learning rate for these four ANN models are 3×10^{-2}.



(a) Model 1

(b) Model 2

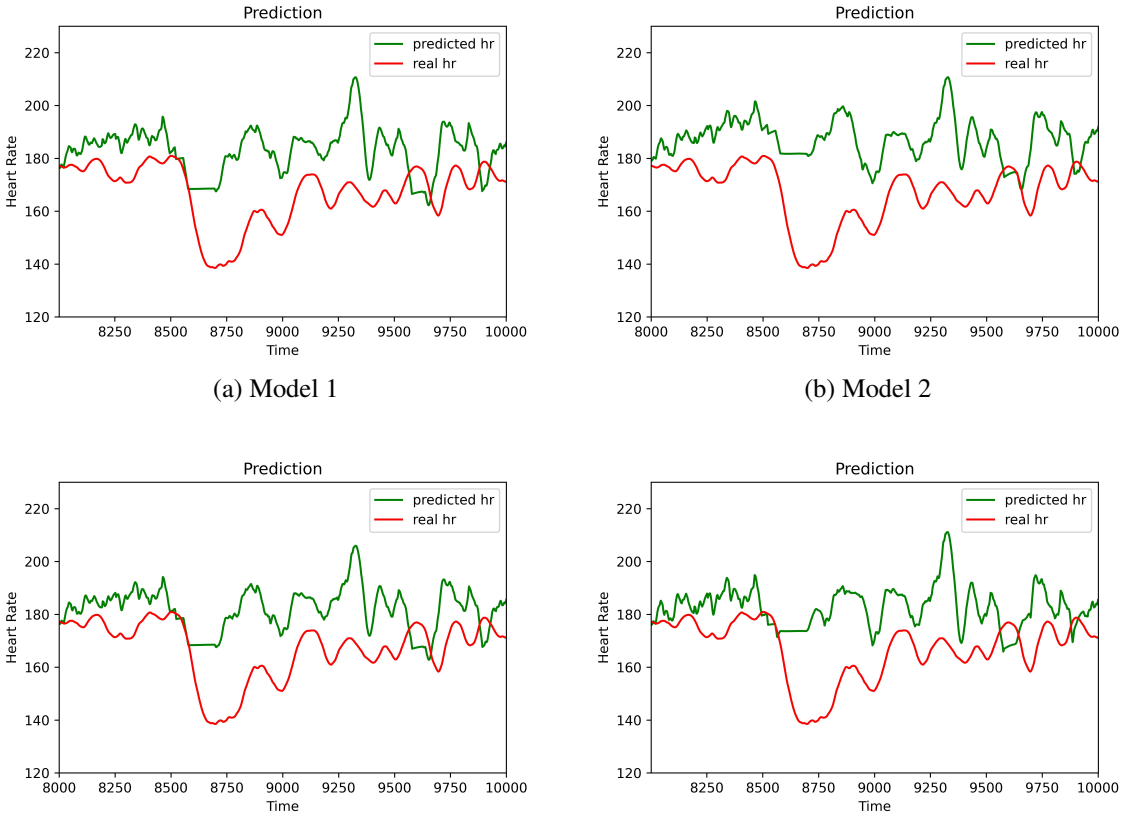
(c) Model 3

(d) Model 4

Figure 3.3: Optimal Learning Rate Search for FFNN Heart Rate Models

Figure 3.4 shows the predicted heart rate and the real heart rate of these four FFNN models. It shows that FFNN models can follow the trend of the cyclist's heart rate on the specific course. The error and error percentage are plotted in Figure 3.5 and Figure 3.6 respectively. Models 3 and 4 show large errors from 8500s to 9000s. In order to investigate the error source, MSEs of the training set and test set are calculated for these four models and are shown in Table 3.4. Model 3 shows an obvious overfitting on the training set while Model 1 and 2 show large bias on the training set. Among these four models, Model 4 shows relatively low bias and variance. Compared with random forest models, the FFNN

model show less variance on the test set but the accuracy is similar.



(a) Model 1

(b) Model 2

(c) Model 3

(d) Model 4

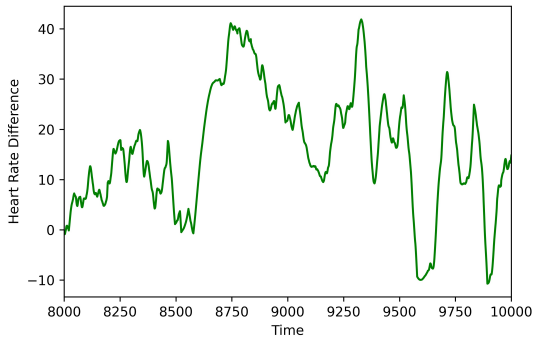
Figure 3.4: Heart Rate Forecasting of FFNN Models

Model Number	MSE on Training Set	MSE on Test Set
1	242.89	405.11
2	238.81	568.08
3	262.47	364.91
4	233.25	402.34

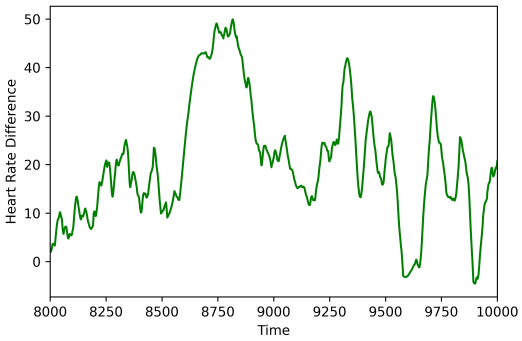
Table 3.4: MSE of FFNN models

3.4 Simple RNN model

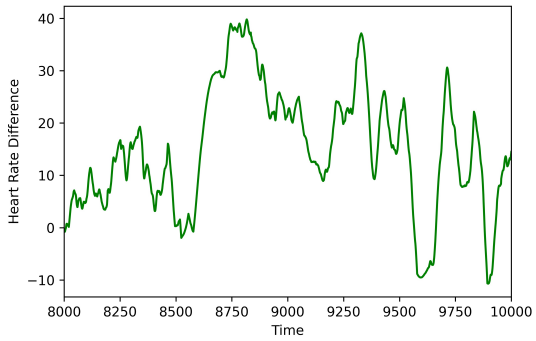
Unlike FFNNs, recurrent neural networks (RNN) can utilize heart rate data in the past to forecast the heart rate at the current moment. The architecture of a simple RNN model



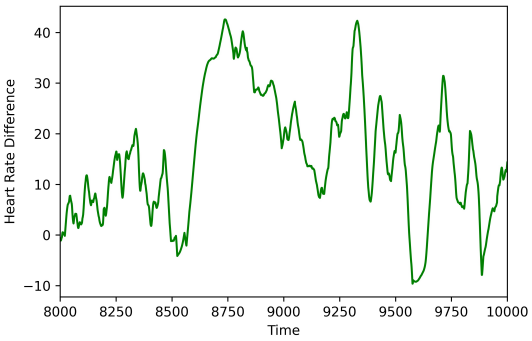
(a) Model 1



(b) Model 2

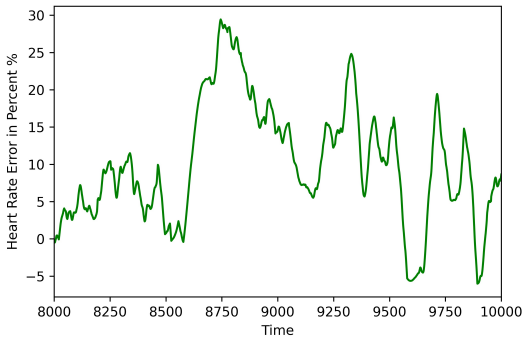


(c) Model 3

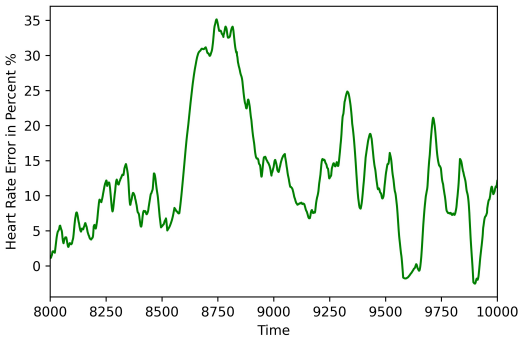


(d) Model 4

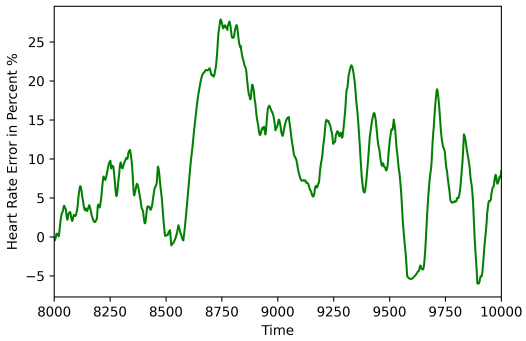
Figure 3.5: Heart Rate Forecasting Error of FFNN Models



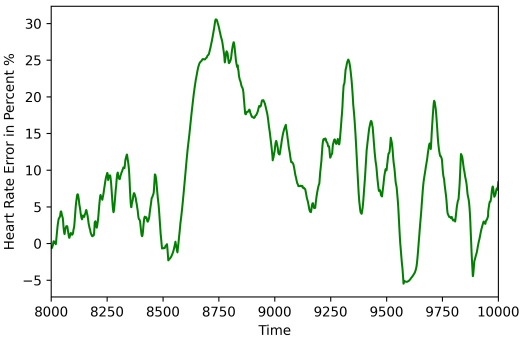
(a) Model 1



(b) Model 2



(c) Model 3



(d) Model 4

Figure 3.6: Heart Rate Forecasting Error in Percent of FFNN Models

for heart rate forecasting is shown in 3.7. The number of simple RNN layers, m, and the number of fully connected layers, also known as dense layers, n, are two important hyper-parameters of a simple RNN heart rate forecasting model. The heart rate sequence is divided by the max heart rate.

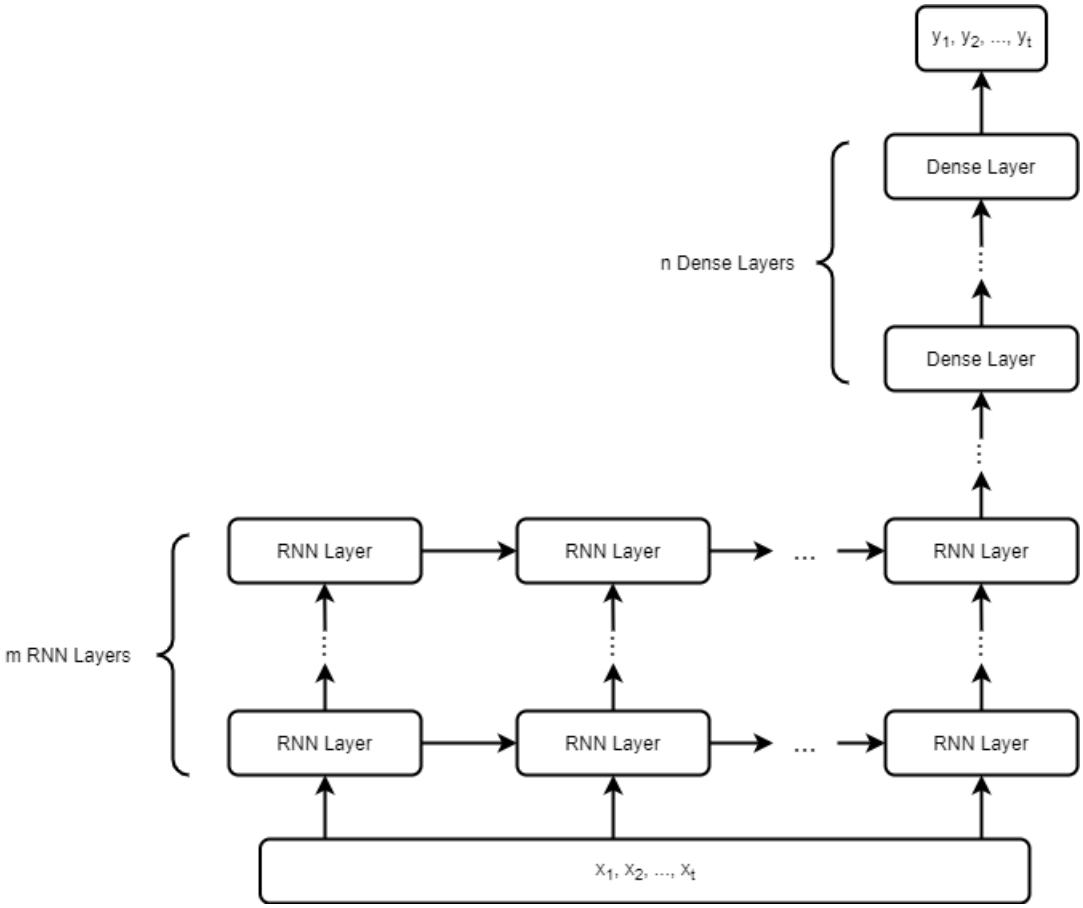


Figure 3.7: Structure of Simple RNN Heart Rate Forecasting Model

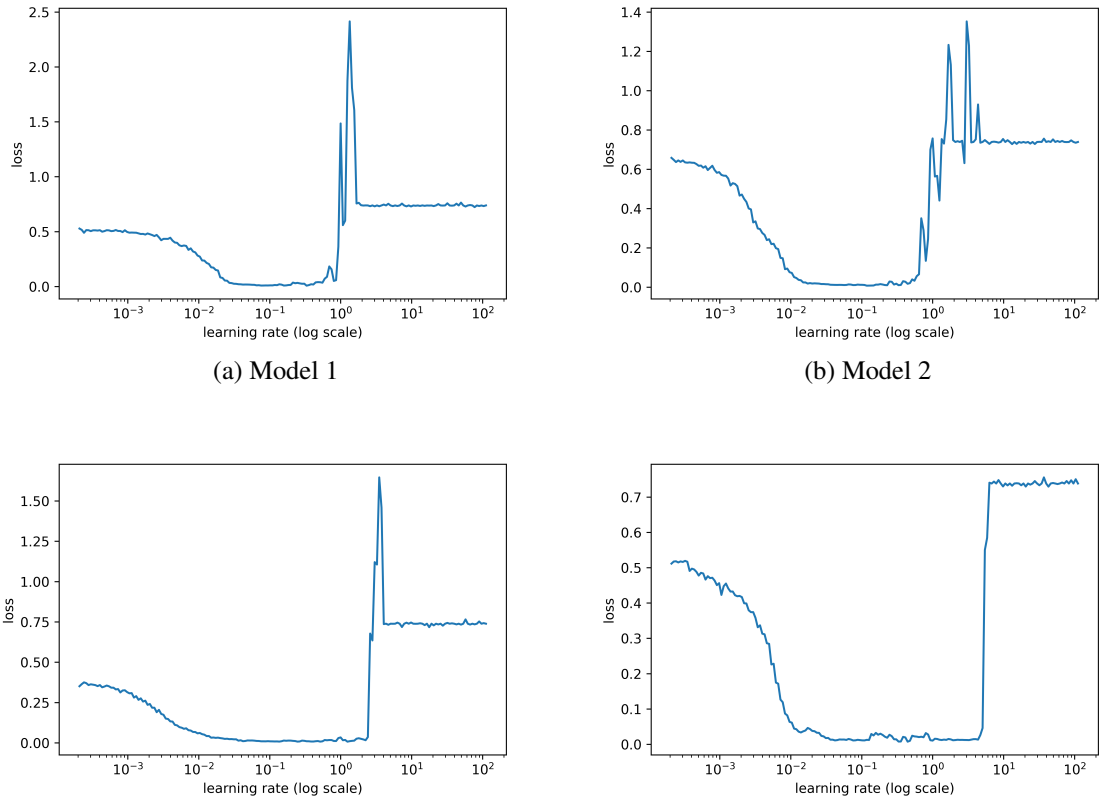
Simple RNN Models with different hyper parameters were built and their Hyper Parameters are listed in Table 3.5. The optimal learning rates were searched via Smith's method and an exponential decay learning rate function was utilized for learning rate searching. The learning rate vs loss curves are plotted in Figure 3.8. From the learning rate vs loss curves shown in Figure 3.8, the optimal learning rate was selected to be 2×10^{-1}.

Table 3.6 shows the MSEs of four simple RNN models with different sets of hyper-parameters on the training set and test set. For model 3, the MSE on the test set is much

Model Number	Number of Dense Layer	Number of Simple RNN Layers
1	1	1
2	1	2
3	2	1
4	2	2

Table 3.5: Hyper-parameters of Simple RNN models

larger than that on the training set, which indicates severe over-fitting.



(a) Model 1

(b) Model 2

(c) Model 3

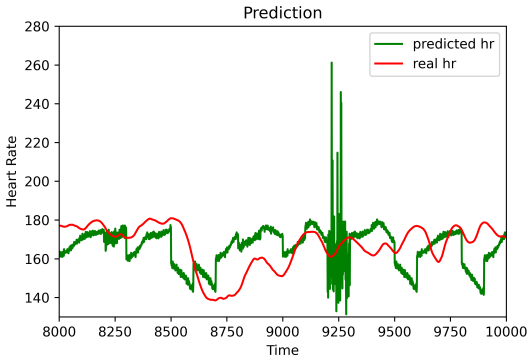
(d) Model 4

Figure 3.8: Optimal Learning Rate Search for Simple RNN Models

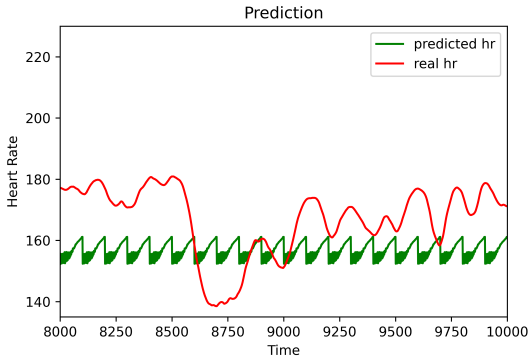
The heart rate of the athlete is predicted in 3.9 and the error and error in percentage are shown in 3.10 and 3.11. Compared with FFNN models, RNN models showed less

Model Number	MSE on Training Set	MSE on Test Set
1	275.11	248.83
2	628.48	237.68
3	210.31	655.18
4	1245.66	964.61

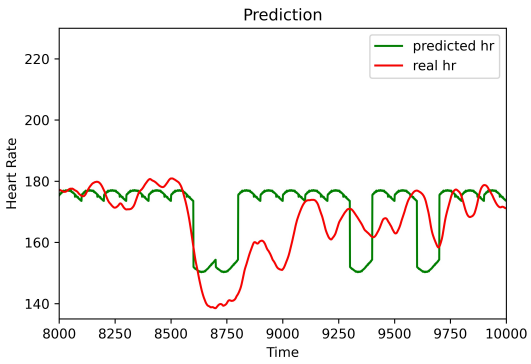
Table 3.6: MSE of Simple RNN models



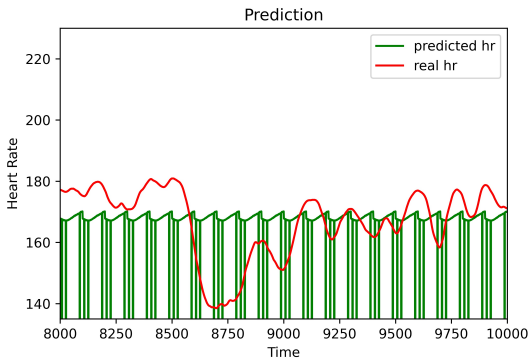
(a) Model 1



(b) Model 2

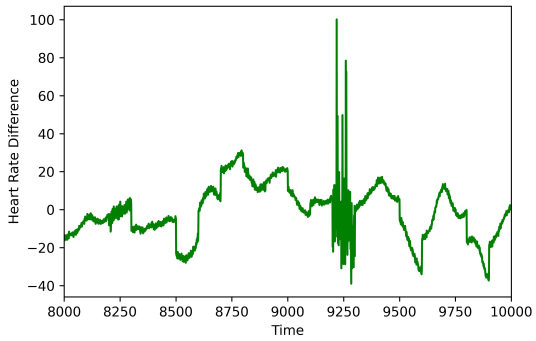


(c) Model 3

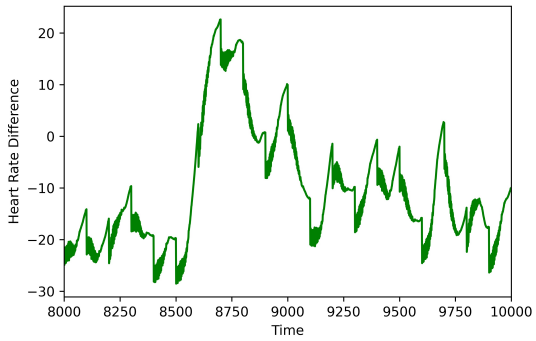


(d) Model 4

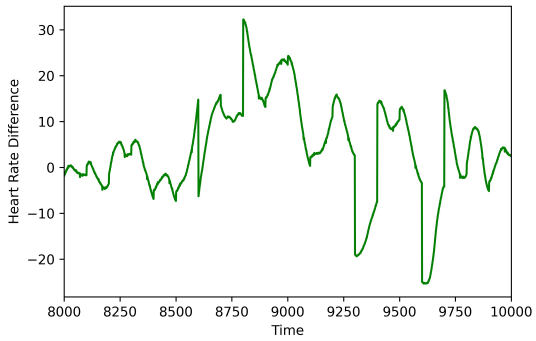
Figure 3.9: Heart Rate Forecasting of Simple RNN Models



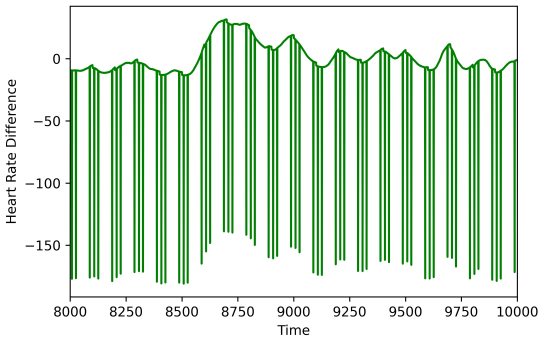
(a) Model 1



(b) Model 2

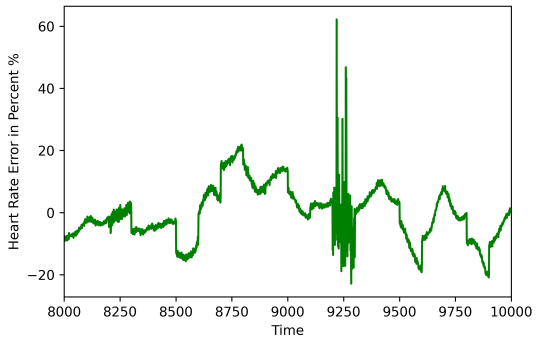


(c) Model 3

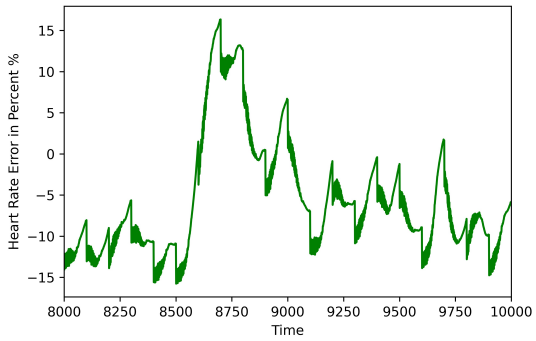


(d) Model 4

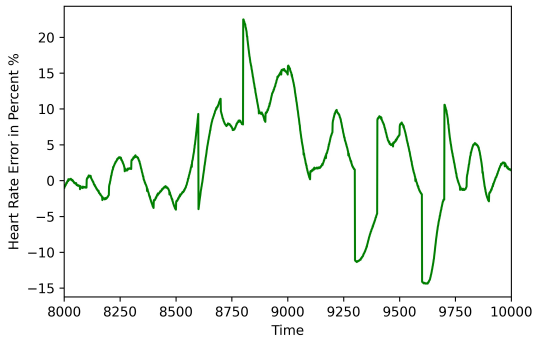
Figure 3.10: Heart Rate Forecasting Error of Simple RNN Models



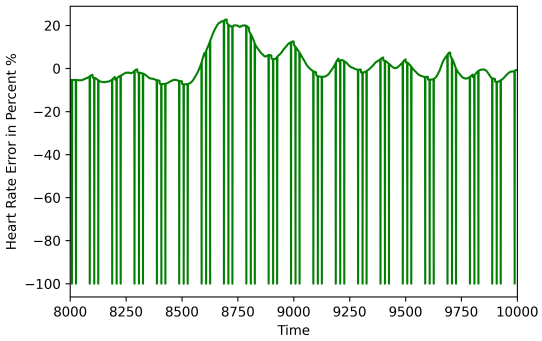
(a) Model 1



(b) Model 2



(c) Model 3



(d) Model 4

Figure 3.11: Heart Rate Forecasting Error in Percent of Simple RNN Models

3.5 LSTM model

As mentioned in Chapter 2, simple RNN models suffers from vanishing gradient problems which limits its application when the sequence is very long. In order to solve this, LSTM models are utilized. The hyper-parameters of LSTM model is shown in Table 3.7. For two-layer LSTM models, error of results are

Utilizing the optimal learning finder, the learning rate vs loss function curves are shown in Figure 3.12. The optimal learning rate for models with only one LSTM layer, namely model 1 and model 3, is 1×10^{-1} and the optimal learning rate for models with two LSTM layers, namely model 2 and model 4, is 1×10^{1}.

Model Number	Number of Dense Layer	Number of LSTM Layers
1	1	1
2	1	2
3	2	1
4	2	2

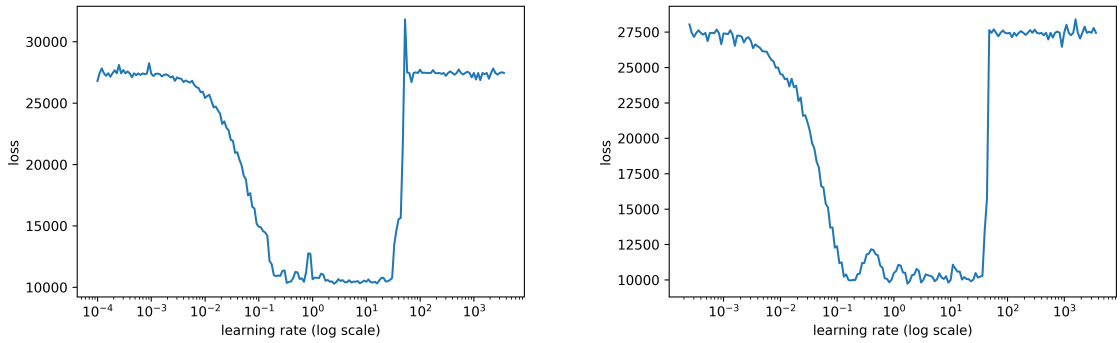
Table 3.7: Hyper-parameters of LSTM models

The result of four LSTM models with different hyper-parameters are shown in Figure 3.13. The MSE on the training set and test set is shown in Table 3.8. The heart rate error and error in percentage are shown in Figure 3.14 and Figure 3.15 respectively.

Model Number	MSE on Training Set	MSE on Test Set
1	141.17	200.50
2	1240.60	1015.87
3	62.47	196.61
4	1480.36	1470.36

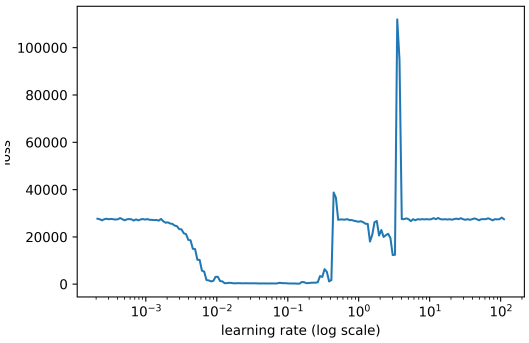
Table 3.8: MSE of LSTM models

From Table 3.10, the models with two LSTM layers show large MSE over both the training set and test set. The larger error may be ascribed to the limitation of data that is not sufficient to training a neural network with two LSTM layers.

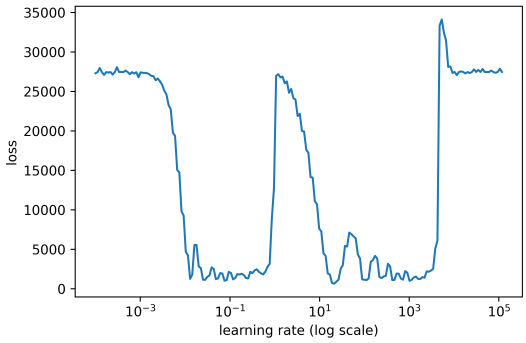


(a) Model 1

(b) Model 2

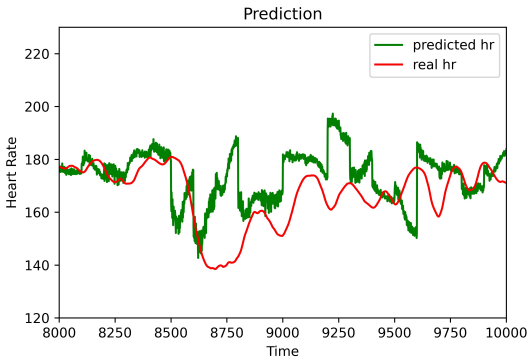


(c) Model 3

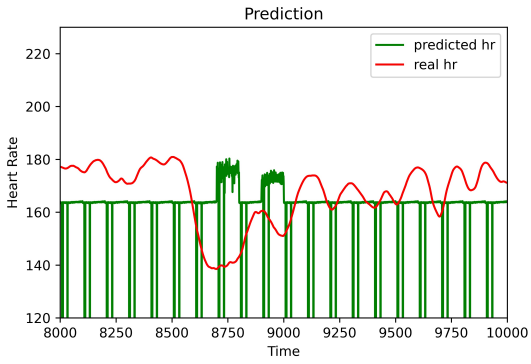


(d) Model 4

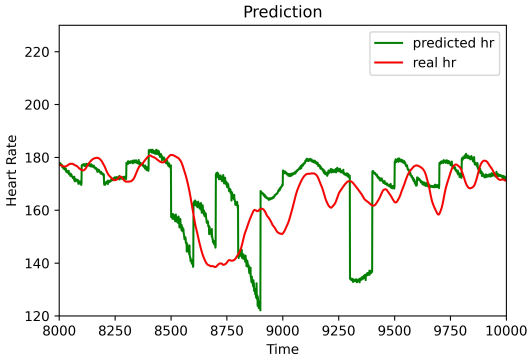
Figure 3.12: Optimal Learning Rate Search for LSTM Models



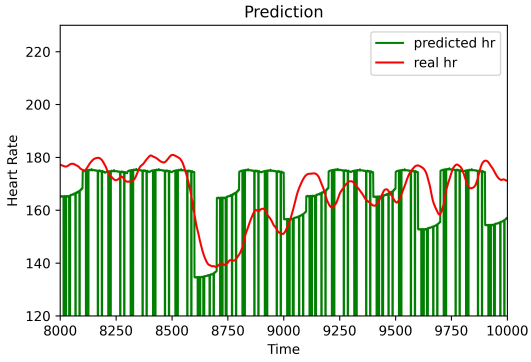
(a) Model 1



(b) Model 2

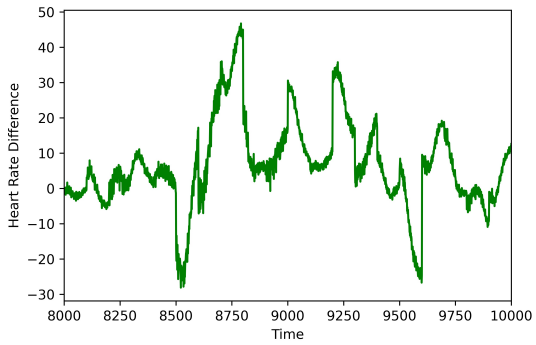


(c) Model 3

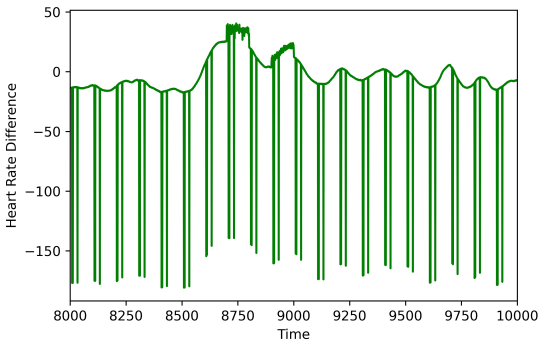


(d) Model 4

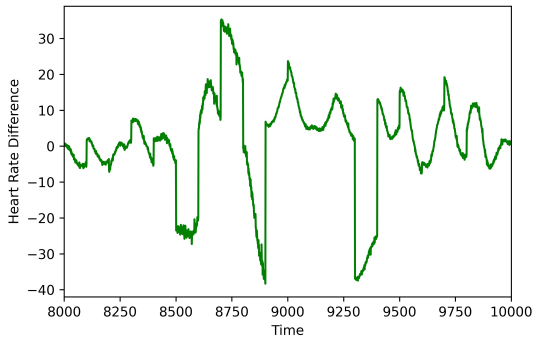
Figure 3.13: Heart Rate Forecasting of LSTM Models



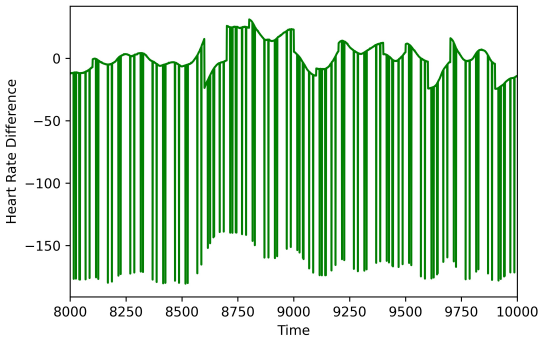
(a) Model 1



(b) Model 2

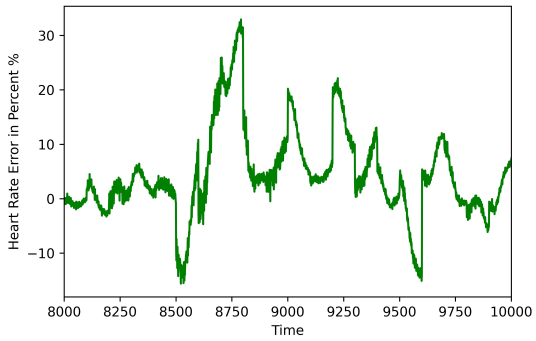


(c) Model 3

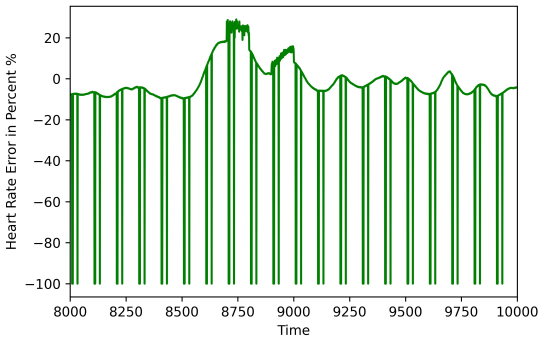


(d) Model 4

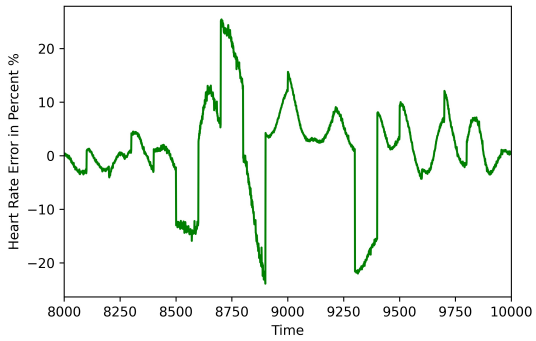
Figure 3.14: Heart Rate Forecasting Error of LSTM Models



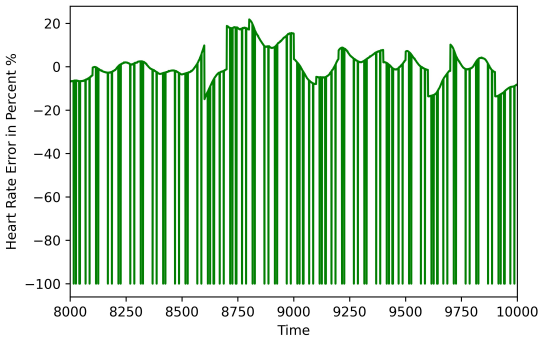
(a) Model 1



(b) Model 2



(c) Model 3



(d) Model 4

Figure 3.15: Heart Rate Forecasting Error in Percent of LSTM Models

3.6 Comparison with Other Heart Rate Models

Considering all four models above, two LSTM models with only one LSTM layer performs better than the other three types of model. In this section, Luo's LSTM model [12] and Ni's LSTM-based model [11] are selected. Specifically, only the structure of their models are utilized and some layers, such as encoding layers in Ni's model, are removed because the heart rate forecasting is the main focus. The structure of models are listed in Table 3.9.

Models	Layer Structure of Heart Rate Forecasting Model
Ni's model	LSTM + Dense + Dropout
Luo's model	LSTM + Dropout + LSTM + Dropout
LSTM model 1	LSTM + Dense
LSTM model 3	LSTM + Dense + Dense

Table 3.9: Structure of Heart Rate Forecasting Models

Their models were first trained on the same training set as our heart rate forecasting models and then were tested on the same test set. The MSE of four models is shown in Table 3.10. The forecasted heart rate is shown in Figure 3.16.

Models	MSE on Training Set	MSE on Test Set
Ni's model	1613.89	288.23
Luo's model	261.20	203.86
LSTM model 1	141.17	200.50
LSTM model 3	62.47	196.61

Table 3.10: MSE of LSTM models

While Ni's model showed less error on both training set and test set, the forecasted heart rate is close to a constant, which is not desirable. Luo's model and our two LSTM models show similar heart rate trend.

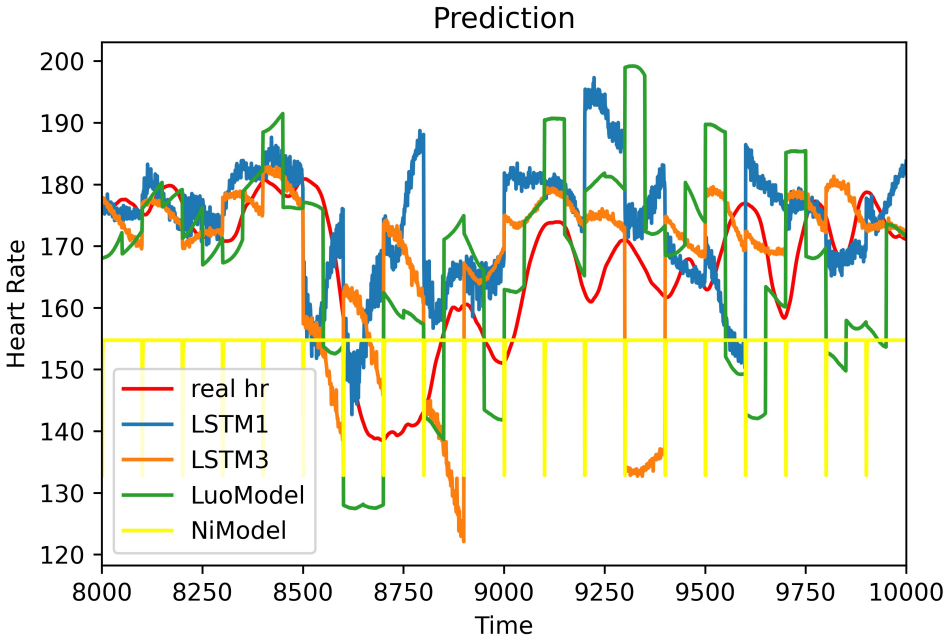


Figure 3.16: Comparison of Different Heart Rate Forecasting Models

Chapter 4

Speed Forecasting

In this chapter, similar to the heart rate forecasting experiments, random forests, feed forward neural networks (FFNNs), simple recurrent neural networks (RNNs), and long short term memory (LSTM) models are utilized to build speed forecasting models to predict an athlete's speed at each second on a specific course. The feature selection process is carried out in the first section. Then the results for the four speed forecasting models are discussed respectively and the mean squared error (MSE) is selected as the comparison metric among speed forecasting models. The error percentage is not analyzed in this chapter since the speed can be zero. In the last section, our models are compared against Ni's model.

4.1 The Dataset and Feature Selection

For speed forecasting, the dataset contains the grade of the course, speed, heart rate, altitude, cadence, and distance at each second. Spearman's correlation coefficients (ρ) of these features are listed in Table 4.1.

According to Table 4.1, the grade of course, cadence, heart rate and altitude have a more significant influence on heart rate than other features. Therefore, these four factors are selected as the features of the heart rate forecasting model.

Factors	Spearman correlation coefficient
heart rate	0.183
grade of course	-0.147
cadence	0.482
distance	-0.058
altitude	0.192

Table 4.1: Spearman Correlation Coefficient between Features and Speed

4.2 Random Forest Model

The speed forecasting result of the random forest model is shown in Figure 4.1 and the error is shown in Figure 4.2. The random forest model captures the speed patterns from 8000s to 10000s. The mean squared error (MSE) on the training set and test set are 0.0010 and 0.3967, which indicates there is a severe over-fitting for random forest model.

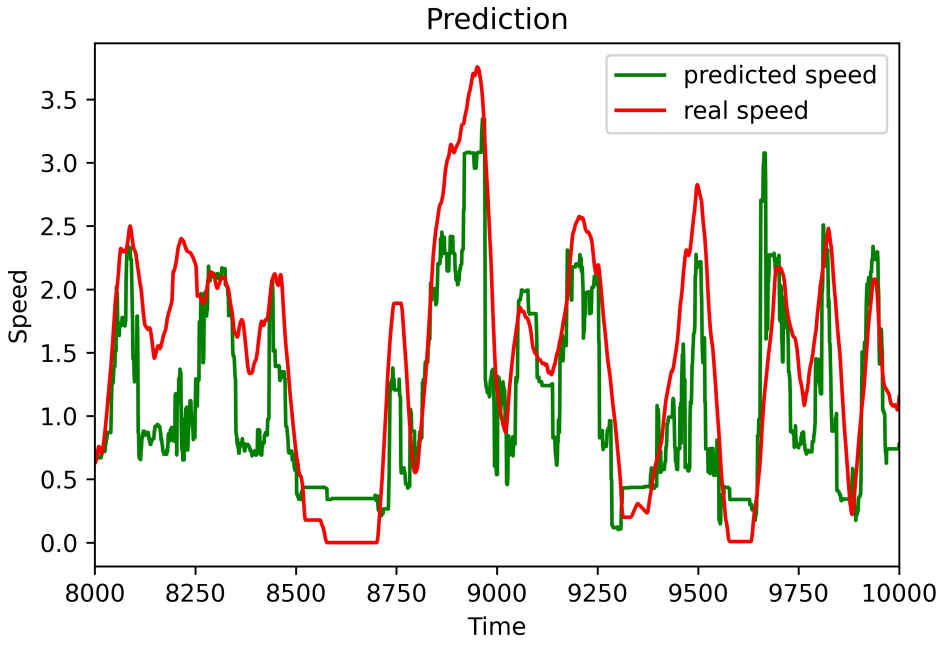


Figure 4.1: Speed Prediction of Random Forest Model

4.3 FFNN Model

In order to have a less overfit model, feed forward neural network models are utilized to forecast the speed of cyclist. The hyper parameters of FFNN model are shown in Table 4.2.

The optimal learning rate finder is also utilized to search for an optimal learning rate. The learning rate vs the loss curves are plotted in Figure 4.3. The optimal learning rate for FFNN models is 1×10^{-1}.

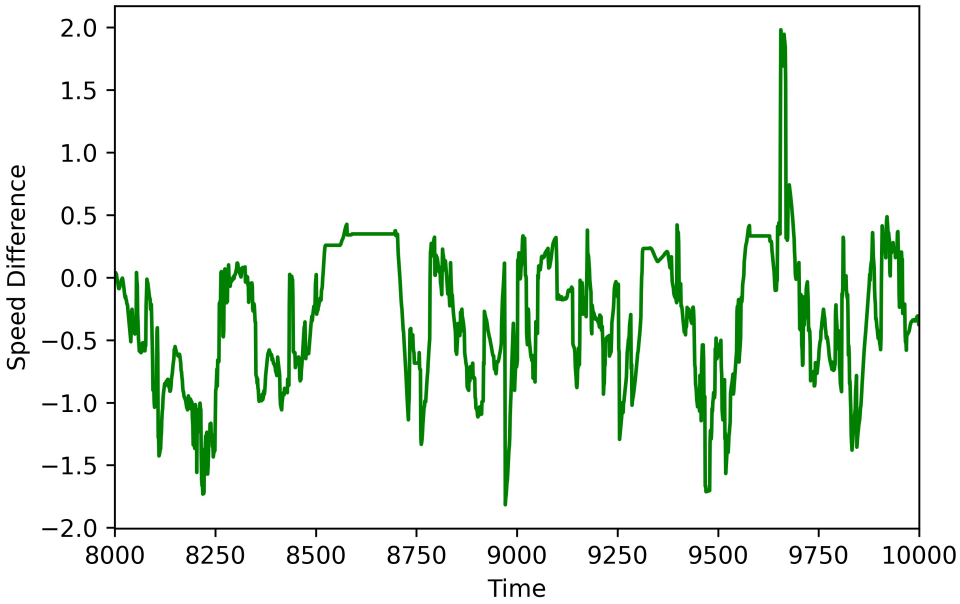
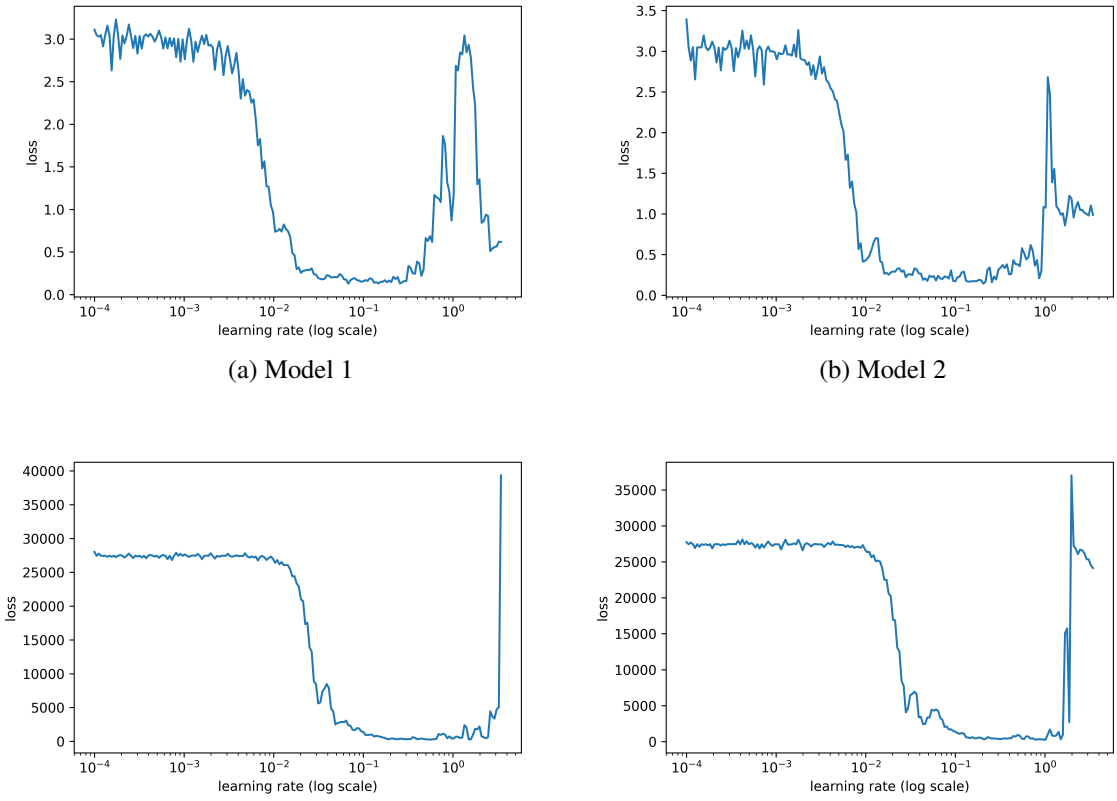


Figure 4.2: Speed Prediction Error of Random Forest Model

Model Number	Number of layers	Number of Neurons in Each Layer
1	2	(5, 1)
2	2	(10, 1)
3	3	(5, 5, 1)
4	3	(10, 5, 1)

Table 4.2: Hyper-parameters of FFNN speed forecasting models

Results from the FFNN models are illustrated in Figure 4.4 and the error of these four models is plotted in Figure 4.5. The MSEs for these four FFNN models are calculated and shown in Table 4.3. The forecasted speed and MSEs of these four models are relatively close. The gap between the prediction and real speed can be ascribed the limitation of FFNN models that they do not take the speed history into consideration.



(a) Model 1

(b) Model 2

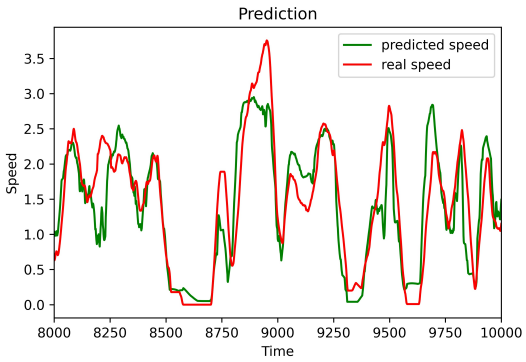
(c) Model 3

(d) Model 4

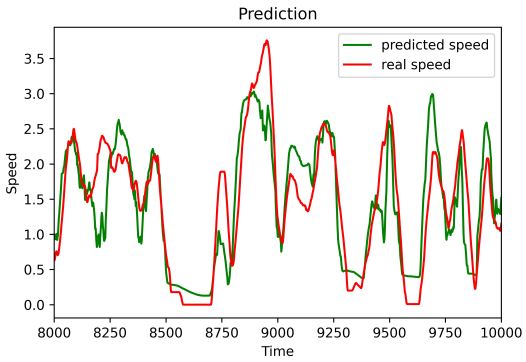
Figure 4.3: Optimal Learning Rate Search for FFNN Speed Forecasting Models

FFNN Models	MSE on Training Set	MSE on Test Set
Model 1	0.1128	0.216
Model 2	0.1078	0.248
Model 3	0.1249	0.234
Model 4	0.1152	0.265

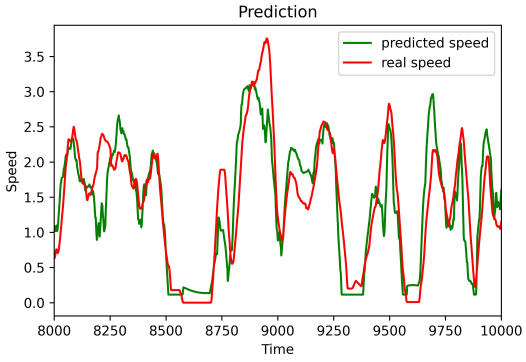
Table 4.3: MSE of FFNN Speed Models



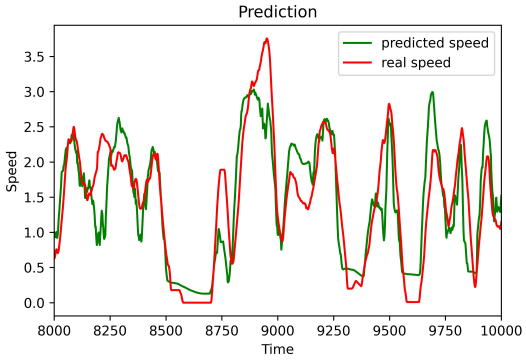
(a) Model 1



(b) Model 2

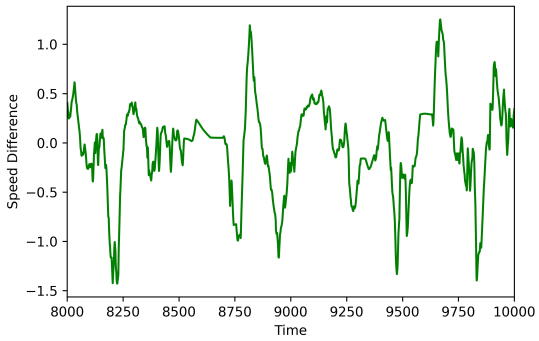


(c) Model 3

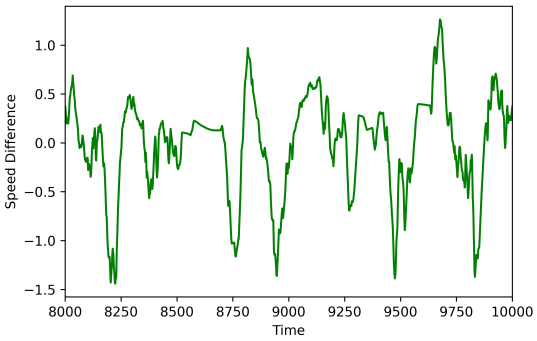


(d) Model 4

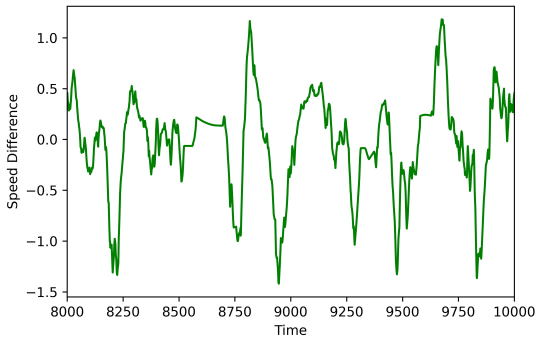
Figure 4.4: Speed Forecasting of FFNN Models



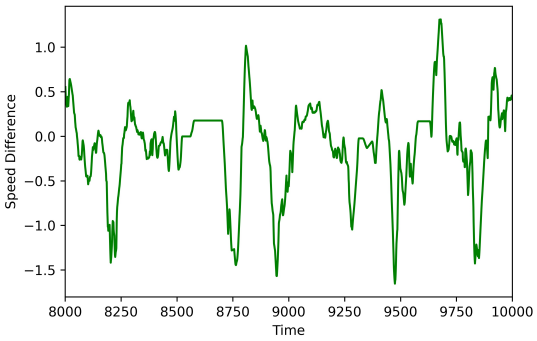
(a) Model 1



(b) Model 2



(c) Model 3



(d) Model 4

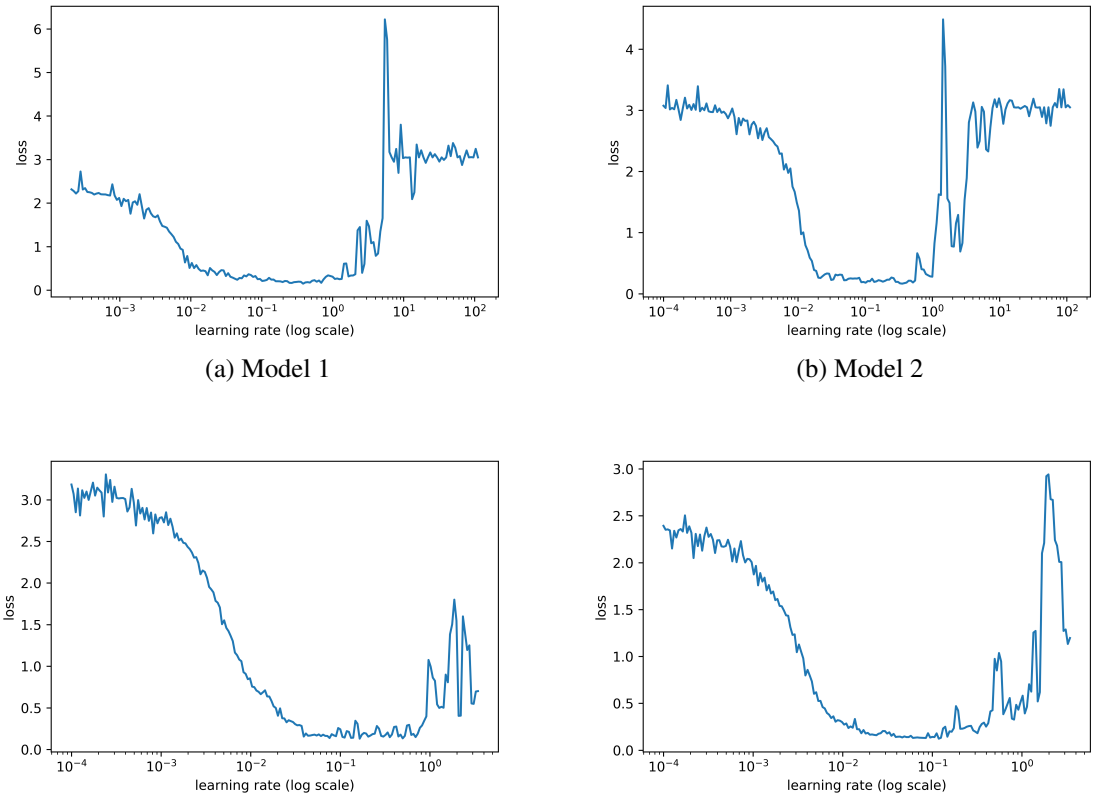
Figure 4.5: Speed Forecasting Error of FFNN Models

4.4 RNN Model

In order to take the historical speed into consideration, simple RNN models are designed and built. The hyper-parameters of these simple RNN model are listed in Table 4.4. The optimal learning rates are searched by learning rate finder and the loss vs the learning rate curves are shown in Figure 4.6. The optimal learning rate can be 1×10^{-1}.

Model Number	Number of Dense Layer	Number of Simple RNN Layers
1	1	1
2	1	2
3	2	1
4	2	2

Table 4.4: Hyper-parameters of Simple RNN models



(a) Model 1

(b) Model 2

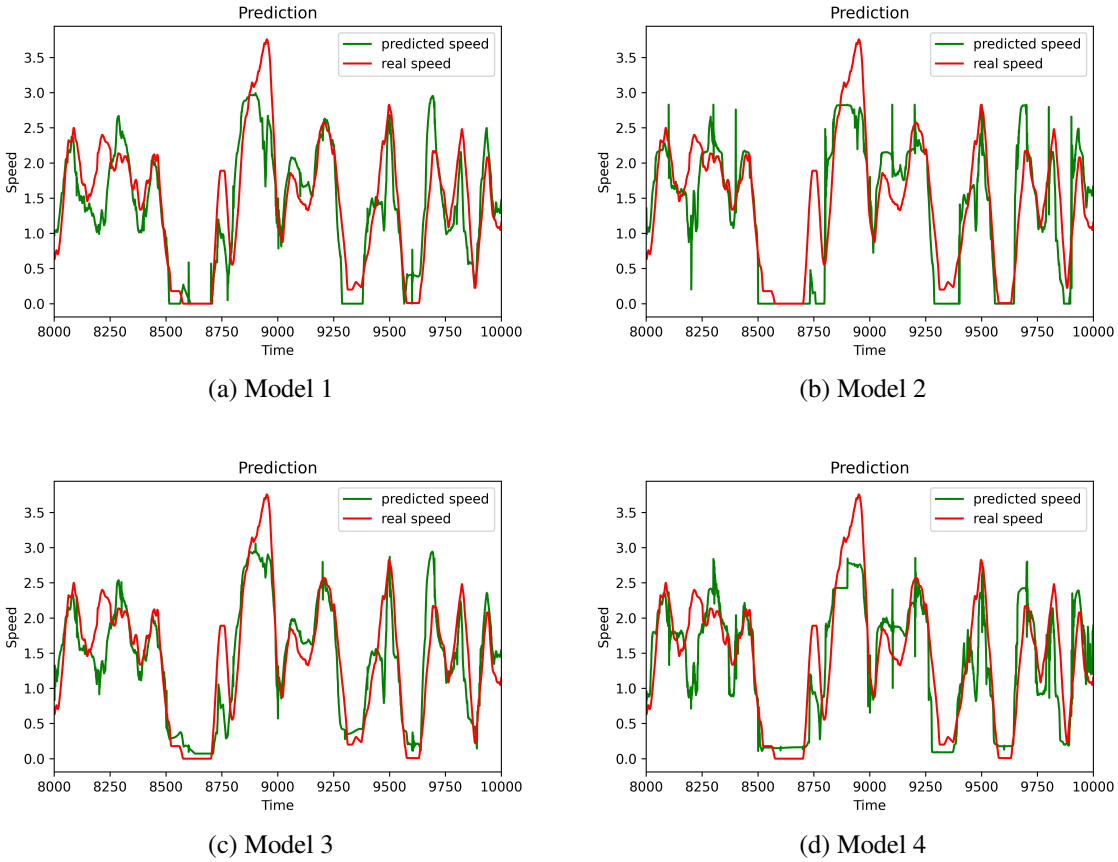
(c) Model 3

(d) Model 4

Figure 4.6: Optimal Learning Rate Search for RNN Speed Forecasting Models

The results and errors are plotted in Figure 4.7 and Figure 4.8. MSEs are calculated and

listed in Table 4.5. Model 3 has the lowest MSE on both the training set and test set among four RNN models.



(a) Model 1

(b) Model 2

(c) Model 3

(d) Model 4

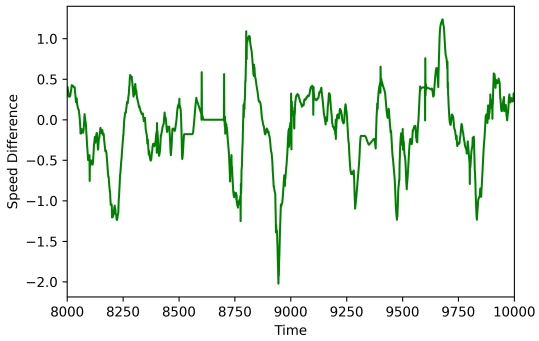
Figure 4.7: Forecasting Results for RNN Speed Forecasting Models

RNN Models	MSE on Training Set	MSE on Test Set
Model 1	0.121	0.235
Model 2	0.155	0.296
Model 3	0.119	0.196
Model 4	0.133	0.243

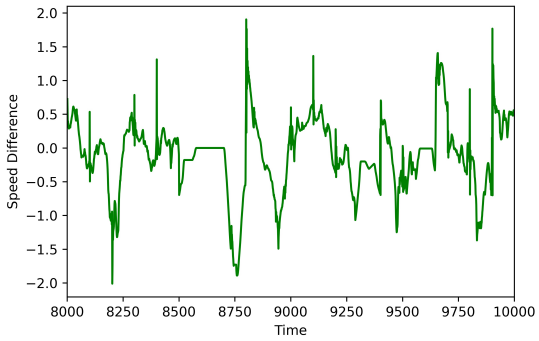
Table 4.5: MSEs of RNN Speed Models

4.5 LSTM model

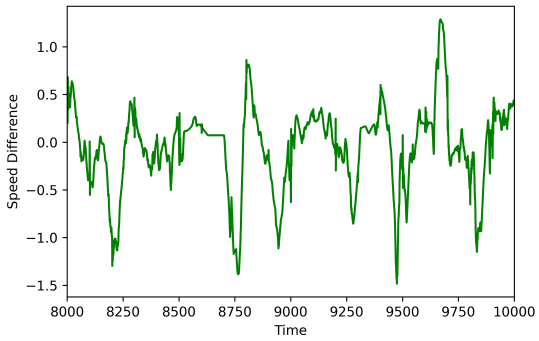
We also build the LSTM models to predict the speed of cyclists. The hyper-parameters are listed in Table 4.6. The optimal learning rates are searched by learning rate finder and



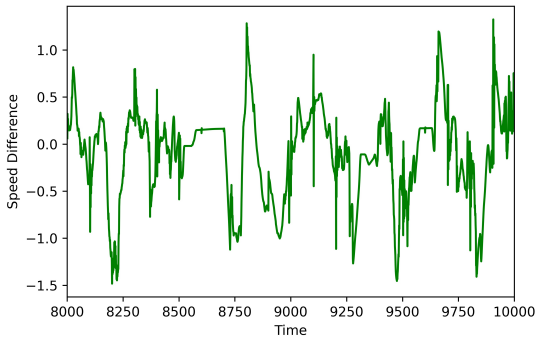
(a) Model 1



(b) Model 2



(c) Model 3



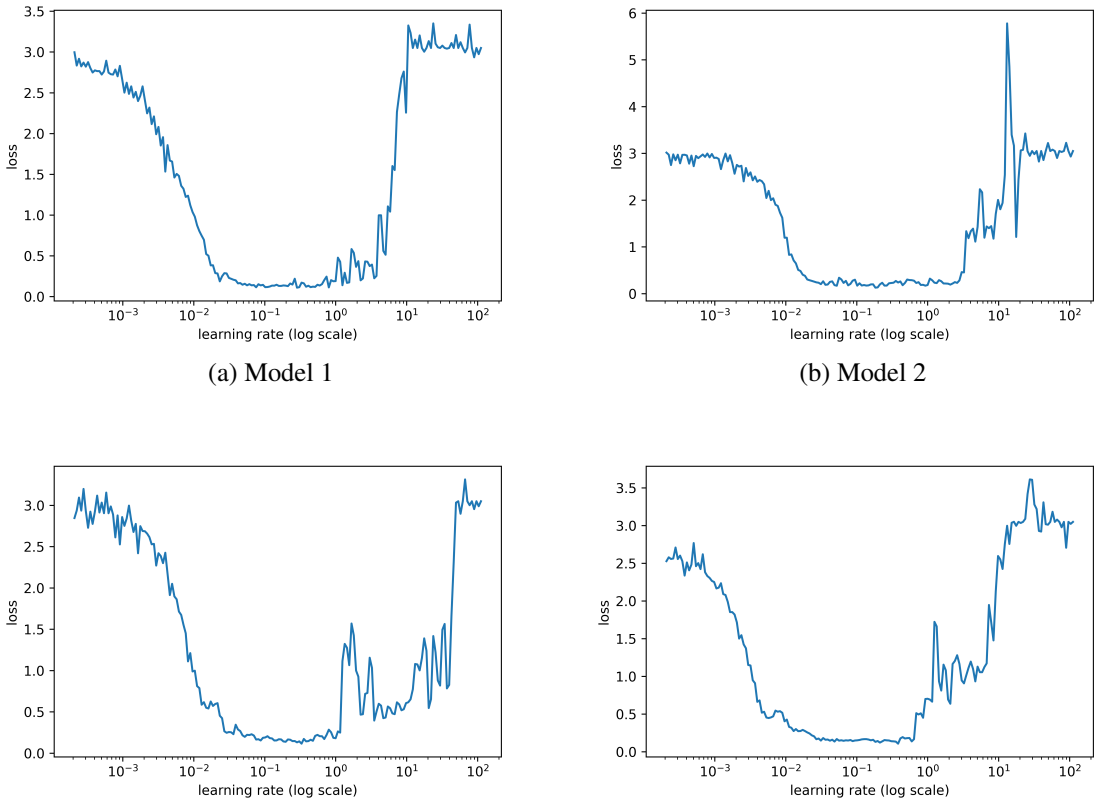
(d) Model 4

Figure 4.8: Speed Forecasting Error of RNN Models

the loss vs the learning rate curves are shown in Figure 4.6. The optimal learning rate is chosen to be 1×10^{-2}.

Model Number	Number of Dense Layer	Number of LSTM Layers
1	1	1
2	1	2
3	2	1
4	2	2

Table 4.6: Hyper-parameters of Simple RNN models



(a) Model 1

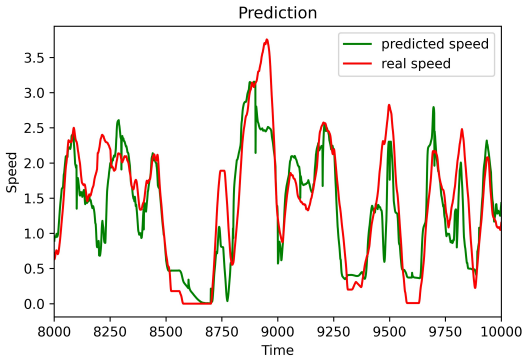
(b) Model 2

(c) Model 3

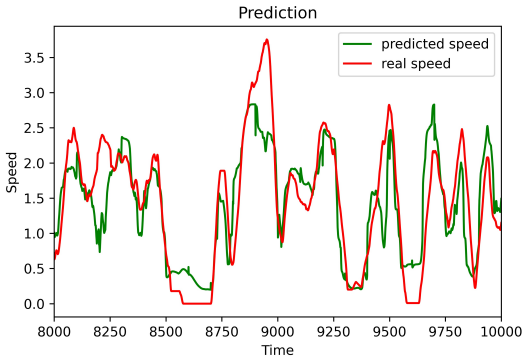
(d) Model 4

Figure 4.9: Optimal Learning Rate Search for LSTM Speed Forecasting Models

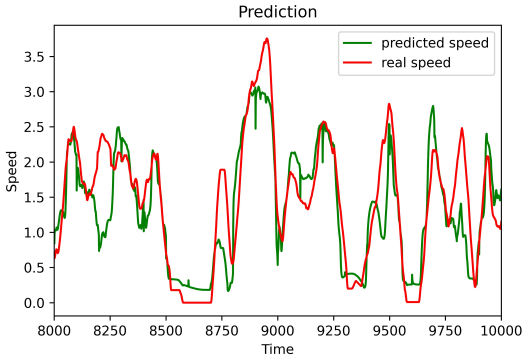
The results and errors are plotted in Figure 4.10 and Figure 4.11. MSEs are calculated and listed in Table 4.7. Among these four LSTM models, Model 4 shows the lowest MSE on both thr training set and test set.



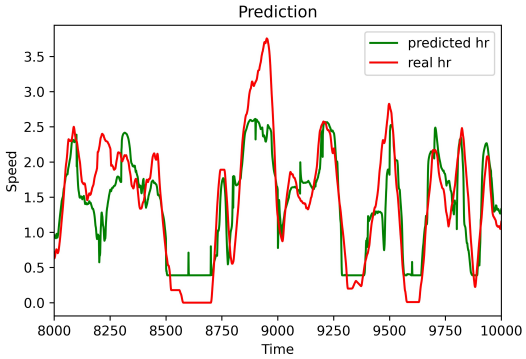
(a) Model 1



(b) Model 2



(c) Model 3

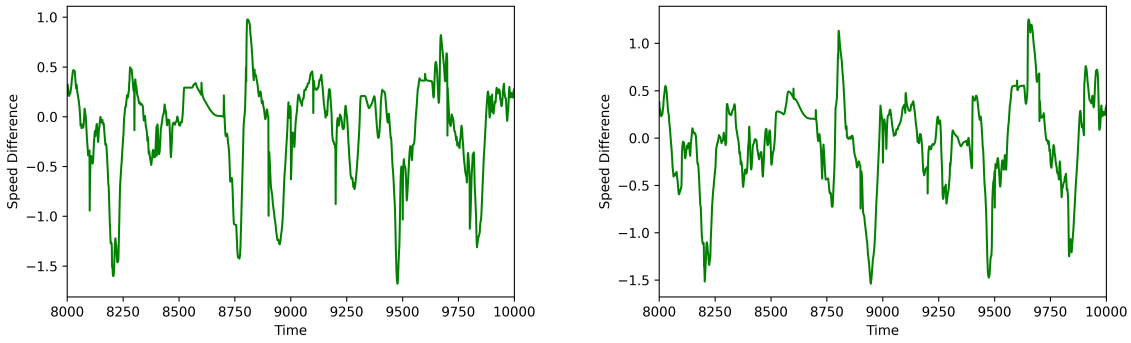


(d) Model 4

Figure 4.10: Forecasting Results for LSTM Speed Forecasting Models

LSTM Models	MSE on Training Set	MSE on Test Set
Model 1	0.134	0.248
Model 2	0.100	0.242
Model 3	0.098	0.235
Model 4	0.0789	0.207

Table 4.7: MSEs of LSTM Speed Models



(a) Model 1

(b) Model 2

(c) Model 3

(d) Model 4

Figure 4.11: Speed Forecasting Error of LSTM Models

4.6 Comparison Among Speed Forecasting Models

In this section, the model with the lowest mean squared error (MSE) in each type is selected and compared with each other. The speed forecasting for these four models are shown in Figure 4.12. The predicted speed among these four models are close which means all four models show relatively similar accuracy.

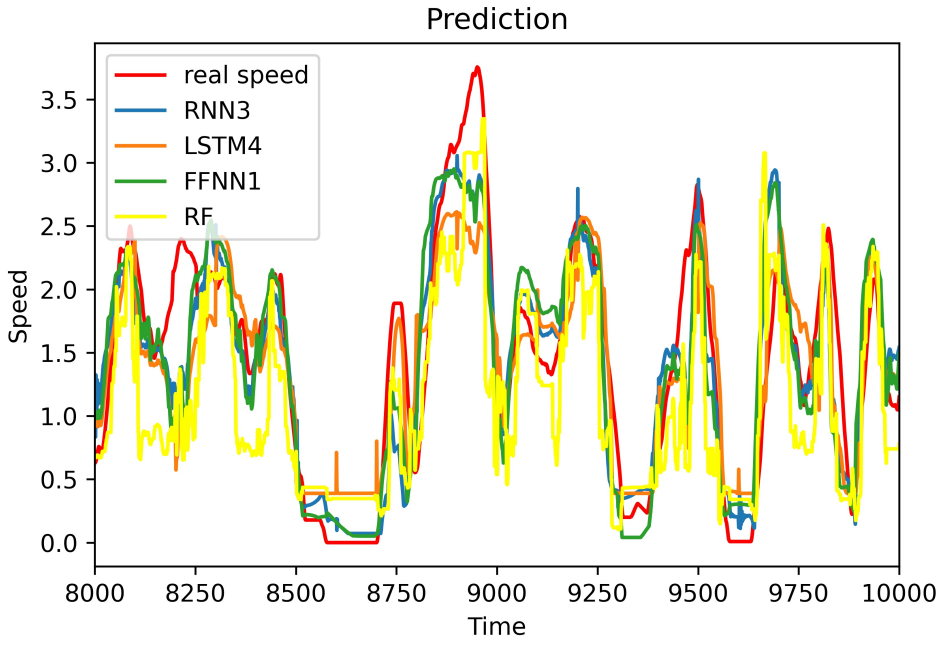


Figure 4.12: Comparison of Different Speed Forecasting Models

Table 4.8 shows the MSE of these four models. Among them, RF model shows severe over-fitting and therefore is not suitable to be applied in reality. The other three neural network models (FFNN, RNN and LSTM models) show similar accuracy on the provided data set.

Models	MSE on Training Set	MSE on Test Set
RF model	141.170.0010	0.397
FFNN model 1	0.1128	0.216
RNN model 3	0.119	0.196
LSTM model 4	0.079	0.207

Table 4.8: MSE of Speed Forecasting models

Chapter 5

Conclusion

In this thesis, personalized course-specific performance forecasting models are proposed. Random forest (RF), artificial neural network (ANN), recursive neural network (RNN), and long short term memory (LSTM) models are designed and implemented to forecast the heart rate and speed of athletes on a specific route. Among these four type of forecasting models, LSTM models and RNN models have the lower mean squared error in both heart rate and speed forecasting.

In the future, we are interested in forecasting cyclists' performance in more complex scenarios (i.e. performance on rocky courses) and utilizing other auxiliary data (i.e. images or videos recorded by camera on the bike).

BIBLIOGRAPHY

[1] Asker Jeukendrup and Adrie Van Diemen. Heart rate monitoring during training and competition in cyclists. *Journal of Sports Sciences*, 16(1):91–99, 1998.

[2] Mingyun Bian, Bo Peng, Wei Wang, and Jing Dong. An accurate LSTM based video heart rate estimation method. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11859 LNCS, pages 409–417. Springer International Publishing, 2019.

[3] Ankang Le, Thomas Jaitner, Frank Tobias, and Lothar Litz. A Dynamic Heart Rate Prediction Model for Training Optimization in Cycling. *The Engineering of Sport 7*, pages 425–433, 2009.

[4] E A Dawson, R Shave, K George, G Whyte, D Ball, D Gaze, and P Collinson. Cardiac drift during prolonged exercise with echocardiographic evidence of reduced diastolic function of the heart. *European Journal of Applied Physiology*, 94(3):305–309, 2005.

[5] Paul O Collinson, Frances G Boa, and David C Gaze. Measurement of Cardiac Troponins. *Annals of Clinical Biochemistry*, 38(5):423–449, sep 2001.

[6] ALEJANDRO Lucia, JESUS Hoyos, and JosÉ L Chicharro. Preferred pedalling cadence in professional cycling. *Medicine and science in sports and exercise*, 33(8):1361–1366, 2001.

[7] Farrokh Mohammadzadeh, Chang S. Nam, and Edgar Lobaton. Prediction of Physiological Response over Varying Forecast Lengths with a Wearable Health Monitoring Platform. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, 2018-July:437–440, 2018.

[8] Yuchi Ming and Jo Jun. Heart rate prediction based on physical activity using feedforward neural network. *Proceedings - 2008 International Conference on Convergence and Hybrid Information Technology, ICHIT 2008*, pages 344–350, 2008.

[9] Feng Xiao, Yi Min Chen, Ming Yuchi, Ming Yue Ding, and Jun Jo. Heart rate prediction model based on physical activities using evolutionary neural network. *Proceedings - 4th International Conference on Genetic and Evolutionary Computing, ICGEC 2010*, pages 198–201, 2010.

[10] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 1724–1734, 2014.

[11] Jianmo Ni, Larry Muhlstein, and Julian McAuley. Modeling heart rate and activity data for personalized fitness recommendation. *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, 2:1343–1353, 2019.

[12] Minmin Luo and Kaibin Wu. Heart rate prediction model based on neural network. *IOP Conference Series: Materials Science and Engineering*, 715:12060, 2020.

[13] Stefan Leijnen and Fjodor Veen. The Neural Network Zoo. *Proceedings*, 47:9, may 2020.

[14] S S Chowdhury, M S Hasan, and R Sharmin. Robust Heart Rate Estimation from PPG Signals with Intense Motion Artifacts using Cascade of Adaptive Filter and Recurrent Neural Network. In *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, pages 1952–1957, 2019.

[15] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[16] SPSS Tutorials. Pearson correlation. *Retrieved on February*, 4, 2014.

[17] John H McDonald. *Handbook of biological statistics*, volume 2. sparky house publishing Baltimore, MD, 2009.

[18] Anthony J Bishara and James B Hittner. Confidence intervals for correlations when data are not normal. *Behavior Research Methods*, 49(1):294–309, 2017.

[19] Jerome L Myers, Arnold Well, and Robert Frederick Lorch. *Research design and statistical analysis*. Routledge, 2010.

[20] Leo Breiman. Bagging Predictors. *Machine Learning*, 24(2):123–140, 1996.

[21] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[22] Simon S Cross, Robert F Harrison, and R Lee Kennedy. Introduction to neural networks. *The Lancet*, 346(8982):1075–1079, 1995.

[23] Steven W Smith. The scientist and engineer's guide to digital signal processing. pages 458–465, 1997.

[24] Yung-Yao Chen, Yu-Hsiu Lin, Chia-Ching Kung, Ming-Han Chung, and I Yen. Design and implementation of cloud analytics-assisted smart power meters considering advanced artificial intelligence as edge analytics in demand-side management for smart homes. *Sensors*, 19(9):2047, 2019.

[25] Aston Zhang, Zachary C Lipton, Mu Li, and Alexander J Smola. Dive into deep learning. *Unpublished Draft. Retrieved*, 19:2019, 2019.

[26] Li Jun and Tom Duckett. *Some Practical Aspects on Incremental Training of RBF Network for Robot Behavior Learning*. jul 2008.

[27] L N Smith. Cyclical Learning Rates for Training Neural Networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472, 2017.

[28] Pavel Surmenok and Jonathan Mackenzie. keras_lr_finder. *GitHub repository*, 2017.

[29] Lei Tai and Ming Liu. *Deep-learning in Mobile Robotics - from Perception to Control Systems: A Survey on Why and Why not*. dec 2016.

[30] Jeffrey L Elman. Finding Structure in Time. *Cognitive Science*, 14(2):179–211, mar 1990.

[31] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, nov 1997.

[32] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with LSTM. 1999.

[33] Jiayu Qiu, Bin Wang, and Changjun Zhou. Forecasting stock prices with long-short term memory neural network based on attention mechanism. *PloS one*, 15(1):e0227222, 2020.