School of Electrical Engineering, Computing and Mathematical Sciences

Statistical Analysis of Delay in Time Series

Manlika Ratchagit

0000-0001-8600-5387

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University

January 2023

Declaration

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgment has been made.

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Signature:

Date: 2-January-2023

Abstract

This dissertation focuses on delay in time series data. The first delay involves the m-delay autoregressive model. This approach considers only the first and the last previous observation of the traditional autoregressive model. The least square method is utilized to estimate two unknown parameters of our proposed model. The numerical results obtained from the simulation and the case study confirm that the m-delay autoregressive model is effective and reduces computation time with a larger delay.

Next, the delay is added to the stochastic differential equation for matching the volatility between real-world financial data and Monte Carlo simulations. Autoregressive coefficient and the differential evolution algorithms are applied to seek the parameters of the stochastic delay differential equation. It indicates that the estimated volatility obtained from Monte Carlo simulations for four stocks fits the actual volatility. In addition, the study becomes more accurate when the delay is increased, and a higher sampling frequency leads to higher estimation accuracy.

Lastly, we propose a new technique to improve prediction accuracy for stock price prediction. The well-known deep learning models, namely the multilayer perceptron, the convolutional neural network and long short-term memory network, are applied to historical stock prices. The novel technique is called a two-delay combination model. Each popular deep learning method combines two delays, half-day and one-day. One challenging task of the linear combination forecast is weight identification. We utilize the differential evolu-

tion algorithm for obtaining optimal weights of the linear combination forecast procedures. The results show that the patterns of the predicted closing price obtained from our proposed method are similar to the actual stock prices. As a result, the combination forecasts perform well compared to the individual deep learning procedure. One benefit of our proposed technique is that applying several deep learning methods in the combination forecast method is unnecessary. Based on the general idea of the combination forecast method, it requires at least two different approaches to build the forecast combination. Under our proposed technique, using only one method with two different delays can reduce the evaluation metrics. Consequently, the two-delay combination model is a potential method with satisfactory prediction performance.

Acknowledgements

First of all, I would like to express my deepest gratitude to my principal supervisor Dr Honglei Xu for his consistent support, assistance and advice during my PhD studies. I would like to express my sincere appreciation to Prof. Yong Wu and Prof. Kok Lay Teo for serving on my committee. They provide valuable comments and suggestions on every stage of PhD career over the years. I am also grateful to Prof. Nikolai Dokuchaev, Prof. Ba Tuong Vo and Assoc. Prof. Nicola Armstrong for providing guidance and support. Furthermore, I would like to extend my sincere gratitude to Assoc.Prof. Benchawan Wiwatanapataphee and Dr. Darfiana Nur for their comments on my research.

I sincerely acknowledge the organizations that provide financial support for my PhD studies, including a Royal Thai Government Scholarship and the Maejo University. I would like to express my most profound sincere to my mother for her support, endless love and understanding. My husband, Grieng-grai, for always loving, supporting, and pushing me to be better. Thanks to my wonderful kid Tonnam for your powerful hug, patience, and belief in me. Furthermore, I would like to thank the Minister Counsellor (Education) and the Office of Educational Affairs, Royal Thai Embassy, Australia staff for their valuable advice and support.

Lastly, I would like to thank all my friends at Curtin University, especially Tony and Lely, who always stayed with me, listened to me and supported me all the time during my PhD life.

My sincerest thanks go to all the above from the bottom of my heart.

Contents

List of Figures

List of Tables

List of publication

This thesis is based upon several works that have been published in journal and conferences proceedings during of the author's PhD study. They are listed in chronological order as

1. **Ratchagit M**, Wiwatanapataphee B, Dokuchaev N. The m-delay Autoregressive Model with Application. *CMES-Computer Modeling in Engineering and Sciences*. 2020;122(2):487-504. doi:10.32604/cmes.2020.08865.

2. **Ratchagit M**, Wiwatanapataphee B, Nur D. On Parameter Estimation of Stochastic Delay Difference Equation using the Two *m*-delay Autoregressive Coefficients. In2020 3^{rd} International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) 2020 Dec 10 (pp. 310-314). IEEE. doi: 10.1109/ISRITI51436.2020.9315414.

3. **Ratchagit M**, Xu H. Parameter Identification of Stochastic Delay Differential Equations Using Differential Evolution. In 2022 37^{th} International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC 2022) 2022 Jul 5-8 (pp. 804-806). IEEE.
doi: 10.1109/ITC-CSCC55581.2022.9894864

4. **Ratchagit M**, Xu H. A Two-Delay Combination Model for Stock Price Prediction. *Mathematics*. 2022;10(19):3447. doi:10.3390/math10193447.

The copyright information to reuse the published works has been provided in Appendix B.

Chapter 1

Introduction

Chapter 1 presents the background, objectives, contributions, and thesis framework. First, the background of time series and several applications of time delay are presented in Section 1.1. Then, Section 1.2 and Section 1.3 introduce the thesis objectives and contributions, respectively. Finally, the thesis structure is represented in Section 1.4.

1.1 Background

Time is an essential part of the information that most people perceive from the world. A collection of observations in a time order are called time series. Hence, time series are everywhere in daily life, and soon everything will be a time series [5]. Time series data and analysis have been applied in various areas, including weather, finance, healthcare, and the environment. Weather prediction is crucial because it can help to minimize risk and save a life. People can plan their activities, including tourism, based on expected weather conditions. For example, people can plan to get dressed differently when the weather is hot, cold, windy or rainy. It is also important for aircraft, boats, and pedestrian transportation, including walking and bicycling. Weather forecast is one of the popular topics for the researchers in many fields such as temper-

ature forecasting [6–8], rainfall forecasting [9, 10], tourism forecasting [11, 12], air traffic [13, 14], marine [15, 16] and forestry [17, 18]. Some sectors require more accurate weather forecasts. Financial forecasting is important because it helps to plan and manage business finance. Stock price prediction is a popular topic for forecasting future price movement. Many scholars proposed various techniques to predict stock price time series [19–21]. Prediction of health situations [22, 23], healthcare need forecast [24, 25] ,and disease prediction [26, 27] are in the area of healthcare forecasting. Accurate forecasting of the healthcare system is beneficial for satisfying healthcare needs, and demand for medical service [28].

Time series analysis can be classified into two groups, namely univariate and multivariate time series analysis. The univariate time series refers to only one variable varying over equal time-space. Future observations of a time series are computed using its past and present values [29]. For instance, Lu et al. [30] developed nine deep learning models and predict of the closing price of the Shanghai Composite Index. The data are obtained from the wind database with 7,083 observations. The first 6,083 prices are used as the training set, and the last 1,000 observations are used as the test set. Xie et al. [12] presented a hybrid complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN) and data characteristic analysis (DCA) to predict tourism demands. The data used in this research are the number of tourist arrivals to Hong Kong between January 2001 and March 2019, totalling 219 observations. The four decomposition-ensemble models are applied to compare with the proposed (CEEMDAN-DCA) method. The data from the above examples depend only on time and past observations. Hence, there are classified in the univariate time series. The multivariate time series involves analysis of two or more variables. Forecasting is based on a sample of time series observations taking into account the effect of other variables [31]. For example, five variables, including age, gender, medical comorbidities, medication records, and

laboratory examination, are used to forecast severe COVID-19 disease in Hong Kong [32]. Nguyen et al. [33] utilized the multivariable time series, including the density, the ratio of water-to-cement, and the ratio of sand-to-cement, to predict the compressive strength of foamed concrete. Bouktif et al. [34] considered five variables, including open, high, low, close and volume (OHLCV), and those price data from Yahoo finance can be used for movement prediction.

Delay is one of significant indicators in time series. Time delay plays an essential role in many fields, especially transportation and medicine. The delay is one of the most remembered performance indicators of any transportation, particularly flight delay prediction. Flight delays have negative impacts, mainly on airline business (increasing the operational costs to airlines.), passengers (increasing the cost to customers), airlines ranking, and airports (having effect on airports management). Therefore, flight delay models are proposed for delay prediction. For example, Gui et al. [35] established random forest-based and the long short-term memory network (LSTM)-based methods for predictably individual flight delays. They combine many factors that potentially influence flight delays, such as flight information, weather condition, traffic flow, flight schedule, and airport information. Yu et al. [36] provided a novel deep belief network to predict delay using high dimension data from Beijing International Airport. Weather, seasonal effects, delay propagation, air traffic control, air route network, and the airport's crowdedness degree are the relevant factors for flight delay prediction. Qu et al. [37] present two flight delay prediction models using the dual-channel convolutional neural network (DCNN) and the squeeze and excitation-densely connected convolutional network (SE-DenseNet) to improve the prediction accuracy. The multivariate variable used in this work are the flight information (e.g., ground traffic, flight number, takeoff/landing flight path and so on) and meteorological data (e.g., wind direction, wind speed, visibility, temperature, air pressure, and so on).

There are also numerous studies on the time delay in diagnosis and initiation of treatment in medical science. Some patients suffer from more severity or mortality because of delayed diagnosis. Bello et al. [38] studied delay in diagnosing or treating pulmonary Tuberculosis (TB). The six critical delays were investigated, including the patient's delay, the diagnostic delay, the treatment delay, the doctor's delay, the health system delay and total delay. Iqbal et al. [39] analyzed various factors contributing to the pediatric population's delayed diagnosis of Congenital Heart Disease (CHD). The possible factors for delayed diagnosis include the first time delayed consultation, delayed or missed diagnosis by the doctor, delayed referral, social factors and financial factors. Hoyer et al. [40] studied risk factors for diagnostic delay in idiopathic pulmonary fibrosis. The risk factors for delays include patient, GP, hospital and waiting.

In this thesis, we investigate delays that are applied in several areas. Chapter 3, we start with the classical autoregressive model. This model relies on its past observations. The m-delay autoregressive (MAR) model is proposed based on the idea of ignoring some variables. Our proposed procedure utilizes only the first and the last observations. The MAR coefficients are presented to estimate two parameters of our proposed model. Chapter 4 studied the stochastic delay differential equation (SDDE) to match the estimated volatility obtained from the Monte Carlo simulations and the real-world volatility. The autoregressive coefficients (ARC) and the differential evolution (DE) algorithms are applied to estimate unknown parameters of the SDDE model. Stock price prediction using the linear combination model is the main task in Chapter 5. A two-delay combination technique based on deep learning (DL) methods is proposed to improve the performance of the individual DL models. Our proposed model and the individual DL models are compared for both univariate and multivariate time series.

1.2 Objectives

The research objective of this thesis is to study delays in time series. This work has the following objectives:

- To formulate the m-delay autoregressive model and develop the least square method for parameter estimation.
- To match the volatility obtained from Monte Carlo simulations using the stochastic delay differential equation and the real-world historical stock price.
- To increase the prediction accuracy of the individual deep learning model using a two-delay combination method.

1.3 Main Contributions of This Thesis

To complete these objectives, the contributions of this thesis are presented in Chapter 3, Chapter 4 and Chapter 5, respectively. The contributions include the following aspects.

- A novel autoregressive model, namely the m delay autoregressive model, is proposed in Chapter 3. This model is a particular case of the classical autoregressive model that requires only two observations. We then develop the least square method to estimate two unknown parameters of our proposed model. The novelty of our model is that we consider a model with unknown and variable m that can be large and that the inference is still feasible. We suggested a computational method allowing statistical inference for this large unknown m and investigated its computational feasibility.
- The autoregressive coefficients and differential evolution algorithms are applied for model identification of the stochastic delay differential equa-

tion to match the volatility between Monte Carlo simulations and case studies in Chapter 4. To the best of our knowledge, we are the first to estimate a model parameter, namely the drift term (λ) and the volatility (σ^2) of the stochastic delay differential equation.

- The two-delay combination technique is proposed in Chapter 5. To the best of our knowledge, there is no attempt to combine two-delay for stock price prediction. Moreover, the differential evolution algorithm is applied to estimate the weight of the linear combination model. Furthermore, three well-known deep learning models, including MLP, CNN, and LSTM, are employed for historical stock prices. Finally, the two-delay combination method is utilised to improve the popular deep learning models' performance and confirm our proposed model's effectiveness.

1.4 Outline of The Thesis

This thesis consists of six Chapters. Chapter 1 gives the background, the objectives and the contribution of the thesis, and the thesis structure. Chapter 2 presents the literature review in time series analysis, including the classical parametric (Box-Jenkins approach) and nonparametric (deep learning) methods. The basics of the Box-Jenkins approach in time series analysis and deep learning techniques are demonstrated in this chapter. Next, the stochastic differential equation and the volatility are reviewed. Finally, the deep learning forecasting with delay is detailed. Chapter 3 represents a classical autoregressive model with skip delay. This model is called the m-delay autoregressive model. The m-delay AR coefficients are proposed to estimate two unknown parameters of the proposed model. The parameter estimation using the modified least square method and the stationary condition is revealed in this part. Next, Chapter 4 focuses on the estimated volatility of the stochastic delay dif-

ferential equation (SDDE). The parameter estimation techniques used in this chapter are the autoregressive coefficients (ARC) and differential evolution (DE) algorithms. Finally, we match the estimated volatility obtained from the Monte Carlo simulations and the real-world volatility. Chapter 5 focuses on the stock price prediction based on the linear combination technique. The two-delay combination method is proposed for improving the prediction accuracy of the individual deep learning methods. We also apply the DE algorithm for the weight identification of the linear combination method. This chapter is divided into two parts, namely univariate and multivariate time series. The performance of the individual DL method and our proposed technique is compared for both cases. A summary of the thesis is given in the first part of Chapter 6. Several possible future studies on the current topic are then given in the second part of Chapter 6. The details of thesis framework is presented in Figure 1.1.

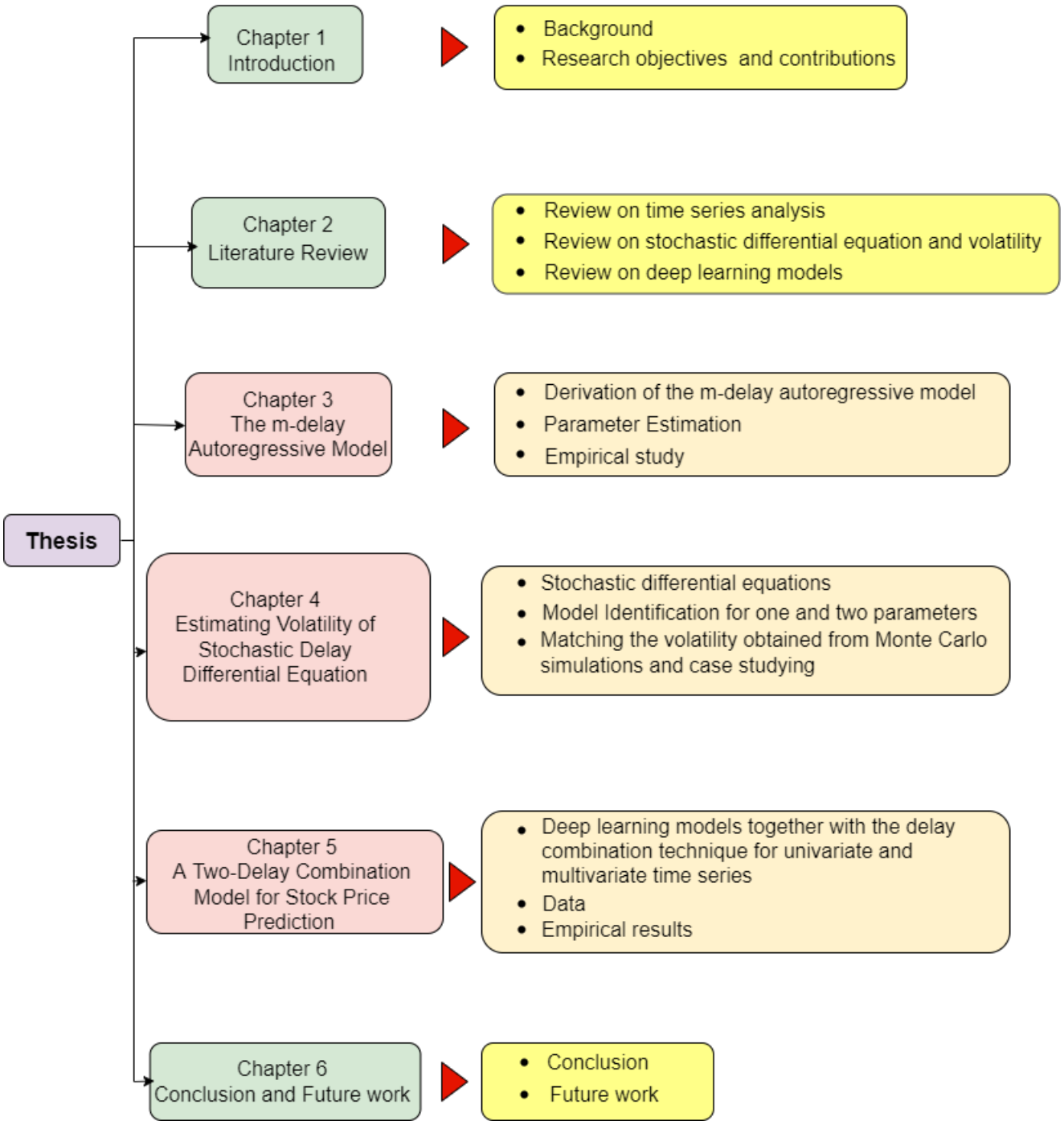


Figure 1.1: Thesis framework.

Chapter 2

Literature Review

This chapter details the literature review utilised in Chapters 3, 4, and 5. The delays in this thesis are used in three areas: the traditional time series model, the stochastic differential equation (SDE), and the deep learning (DL) methods for stock price prediction. First, Section 2.1 describes the time series analysis. Next, the stochastic differential equation is presented in Section 2.2. Then, the deep learning models in time series forecasting are displayed in Section 2.3. Deep learning for time series with delay is represents in Section 2.4. Finally, we provide a brief content of this chapter in Section 2.5.

2.1 Time Series Analysis

Time series analysis has been widely applied in many branches, such as agriculture, environment, etc. For example, the rainfall forecast is a popular topic in time series analysis for irrigation management proposed for agriculture applications. A time series model has been used to predict the amount of rainfall for agriculture target planning. This would probably help the farmer to deal with their resources and gain more agricultural products [41]. Furthermore, the weather forecast is essential for the environment as the temperature is crucial for government planning and management in many aspects such as

tourism, vegetation, and public health [42].

Various approaches based on time series methods have been proposed to fit the data set. These approaches include the Box-Jenkins method, neural networks, hybrid technique and fuzzy time series method. Box-Jenkins method is one of the famous methods consisting of the autoregressive (AR) model, moving average (MA) model, autoregressive moving average (ARMA) model and autoregressive integrated moving average (ARIMA) model [43]. This method has been applied to a wide range of applications including economic especially a stock process sequence [44], engineer [45], industry [46], medicine [47] ,and science [48]. The artificial neural networks (ANN) approach is the main tool for nonlinear data as it is a brain-inspired system [49]. The hybrid technique has been established since the individual model may not be adequate to analyze all the actual observations [50]. This method combines at least two individual forecast methods to improve forecasting accuracy [51]. ANN approach integrated with either AR or ARIMA method is called AR-ANN model [52] or ARIMA-ANN model [53]. Fuzzy time series is a forecasting method using the fuzzy principle as the basis. It is suitable for numerical and linguistic data [54]. Apart from the development of forecasting methods, various estimating model-parameter approaches used in the time series model have been proposed, such as the maximum likelihood method, the method of moment and the Yule-Walker procedure. For the maximum likelihood method, studies aim to find the parameter values giving the distribution that maximizes the probability of observing the data. The method of the moment is a simple procedure. By equating the sample moments to the corresponding population moments, the estimating model parameter is solved [55]. The Yule-Walker method (or autocorrelation) fits an AR model to the window input data by maximizing the error in the least squares sense [56].

Time series data are commonly classified into two groups; stationary and non-stationary. For stationary data, its mean and variance are consistent. How-

ever, its sequence does not reveal an upward (or downward) trend and seasonal pattern. In contrast to the stationary data, the mean and variance of non-stationary data vary over time. Three common ways to check the stationary of the data are time series plot, unit root test [57], and roots of the characteristic equation [58]. For the unit root test, we apply the Augmented Dickey-Fuller (ADF) test to check whether the data are stationary or not stationary. The presence of a unit root means the time series is non-stationary. Hence, the null hypothesis (H_0): the data containing unit root (the data are non-stationary) versus the alternative hypothesis (H_1): the data are stationary. For roots of the characteristic equation, the roots of the characteristic equation may be real and/or complex numbers. The data are stable if all real roots are greater than one or the modulus of each complex root is greater than one. The data is considered non-stationary if at least one root falls between minus and plus one or falls inside the unit circle [43]. As the AR model requires stationary data to predict future data, the single/double differencing method and mathematical transformation are two standard procedures to convert non-stationary data to stationary data [59]. The differencing technique is used to remove the trend component of the data. At the same time, mathematical transformations, including the square/cube root transformation and log transformation, are applied when the variance of data is unstable [60].

AR model has been widely applied in several areas. Many researchers have developed the AR model for time series prediction. For example, Pena-Sanchez et al. [61] readdressed the AR model for short-term forecasting of sea surface elevation. In their work, the authors compared four techniques, namely the AR model using a linear least square (AR_{LLS}) method, the AR model via long-range predictive identification (AR_{LRPI}) method, the direct multistep using the spectrum (DMS_{Sp}) method and the direct multistep via the linear least square (DMS_{LLS}) method to predict wave elevation in Belmullet, Ireland. The results obtained from simulation and observed data indicated

that AR models are suitable for short-term prediction of sea surface elevation comparable to state-of-the-art procedures. Kandula and Shaman [62] proposed three AR-base models to forecast the number of outpatient influenza-like illnesses (ILI) in the USA. The three AR-base models include an autoregressive integrated moving average (ARIMA) model, an ARIMA model with seasonal and trend decomposition (ARIMA-STL), and a feed-forward autoregressive artificial neural network with a single hidden layer (AR-NN). The results show that the hybrid AR-NN model outperforms the other two procedures. Maleki et al. [63] improved autoregressive models based on two-piece scale mixture normal distributions, called TPSMNAR model, to predict the number of confirmed and recovered cases of COVID-19 in the world. The results confirm that the proposed model performs well in terms of the mean absolute percentage error (MAPE) to estimate the number of confirmed and recovered cases of COVID-19 for ten days during 21 - 30 April 2020. The summary of reviewed papers in this section is shown in Table 2.1.

2.2 Stochastic Differential Equation (SDE)

A stochastic process is a process that involves random variables changing over time [64]. The stochastic processes can be divided into discrete-time and continuous-time to model the asset price. The price can be discrete or continuous for both types of stochastic processes. However, a discrete price can only assume a countable number of possible values, while a continuous price assumes any positive actual number. A stochastic differential equation (SDE) is a differential equation that contains one or more of the terms is a stochastic process. This model is essential for modelling random phenomena in the financial field.

Table 2.1: The literature summary in Section 2.1 is separated by experiment types, data types, and the delay problem.

Reference	Method	Experiment		Data		Delay	
		Simulation	Case study	Univariate	Multivariate	Yes	No
Ip et al. [44]	MAR	✓	✓		✓	✓	
Amo-Salas et al. [45]	OED-AR	✓	✓	✓			✓
Acedański [46]	NF, SAR, LI		✓	✓			✓
Sharafi et al. [47]	SRIMA		✓		✓	✓	
Tsitsika et al. [48]	ARIMA		✓	✓	✓		✓
Pan et al. [51]	AR-ELM		✓	✓			✓
Qi, M and Zhang, G [52]	ARIMA-ANN		✓	✓			✓
Pena-Sanchez et al. [61]	AR_{LLS}, AR_{LRPI}, DMS_{SP}, DMS_{LLS}	✓	✓	✓			✓
Kandula and Shaman [62]	ARIMA-STL, ARIMA,AR-NN		✓		✓		✓
Maleki et al. [63]	TP-SMN-AR		✓	✓			✓

The general form of the SDE consists of deterministic or average drift and diffusion terms.

$$dY(t) = a(Y(t), t)dt + b(Y(t), t)dw(t), \tag{2.1}$$

where $a(Y(t), t)$ and $b(Y(t), t)$ are drift and diffusion term, respectively, $Y(t)$ denotes the return of stock prices, $t \in [0, T]$, and $T > 0$. In financial statistics, $b(Y(t), t)$ is termed the "volatility" [65]. We estimate the solution through discretization of SDE.

Stochastic processes are widely used in many areas, especially finance, where asset prices evolve and form a stochastic process. Discrete-time and continuous-time are two main kinds of stochastic methods for modelling the price of an asset. The discrete-time stochastic process can be defined as the price changes instantaneously. For instance, the daily closing price of IBEX 35 on the Spanish Continuous Market is classified as a discrete-time stochastic process due

to the price change only at the closing of a trading day. On the other hand, in the continuous-time process, the price changes continuously even though the price is only observed at discrete time points.

A stochastic differential equation (SDE) is an equation in which one or more terms is a stochastic process. The SDE has been used to model an asset price and its volatility in finance [66, 67]. Investors use historical stock prices to predict the market movement and make investment decisions. For the mathematical model in finance, the Black-Scholes model was introduced to describe derivatives of the stock price with geometric Brownian motion (GBM) behaviour [68]. However, it was reported that GBM could not capture forward or backward behaviour [69–71]. The classical SDE has introduced a time delay to overcome the problem of capturing market dynamics and financial derivatives. Many researchers have established stochastic delay differential equations (SDDEs) and applied them to finance studies [72–75].

Tambue et al. [72] compared numerical solutions obtained from the SDDE and classical Merton model with the actual corporate data. They stated that the SDDE with constant delay provides stable and accurate behaviour for pricing equity. Eissa and Tian [73] proposed SDDE with a variable delay to model the price of a firm and European option. They reported that their proposed model is more flexible to fit accurate market data. Lee et al. [74] modified SDDE to capture the feedback effects of the option price. They confirmed that the modified SDDE model was not a risk-neutral one for generality and captured the main idea of the delay effect for option pricing. Finally, Ernst and Soleymani [75] provided a numerical algorithm for solving the SDDE. Their algorithm employed either the Legendre collocation method or the Chebyshev-type method. The numerical results from both ways were compared and confirmed that the proposed method was computationally faster and more accurate than the Chebyshev-type method.

Volatility is a significant task as it is the most important in pricing deriva-

tive securities. Higher volatility can cause essential variations of return. Hence it can be a higher risk in the financial market. The important thing is that financial volatility is not directly observable, like the return process observed from the price process. It requires some techniques to estimate the volatility. The three main classes of volatility estimation are historical volatility, implied volatility and stochastic volatility. Historical volatility (HV) is a statistical measure of the dispersion of returns based on historical return movements. The most usual method is the standard deviation of the log of price returns. This thesis focuses only the historical volatility because this measure is simple to use and has been investigated in the literature [76].

For volatility estimation, Masset [77] pointed out that volatility computed at different time scales has different information content. In general, volatility forecast accuracy improves as data sampling frequency increases for some financial time series [78]. However, when more frequent samples are used can cause volatility decreases for other financial time series. Concurrent with the move toward the use of higher frequency data. Zhang et al. [79] reported that increasing sampling frequency leads to higher volatility. Luong and Dokuchaev [80] estimated the volatility of the most traded indices and US stocks using a stochastic delay differential equation at different sample high-frequency data such as 15 minutes, 5 minutes and 1-hour intervals. The author confirmed that an additional delay term in the SDE better matches the volatility obtained from the Monte Carlo simulations and the historical data. The SDDE topics are summarized in Table 2.2.

2.3 Deep Learning Models in Time Series Forecasting

Nowadays, deep learning techniques have become one of the most researched algorithms due to their ability to deal with non-linear and multi-

Table 2.2: The summary of literature in SDDE is separated by experiment types, data types, and analysis problems.

Reference	Experiment		Data type				Analysis		
	Numerical	Case study	Equity	Option price	Return	Currency rate	Forecasting	Comparison	Derivative
Tambue et al. [72]		✓	✓				✓	✓	
Eissa and Tian [73]	✓			✓					✓
Lee et al. [74]	✓			✓					✓
Ernst and Soleymani [75]	✓			✓					✓
Lahmiri [76]		✓				✓	✓	✓	
Masset [77]		✓			✓			✓	
Zhang et al. [79]	✓				✓				✓
Luong and Dokuchaev [80]	✓	✓			✓			✓	✓
Andersen et al. [81]		✓			✓		✓	✓	

dimensional problems. These methods are a part of machine learning (ML) based on artificial intelligence (AI). The human nervous system inspires the ideas of deep learning. Deep learning approaches learn optimal features directly from large amounts of data, without any visible [82]. These techniques outperform the machine learning methods in terms of flexibility to solve several problems, including complex functions. Time series forecasting using deep learning methods has been viral among researchers over the past decade. Several published deep learning predictions with application to various areas have been presented in the last few years. The Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) is presented to report a wide array of systematic reviews and meta-analyses on deep learning methods in time series analysis specifically finance [83]. The criteria to produce the articles search are as follows:

- The search query used is: (deep-learning) AND (financ* OR stock) AND (prediction OR forecasting).

- The databases used in this study are Scopus and Web of Science.
- The search results are limited between 2013 and 2022.
- The search results are considered only the published work in the English language.
- All the documents searched are conference papers, articles, conference reviews, book chapters, reviews and books.

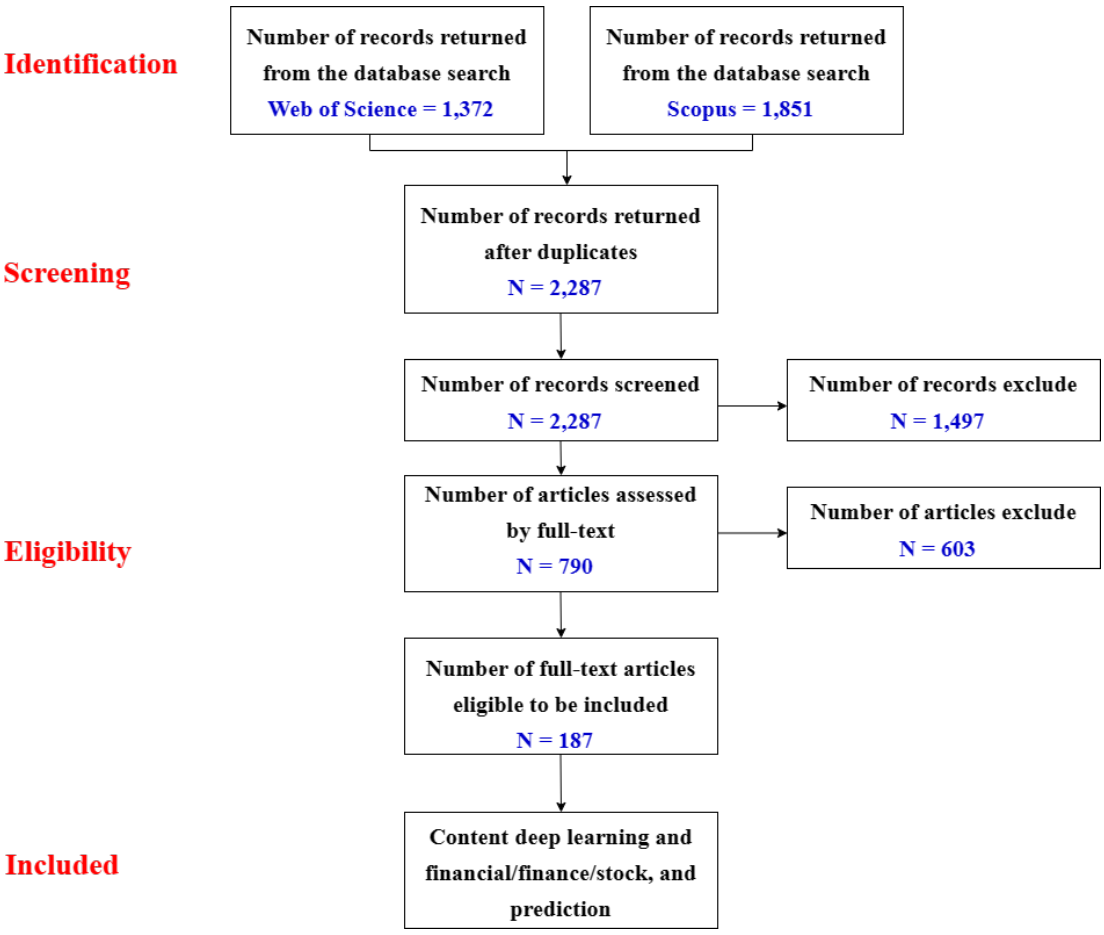


Figure 2.1: The PRISMA diagram.

Figure 2.1 displays the PRISMA diagram. First, we collected 1,372 articles from the Web of Science database and 1,851 articles from the Scopus database. After that, we removed duplicate articles from both databases. The total 2,287

remaining articles were documented. Then, only full-text articles are recorded, with 790 articles available for eligibility assessment. In the last step, the 187 full-text articles were included covering deep learning, finance, and prediction.

Deep Learning refers to the arrangement of multiple hidden layers of neurons in between the input layer and output layer. This technique can be divided into two general groups, supervised and unsupervised learning. The fundamental difference between supervised and unsupervised is that the algorithms learn from the input data to predict the output, and all observations are labelled. In contrast, the data used for unsupervised learning are unlabeled, and the algorithms learn the inherent structure from the input data. Unsupervised learning focuses on finding relationships in a data structure without having a measured outcome, while supervised learning is used for forecasting/-classification a specific result of interest [84]. Popular programming languages used for deep learning algorithms such as MATLAB, R, and Python [82]. Figure 2.2 represents deep learning architectures in time series analysis.

2.3.1 Unsupervised Techniques

This technique makes it possible to implement the learning process with unlabelled data. These algorithms utilize only the input data to mine for rules and detect patterns in the data. Staked autoencoder (SAE) is one of the famous models for unsupervised techniques because it improves feature extraction accuracy and reduces dimensional feature space [85]. Another helpful model is the deep belief network (DBN) [86]. This approach is an effective model when applied to multi-source inputs.

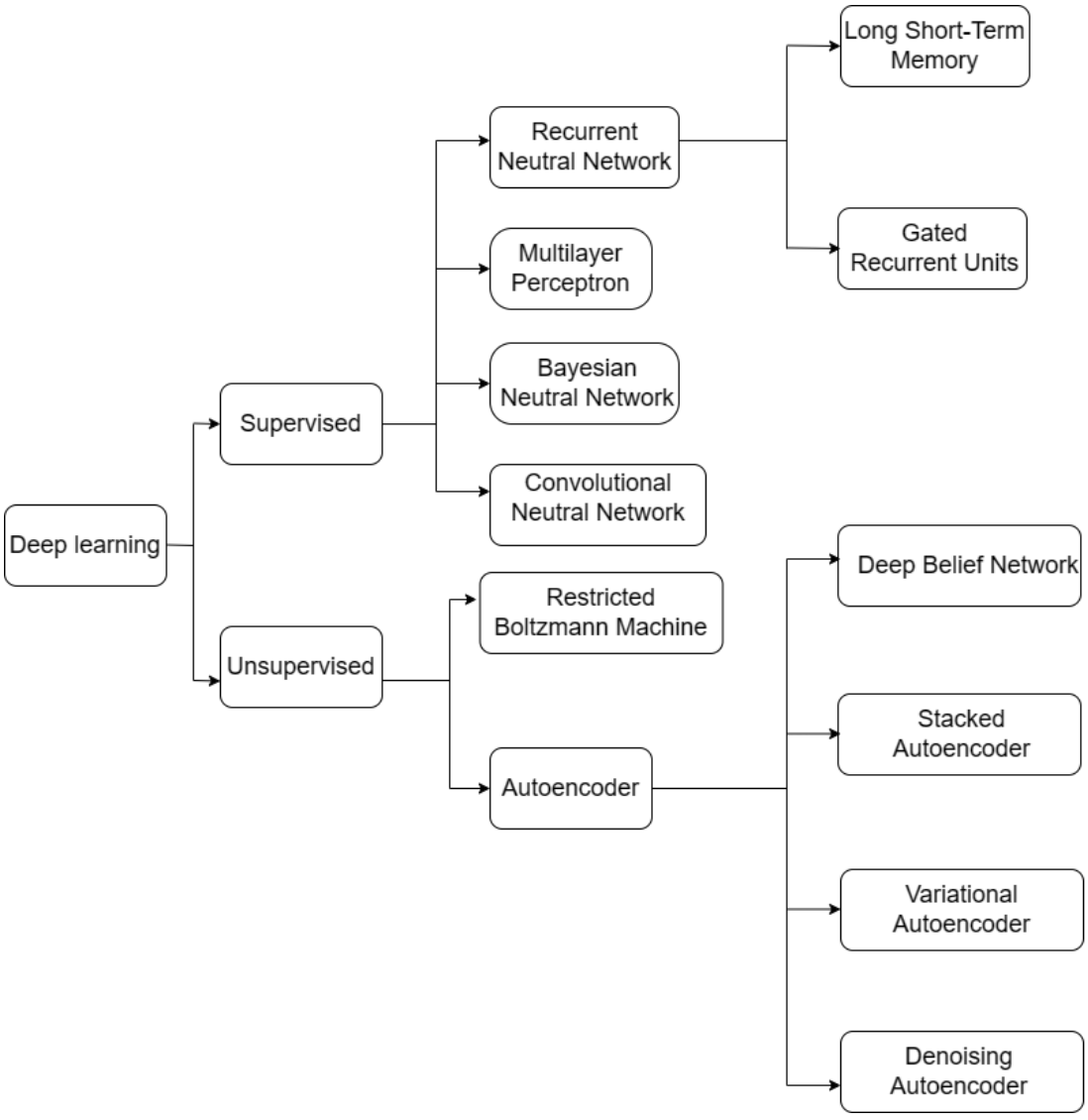


Figure 2.2: The most common deep learning architectures for analyzing time series data.

However, the major weakness of unsupervised learning is less accurate data sorting because this technique learns from untagged data [87].

2.3.2 Supervised Technique

Supervised techniques, including the artificial neural network (ANN), the multilayer perceptron (MLP), the convolutional neural network (CNN), recurrent neural network (RNN), and the long short-term memory network (LSTM), are ordinaries used in deep learning approaches. The main advantage of this technique is enabling us to collect data or generate output from prior knowledge. However, the big drawback is that the decision boundary may be overstrained if the training set does not own samples that should be in a class. To sum up, this technique is more straightforward than other techniques in learning with high-performance [87].

- **Artificial Neural Network (ANN)**

ANN is the initial nonparametric nonlinear time series model. It is also known as a feed-forward neural network because inputs are processed only forward. The basic structure of ANN consists of a network of computing units called neurons. These neurons are represented as nodes in artificial neural networks. The nodes are connected through weights. ANN has become widespread in research because ANN has flexible and capable of modelling nonlinear processes. ANN is useful when the data are non-stationary and have unknown statistical distribution. There are numerous ANN methods such as feed-forward back propagation (FFBP) [88], radial basis function-based neural networks (RBF) [89], generalized regression neural networks (GRNN) [90]. However, there are two limitations to using ANN, including overfitting issues due to large numbers of parameters to fix and little to no historical information about the importance of inputs in analyzed problems. The other is that ANN is ineffective for inference because the ANN process is not required training

for the whole data set [91].

- **Multilayer Perceptron (MLP)**

Multilayer perceptron (MLP) is a feedforward artificial neural network (ANN) class. A perceptron is a neuron network unit that helps categorize the input data. The MLP is characterized by an input layer, one or more hidden layers and an output layer. This model has high self-learning ability and fault tolerance. However, it has limitations in the learning process as the stock price pattern has tremendous amounts of noise and a high dimension [92]. Many scholars have developed the MLP model to predict future data especially in finance. Rajabi et al. [93] proposed learnable window size-multilayer perceptron(LWS-MLP) to predict bitcoin prices. The results indicate that the LWS-MLP model is superior to the other six methods: ARIMA, random forest (RF), support vector regression (SVR), LSTM stochastic, MLP stochastic, and wavenet methods. Peng et al. [94] compared the performance of three individual methods (ARIMA, MLP, RNN) and the hybrid methods (ARIMA-MLP, ARIMA-RNN) to forecast the weekly closing price of three stock exchanges. The finding reveals that hybrid models provide less error than individual methods in terms of MAE, RMSE, and MAPE.

Zhang et al. [95] presented the GA-MLP algorithm to seek the hyperparameter of the MLP model using the genetic algorithm (GA). Then, the diversity-considered GA-MLP ensemble algorithm (DGAMLPE) is presented to forecast the financial distress of Chinese listed companies. Two novel algorithms are compared with other five algorithms, namely random forest (RF), extreme gradient boosting (Xgboost), weighted count of errors and correct (WCEC), integrated of the unsupervised classifier deep belief network (DBN) and support vector machine in terms of accuracy, F1-score, and area under the curve (AUC). The empirical results demonstrated that the DGAMLPE algorithm outperformed the other seven algorithms in terms of accuracy, F1 score and area

under the curve. Finally, EL-SAID et al. [96] outlined an advanced squirrel search optimization algorithm (ASSOA) to search for the hyperparameter of the MLP. The proposed algorithm (ASSOS-MLP) is compared with three classifiers based on the MLP, including the basic squirrel search (SS) optimization algorithm (SS-MLP), grey wolf optimizer (GWO) optimization algorithm (GWO-MLP), and genetic algorithm (GA-MLP). The data used in this work are chest X-ray (Pneumonia-COVID-19) images from the Kaggle dataset containing 5,863 X-rays. The experimental results demonstrate that the proposed ASSOA - MLP algorithm performs very well in detecting chest X-ray COVID-19 images. Furthermore, the ASSOA algorithm provides better results than the SS, GWO, and GA algorithms because it gives the lowest average error and standard deviation.

The hyperparameter of the network is the number of layers and the number of neurons per layer. Several nonlinear activation functions, including sigmoid, hyperbolic tangent, rectified linear unit (ReLU), leaky-relu and softmax, are used in the MLP model. The structure of MLP is displayed in Figure 2.3.

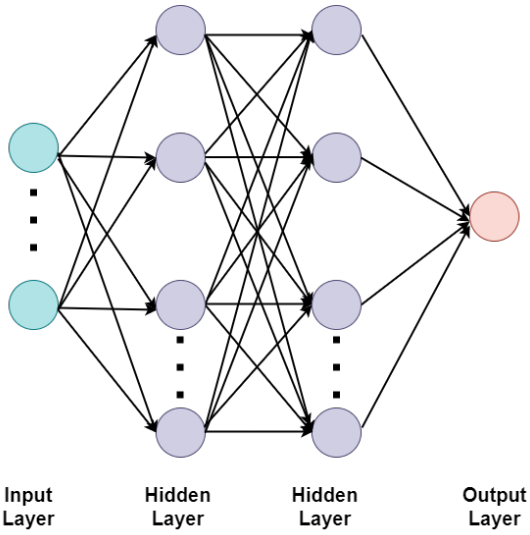


Figure 2.3: Architecture of Multilayer Perceptron [1]

A perceptron is a linear classifier. Each neuron contains three terms, namely input, weight and bias terms in the hidden layers. The general neuron unit is

defined as

$$y_i = \sigma(\sum_{j=1}^{n} w_j x_j + b), \tag{2.2}$$

where y_i are the outputs, σ is the nonlinear activation function, x_j are inputs to the neuron, $i, j = 1, 2, \ldots, n$, n is the number of the neuron, w_j are the weights and b is the bias [97]. In addition, the nonlinear activation functions are as follows [98]:

- **Sigmoid** $\sigma(z) = \frac{1}{1+e^{-z}}$
- **Hyperbolic tangent** $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$
- **Rectified Linear Unit** $R(z) = max(0, z) = \begin{cases} z, & \text{if } z > 0 \quad (Active\ state) \\ 0, & \text{if } z \leq 0 \quad (Inactive\ state) \end{cases}$
- **Leaky-ReLU** $R(z) = max(kz, z) = \begin{cases} z, & \text{if } z > 0 \quad (Active\ state) \\ kz, & \text{if } z \leq 0 \quad (Inactive\ state) \end{cases}$
- **Softmax** $f_j(z) = \frac{e^{z_j}}{\sum_{k=1}^{n} e^{z_k}}$

• **Convolutional Neural Network (CNN)**

A convolutional neural network (CNN) is a class of deep learning methods. There are three types of different dimensions of CNN, namely one-dimensional CNN (1D-CNN), two-dimensional CNN (2D-CNN) and three-dimensional CNN (3D-CNN), which are usually used for time series data, image data and 3D image data, respectively. The advantage of CNN compared to its predecessors is that it automatically identifies the relevant features without any human supervision [87]. Using the convolution function, CNN can extract essential and distinctive features from images. However, this method requires large memory and computation load because big data must be prepared [99]. CNN is most

effective in many fields including face recognition [100], object detection [101], recommender systems [102], medical image analysis [103], traffic flow [104] and stock prediction [20].

Focusing on stock prediction, Lu et al. [20] proposed a CNN-BiLSTM-AM for closing price prediction. The novel technique is built based on convolutional neural networks (CNN), bi-directional long short-term Memory (BiLSTM), and attention mechanism (AM). Then, the seven methods are utilised to compare with the proposed model. The finding shows that the CNN-BiLSTM-AM technique performs better than the other model in terms of the MAE, RMSE and R^2. Liang et al. [105] proposed a new hybrid between ICEEMDAN and LSTM-CNN-CBAM models to predict the gold price. The results confirm that the ICEEMDAN-LSTM-CNN-CBAM model provides better performance than the other hybrid models in terms of MAE, RMSE, MAPE, SMAPE, and R^2. Kanwal et al. [106] proposed a novel hybrid technique to predict stock prices. This model combines a bidirectional cuda deep neural network long short-term memory (BiCuDNNLSTM) and 1D-CNN model. The other four DL models, including LSTM, CuDNNLSTM, LSTM-CNN, and LSTM-DNN, are used to compare the performance of the proposed model. The results illustrate that the BiCuDNNLSTM-1DCNN method is a superior technique for predicting GDAXI and HSI stock prices.

The convolutional neural network (CNN) is a type of deep neural network (DNN) that consists of convolutional layers based on the convolutional operation. CNN mainly comprises two layers, namely the convolution layer and the pooling layer. The convolution layer is shown in the following formula:

$$l_t = tanh(x_t k_t + b_t), \tag{2.3}$$

where l_t denotes the output value after convolution, k_t presents the weight of the convolution kernel, b_t is the bias of the convolution kernel [20], and t corresponds to its order in the series.

CNN architectures have different layers, including convolutional, max-pooling, dropout, and fully connected multilayer perceptron (MLP) layer. The number of hidden layers, number of units per layer, network weight initialization, activation functions, learning rate, momentum values, the number of epochs, batch size (minibatch size), decay rate, optimization algorithms, dropout, kernel size, and filter size are hyperparameter of CNN model [107]. Figure 2.4 depicts the overall architecture of CNN.

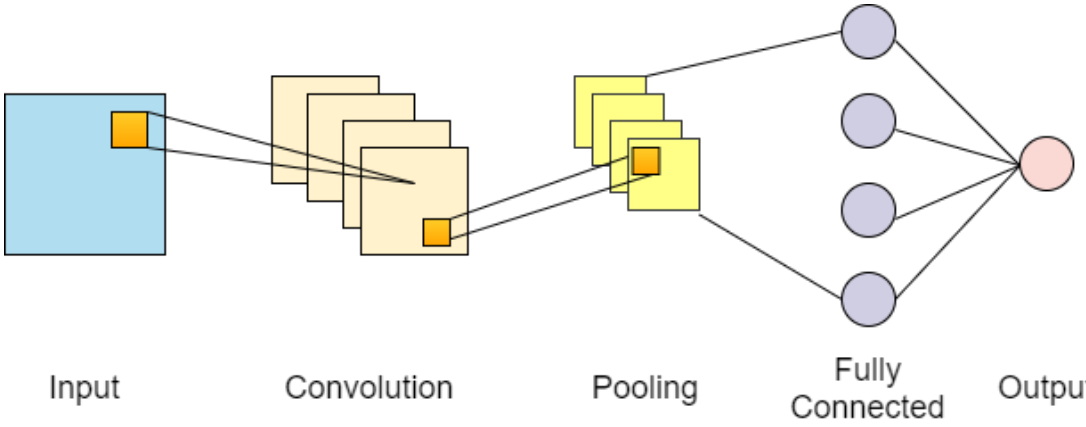


Figure 2.4: Fully Convolutional Neural Network Architecture [2]

• **Recurrent Neural Network (RNN)**

The recurrent neural network (RNN) is an alternative to CNN for processing sequential data and handling dynamic relationships and long-term dependencies [108]. For the basic idea of RNN, each input data are treated with the same function, whereas the output of the current input relies on its past computation. After the output is released, it is copied and sent back to the input to predict the layer's output. RNN is useful for numerous disciplines such as speech recognition [109], image captioning [110], machine translation [111], time series forecasting [112], and natural language processing [87]. However, RNN models cannot deal with long-term data dependency and require multiple parameters to update. Moreover, the parameter updates are poor when the slight gradient becomes too small. Therefore, it is difficult to make the learn-

ing of long-term data. Thus, a Long Short-Term Memory Network (LSTM) is presented to deal with gradient problems.

• **Long Short-Term Memory Network (LSTM)**

The long short-term memory (LSTM), introduced by Hochreiter and Schmidhuber [113], is presented to deal with the weakness of the exploding gradient problem from RNN. The primary objectives of LSTM are to capture long-term dependencies and define the optimal time delay (lag) in time series problems. The LSTM model has two subcategory states: a short-term state (similar to the RNN) and a long-term state. The information is collected to capture the long-term dependencies between the present and past hidden states over time. Traversing from left to right, the long-term state passes through a forget gate. Some memories are abandoned, and some of them are inserted using the addition operation. LSTM is insensitive to the input data length compared to other methods that handle a series of data [114]. The weakness of LSTM is that the big data and massive processing time for training are required [115].

LSTM is a robust recurrent neural network (RNN) because the network can remember both short- and long-term dependencies. The LSTM model architecture contains various numbers of layers and numerous units per layer. The number of hidden layers, the number of units per layer, regularization techniques, network weight initialization, activation functions, learning rate, momentum values, the number of epochs, batch size (minibatch size), and sequence length are the hyperparameter of the LSTM model [112, 116]. The LSTM structure is presented in Figure 2.5.

The following equations detail the operations of the single cell long short-term memory:

$$i_t = \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}), \tag{2.4}$$

$$f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}), \tag{2.5}$$

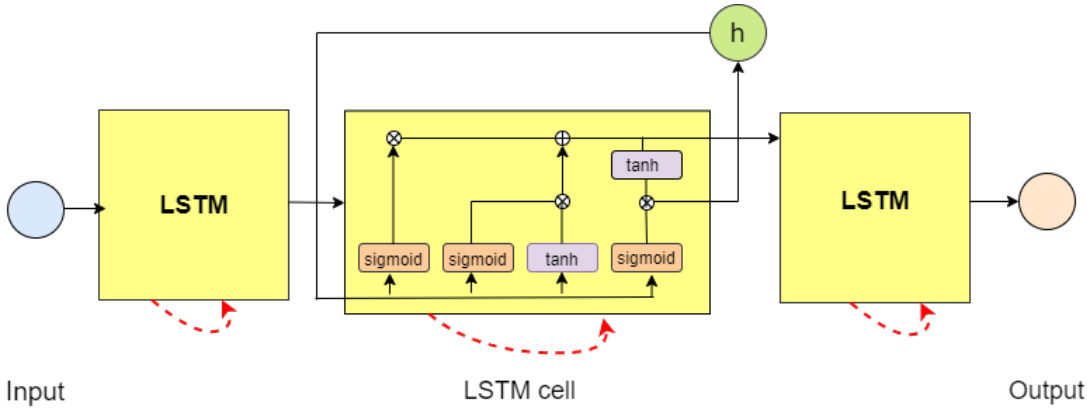


Figure 2.5: LSTM model architecture [3]

$$o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}), \tag{2.6}$$

$$c_t = i_t * [tanh(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)})] + f_t * c_{t-1}, \tag{2.7}$$

$$h_t = o_t * tanh(c_t). \tag{2.8}$$

Where x_t denote the input vector at instant time t, h_t represents the hidden state vector, σ displays logistic sigmoid function, tanh is a hyperbolic tangent function, i_t indicates the put gate vector, o_t is a output gate vector, c_t represents the memory cell state vector, f_t is a forget gate vector and , * presents the pointwise multiplication of two vectors [117].

LSTM has been effectively employed in universal studies such as time series prediction [118], wind power prediction [119], human trajectory prediction [120] and stock price prediction [121–125]. In 2022, many papers applied the LSTM model and combined LSTM with another technique for stock price forecasting. Bathla et al. [121] studied high variation price movement for seven months in 2020 (January-July). During this time, the price movements of popular stocks, including NSE, BSE, NYSE, Dow Jones, Nikkei 225, S&P 500, and NASDAQ, fluctuated. Moreover, only a few works consider the price movement and predict the stock price at that time. Therefore, the authors conducted the LSTM model with a specific model hyperparameter and compared it with the state-of-the-art techniques, namely the ARIMA, linear regression

and ARIMA-RNN methods. The comparison between the LSTM model and the state-of-the-art methods using the MAPE criterion indicates that it provides the lowest error (MAPE) and can be used to capture the high variation price movement.

Bhandari et al. [122] studied the single and multiple layers of the LSTM model for S&P 500 stock price prediction from 2006 to 2020. The data used in their work covered two significant events including the financial crisis in 2008 and the COVID-19 pandemic in 2020. The evaluation metrics utilised in this work are the RMSE, MAPE and the correlation coefficient (R). The experimental results for short-term and long-term closing price prediction show that the single LSTM model is a high-accuracy model and outperforms multilayer LSTM models. Ahmad and Singh [123] compared the machine learning and the deep learning models for the daily NIFTY50 stock prices forecasting. Linear regression, support vector machine and random forest are three machine learning used in their work. For deep learning techniques, LSTM and CNN are considered. The experimental results illustrate that the DL techniques perform better than the ML methods in terms of RMSE. Furthermore, the LSTM model is outstanding over the other techniques as this model provides a more accurate stock price prediction.

Liu et al. [126] proposed a new hybrid model between the convolutional autoencoder (CAE) and the long-short-term memory (LSTM) called the CAE-LSTM model for short-term prediction. The multivariate data used in their work are the opening, the highest, the lowest and the closing prices of the Shanghai Stock Exchange Index from 4 January 2000 to 27 May 2021. The comparison between the proposed model and the LSTM model indicates that the CAE-LSTM model provided better results than the LSTM model in terms of the RMSE.

Lin et al. [125] proposed a novel hybrid method for multi-step precious metal prices prediction. The MEEMD-LSTM is integrated based on the modi-

fied ensemble empirical mode decomposition (MEEMD) method and the long-short-term memory (LSTM) method. The well-known DL technique, including the multilayer perceptron (MLP), support vector regression (SVR) and a combination forecasting model super learner (SL) methods, are utilised to evaluate the performance of the MEEMD-LSTM model. The four metal time series used in this work are gold, silver, platinum, and palladium. The results show that the MEEMD-LSTM is a superior hybrid model for multi-step precious metal price prediction.

2.4 Deep Learning for Time Series with Delay

Delay is one of the key indicators of time series. Time delay plays an essential role, especially in forecasting problems. Delay, which is investigated in this thesis, refers to lag time (time window/timestep/look back). Some scholars work with delays in time series. For example, Gers et al. [127] applied the MLP and LSTM methods to deal with time series data when time windows are fixed. The univariate data used in their work are recorded from a Far-Infrared (FIR) Laser in a chaotic state, totaling 1,100 observations. The first 1,000 observations were used for training, and the remaining 100 points were used for testing. The results indicate that the MLP model outperformed the LSTM because the MLP model could detect chaotic behavior.

Ma et al. [128] presented a novel LSTM model for travel speed prediction. In order to capture the long-term temporal dependency, the proposed model is called the Long Short-Term Neural Network (LSTM NN). The authors also proposed a novel algorithm for searching for the optimal time window. The novel method was compared with classical time series prediction models: ARIMA, Kalman Filter, Support Vector Machine, Elman Neural Network, Time-Delay Neural Network, and Nonlinear Autoregressive with Exogenous Inputs (NARX) Neural Network. The multivariate travel speed data, includ-

ing volume, occupancy and speed, were collected from two major ring roads around Beijing in June 2013. The first 25 days were classified as a training set, and the rest five days were classified as a testing set. Empirical results indicate that speed prediction accuracy increases when an extensive time lag is considered. The LSTM NN performs better than other classical time series prediction techniques in capturing long-term dependency and automatically defining the optimal time lags.

Saud and Shakya [129] applied three popular deep learning methods, namely Vanilla RNN (VRNN), Long Short-term Memory (LSTM), and Gated Recurrent Unit (GRU) for next-day closing price prediction. The data used in this work are from Nepal. Investment Bank (NIB) and Nabil Bank Limited (NABIL) stocks. The multivariate variables, including Open, High, Low, Close, Trade Volume, Trade Amount, High-Low, Close-Open, 3day MA, 10day MA, 30day MA, Standard Deviation, Relative Strength Index, and William %R are input variables. The next-day forecast varies based on different steps. The results show that the GRU performs better than the LSTM and the VRNN in MAPE. However, forecasting performance may not be improved by increasing the time step.

Kang et al. [130] study on wastewater flow rate prediction using the Bidirectional LSTM (bi-LSTM) method. To evaluate the performance of the bi-LSTM method, three popular deep learning methods, namely SVM, GRU, and LSTM, are employed. The wastewater flow rates are obtained from the Yangju wastewater treatment plant from 19th August 2017 to 18th September 2017, with a total of 4,464 observations. A bi-LSTM based prediction idea is to use a current slide window from the training period for m-step ahead forecasting. The results indicate that bi-LSTM performs better than the well-known deep learning methods in RMSE.

Wu et al. [131] presented a Transformer-based method for predicting influenza prevalence. The novel method is compared with the ARIMA, LSTM,

and sequence-to-sequence (Seq2Seq) model. The influenza prevalence data were gathered from the Centers for Disease Control and Prevention (CDC) between 2010 and 2018. The empirical results by fixing the length time window show that the Transformer-based method outperformed ARIMA, LSTM, and Seq2Seq-based models in terms of RMSE.

Fan et al. [132] focus on short-term predictions of building energy based on three techniques, including recursive, direct, and multi-input, multi-output (MIMO) techniques constructed on RNN, LSTM, and GRU methods. The multivariate time series used in this work are building operations observations from an educational building in Hong Kong in 2015 over 17,000 points. The experiments demonstrated that the direct approach based on recurrent models performed well in terms of computation time.

Marino et al. [133] compare the performance of standard LSTM and LSTM-based Sequence to Sequence (S2S) architecture for building-level energy load forecasting. Individual household electric power consumption time series used in their work are extracted from a benchmark electricity consumption dataset for a single residential customer with two different time scales, namely one-minute and one-hour time intervals. The experimental results indicate that on both time scales, the S2S LSTM-based algorithms perform well in terms of RMSE.

Vijayakumar et al. [134] investigated network traffic prediction from the GÉANT backbone networks in 2004 by utilizing feed-forward networks (FFN), recurrent neural networks (RNN), long short-term memories (LSTM), gated recurrent units (GRU), and identity recurrent units (IRNN). Three traffic data sets were collected: the training set, the validation set, and the testing set, consisting of 370,300 observations, 105,800 observations and 158,700 observations, respectively. According to the results, the LSTM method performs better than the other techniques in terms of MSE. An overview of the delay in time series prediction using deep learning techniques is provided in Table 2.3.

Table 2.3: A summary of the literature on delay in time series prediction using deep learning methods is presented by data types, forecasting problems, and evaluation metrics.

Reference	Method	Data		Forecasting		Evaluation metrics			
		Univariate	Multivariate	Single-step	Multi-step	NMSE	MAPE	MSE	RMSE
Gers et al. [127]	LSTM,MLP	✓		✓		✓			
Ma et al. [128]	LSTM NN		✓	✓			✓		
Saud and Shakya [129]	Vanila RNN, LSTM, GRU		✓	✓			✓		
Kang et al. [130]	Bi-LSTM		✓		✓				✓
Wu et al. [131]	Transform-based LSTM, ARIMA	✓		✓					✓
Fan et al. [132]	RNN, LSTM,GRU		✓		✓				✓
Mario et al. [133]	LSTM, S2S-LSTM	✓		✓					✓
Vinayakumar et al. [134]	FFN, RNN, LSTM GRU, IRNN		✓	✓				✓	

2.5 Conclusion

In this chapter, we present a comprehensive literature review. First, time series analysis is explained. An overview of Box-Jenkins, neural networks, and hybrid methods that combine Box-Jenkins and neural networks is provided. We then describe the background of the stochastic differential equation (SDE), including adding delay to the SDE model and volatility. A summary of published works based on time series analysis and the SDDE is presented. Next, deep learning techniques in time series forecasting are reviewed. It covers supervised and unsupervised approaches. In addition, it includes publications that apply deep learning methods to various time series problems. The PRISMA diagram represents a wide range of systematic reviews and meta-analyses of time series analysis with deep learning methods. The review of deep learning for time series with delay is outlined in the last section. It covers single deep learning techniques and hybrid deep learning techniques for predictions in numerous areas. Finally, a literature review summary is displayed based on different data types, forecasting problems, and evaluation metrics.

Chapter 3

The m-delay Autoregressive Model

In this chapter, we propose the m-delay AR model in which only two parameters are determined. This model is a special case of the classical autoregressive (AR) model. As the classical AR model uses more parameters in the formula to predict future observation, the idea of ignoring some parameters is presented. The m-delay AR formula based on the least squares method is derived, and an optimal delay algorithm based on a brute-force technique is developed. The organization of this chapter is structured as follows. Section 3.1 concerns the derivation of the m-delay AR model to obtain two parameters of the m-delay AR model. The performance of parameter estimation is presented in Section 3.2, and the empirical study is reported in Section 3.3. Conclusion is given in Section 3.4. Note that the work in this chapter has been published in [135].

3.1 Derivation of the m-delay Autoregressive Model

Generally, an explicit formula of the present value x_t is determined by the standard AR model [43], i.e.

$$x_t = \epsilon_t + \sum_{i=1}^{m} \phi_i x_{t-i}, \tag{3.1}$$

where x_t represents the present value at instant time t, $\{x_{t-1}, x_{t-2}, \ldots, x_{t-m}\}$ is the list of the past observations, $\phi_i (i = 1, \ldots, m)$ are i^{th} coefficients of AR model and ϵ_t is a gaussian white noise process which is assumed to be the normal distribution, $N(0, \sigma^2)$.

In this study, we propose a m-delay AR (MAR) model to approximate the present value x_t by using only the first term at $t-1$ and the last term at $t-m$,

$$x_t = \epsilon_t + \phi_1 x_{t-1} + \phi_m x_{t-m}. \tag{3.2}$$

The first and the last coefficients of the AR model are estimated by the below equation based on the least squares method. The principle concept of the least squares procedure is to minimize the sum of square error functions [136].

$$S_c(\hat{\phi}_1, \hat{\phi}_m) = \sum_{t=m+1}^{n} \left[x_t - \hat{\phi}_1 x_{t-1} - \hat{\phi}_m x_{t-m} \right]^2. \tag{3.3}$$

To find the optimal values of $\hat{\phi}_1$ and $\hat{\phi}_m$ in Eq. (3.3). We differentiate Eq. (3.3) with respect to $\hat{\phi}_1$ and $\hat{\phi}_m$ and set them to zero. This step is shown in Eq. (3.4).

$$\begin{cases} \dfrac{\partial S_c}{\partial \hat{\phi}_1} &= -2 \sum\limits_{t=m+1}^{n} \left[x_t - \hat{\phi}_1 x_{t-1} - \hat{\phi}_m x_{t-m} \right] x_{t-1} = 0, \\ \dfrac{\partial S_c}{\partial \hat{\phi}_m} &= -2 \sum\limits_{t=m+1}^{n} \left[x_t - \hat{\phi}_1 x_{t-1} - \hat{\phi}_m x_{t-m} \right] x_{t-m} = 0. \end{cases} \tag{3.4}$$

We finally obtain the m-delay formula for approximating $\hat{\phi}_1$ and $\hat{\phi}_m$, i.e.

$$\begin{cases} \hat{\phi}_1 = \dfrac{\sum_{t=m+1}^{n} x_t x_{t-1} \sum_{t=m+1}^{n} x_{t-m}^2 - \sum_{t=m+1}^{n} x_{t-1} x_{t-m} \sum_{t=m+1}^{n} x_t x_{t-m}}{\sum_{t=m+1}^{n} x_{t-1}^2 \sum_{t=m+1}^{n} x_{t-m}^2 - \left[\sum_{t=m+1}^{n} x_{t-1} x_{t-m}\right]^2}, \\ \hat{\phi}_m = \dfrac{\sum_{t=m+1}^{n} x_{t-1}^2 \sum_{t=m+1}^{n} x_t x_{t-m} - \sum_{t=m+1}^{n} x_t x_{t-1} \sum_{t=m+1}^{n} x_{t-1} x_{t-m}}{\sum_{t=m+1}^{n} x_{t-1}^2 \sum_{t=m+1}^{n} x_{t-m}^2 - \left[\sum_{t=m+1}^{n} x_{t-1} x_{t-m}\right]^2}. \end{cases} \tag{3.5}$$

We call $\hat{\phi}_1$ and $\hat{\phi}_m$ as the new approximation of the m-delay AR coefficients. As the standard AR model is a stationary time series process, we now determine the stationarity condition of the m-delay AR model. In this study, we investigate the stationarity condition of our proposed model in Eq. (3.2) by computing the roots of AR characteristic equation.

To discuss the stationarity, we define the AR characteristic polynomial as

$$\phi(t) = 1 - \phi_1 t - \phi_m t^m, \tag{3.6}$$

where the characteristic equation is

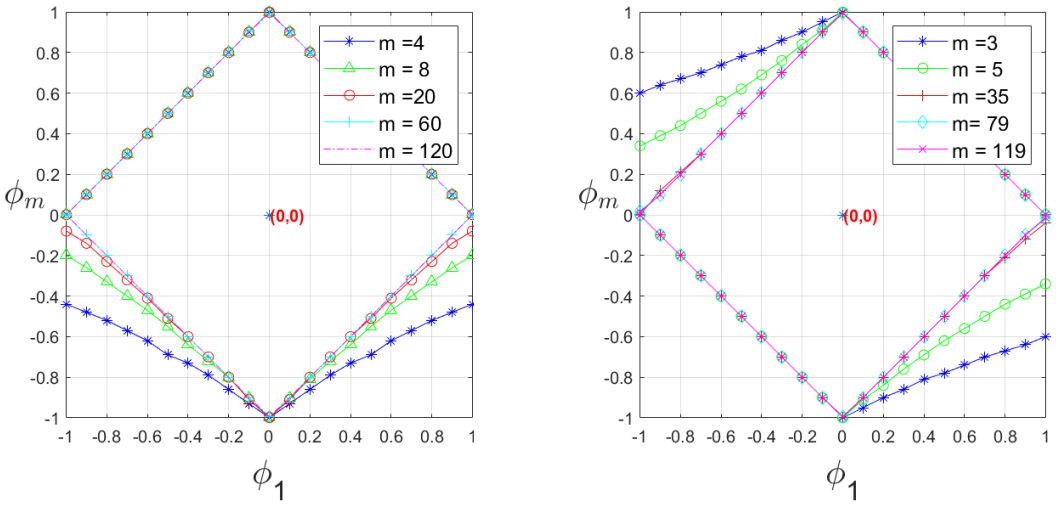
$$1 - \phi_1 t - \phi_m t^m = 0. \tag{3.7}$$

To solve Eq. (3.7) a sufficient stationarity condition of our model is

$$\mid \phi_1 \mid + \mid \phi_m \mid < 1, \tag{3.8}$$

where the delay (m) is between 3 and 120.

This study selected a delay range from 3 to 120 in order to account for small delays (m = 3,5,20), medium delays (m = 35,60,79) and large delays (m = 119,120). The sufficient stationarity condition is shown in Figure 3.1.



(a) (b)

Figure 3.1: Linear relationship between ϕ_1 and ϕ_m for different delays (m); (a) m is even number, (b) m is odd number

We then evaluate the performance on the MAR model using the root mean square error. The formula is given as

$$RMSE = \sqrt{\frac{\sum_{l=1}^{L}\left[\left(\phi_1 - \hat{\phi}_1^{(l)}\right)^2 + \left(\phi_m - \hat{\phi}_m^{(l)}\right)^2\right]}{L}}, \tag{3.9}$$

where ϕ_1 and ϕ_m are parameters of the MAR model, $\hat{\phi}_1$ and $\hat{\phi}_m$ are approximated parameters, m is the delay and L is the number of simulations runs.

3.2 Parameter Estimation

This section concerns the effectiveness of the m-delay formula Eq. (3.5) via Monte Carlo simulations, and the optimal delay using brute-force technique.

3.2.1 Effectiveness of the m-delay Formula

To examine the coefficients ϕ_1 and ϕ_m obtained from the m-delay formulation

(see Eq. (3.5)), we set the initial value of m data to zero $(m \ll n)$. The observations x_t determined by Eq. (3.2) are generated using Monte Carlo technique. Algorithm 1 presents the examination process using computation scheme as shown in Table 3.1. We perform the same process for the different sample sizes with different delays.

Algorithm 1

Require: delay (m), sample size (n), the m-delay AR parameter (ϕ_1, ϕ_m), set initial iteration $(l = 0)$, the maximum number of simulations runs $(L = 1,000)$, ε = 0.0001, initial summation of error $(SUM_E = 0)$ and set $x_t = 0$ for t = 1,2,...,m

while($l < L$**)**
    for $t = m + 1$ to n
    $x_t = \phi_1 x_{t-1} + \phi_m x_{t-m} + \epsilon_t$
    end for
    compute $\hat{\phi}_1, \hat{\phi}_m$ using Eq. (3.5)
    compute $E = \sqrt{(\phi_1 - \hat{\phi}_1)^2 - (\phi_m - \hat{\phi}_m)^2}$
    $SUM_E = SUM_E + E$
    RMSE = SUM_E / l
      if RMSE $> \varepsilon$
       $l = l$ +1
       end if
end while
print RMSE

Table 3.1: Computation scheme

Sample size	50	100	300	500	1,000	2,000	5,000	10,000
Delay	5	5	5	5	5	5	5	5
	20	20	20	20	20	20	20	20
			120	120	120	120	120	120

Figure 3.2 presented the estimated two parameters (ϕ_1 and ϕ_m) with 1,000 iterations when the sample size is 300. The simulation results indicate that the average $\hat{\phi}_1$ and $\hat{\phi}_m$approach the actual ϕ_1 and $\phi_m$for three delays.

Figure 3.3 illustrates the relationship between average two m-delay coefficients, $\hat{\phi}_1$ and $\hat{\phi}_m$, and sample size for three different delays including $m =$

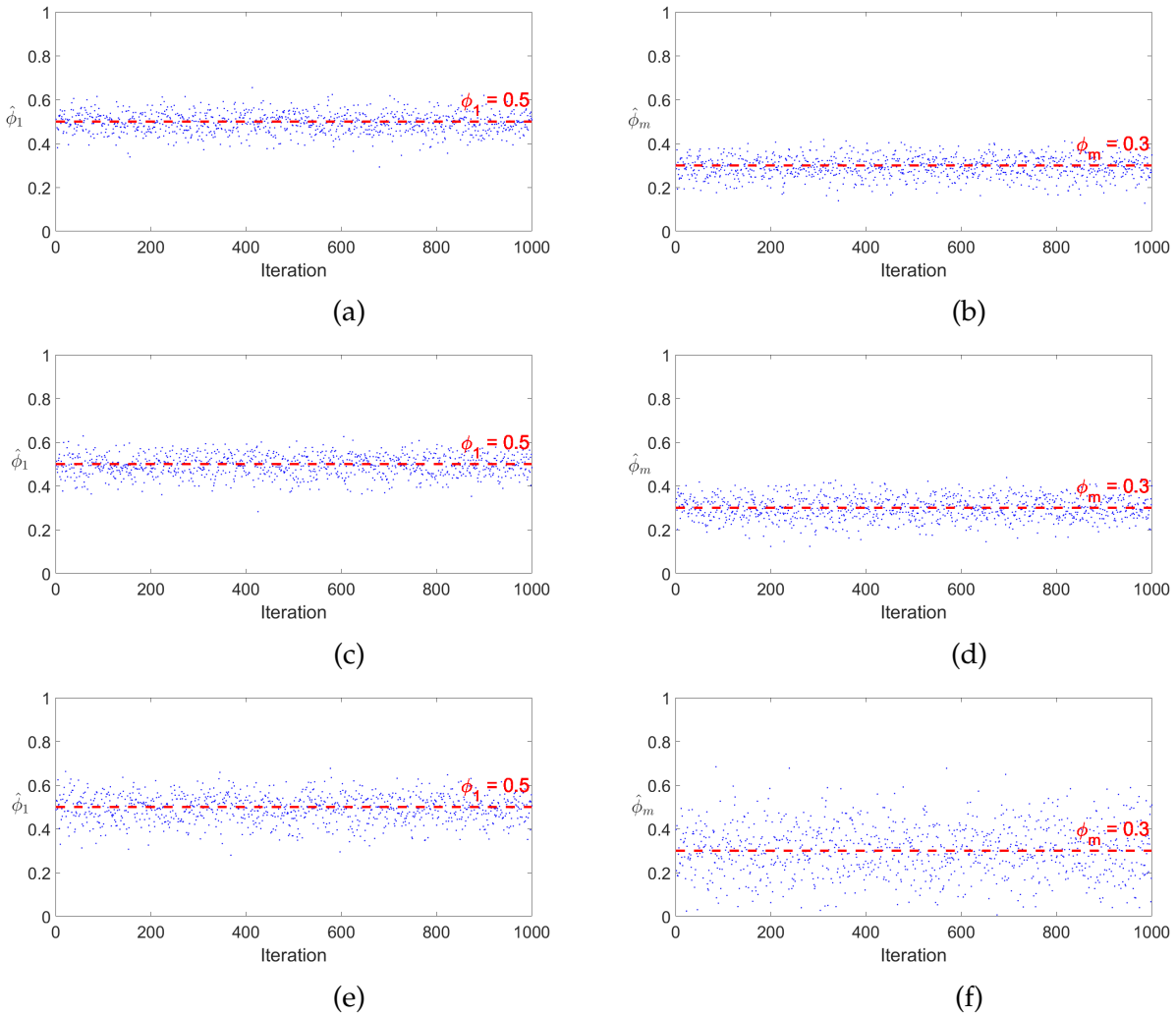
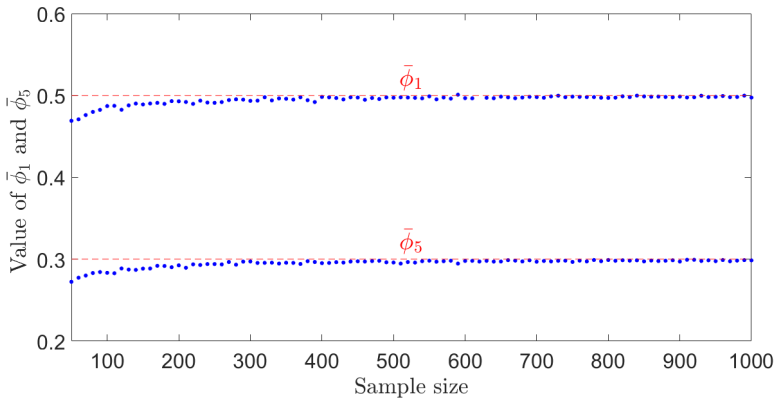
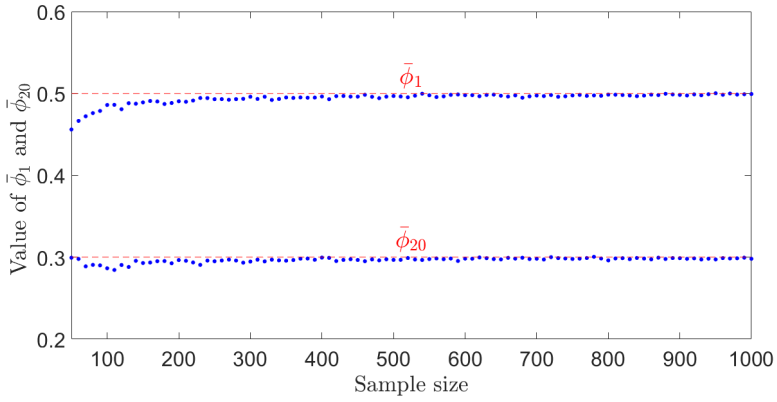


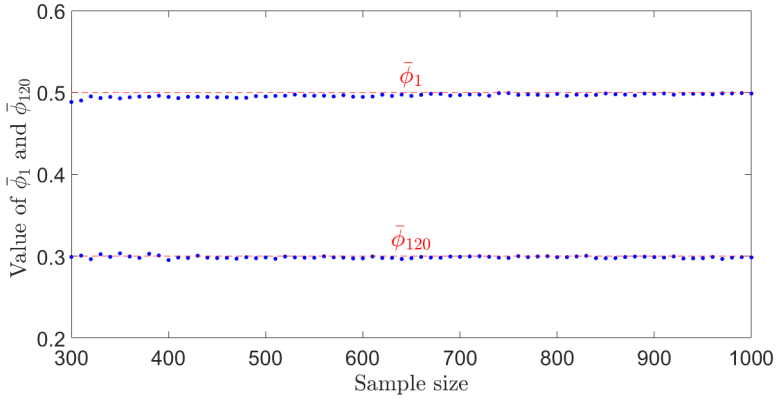
Figure 3.2: Scatter plots of the estimated $\hat{\phi}_1$: (a) $m = 5$; (c) $m = 20$; (d) $m = 120$ along with scatter plots of the estimated $\hat{\phi}_m$: (b) $m = 5$; (d) $m = 20$; (f) $m = 120$



(a)



(b)



(c)

Figure 3.3: Scatter plots of the average m-delay coefficients $\hat{\phi}_1$ and $\hat{\phi}_m$: (a) $m = 5$; (b) $m = 20$; (c) $m = 120$.

5, 20, 120. For the case of delay 5 and 20, the sample size starts from 50 to 1,000. Figure 3.3 indicates that sample size greater than 300 gives the reasonable results of approximated of $\bar{\phi}_1$ and $\bar{\phi}_m$. In case of the delay 120, the process starts with the sample size of 300. The estimated $\bar{\phi}_1$ and $\bar{\phi}_m$ approach the actual ones when the sample size about 500. It is noted that Eq. (3.5) is an effective formula for approximating the m-delay parameter ϕ_1 and ϕ_m.

3.2.2 Determination of Optimal Delay (m)

In this part, we present the Algorithm 2 to seek the optimal delay using the brute-force technique.

The Algorithm 2 begins with prescribing input parameters including the sample size (n), the initial delay (m_0) and the actual m-delay AR parameters (ϕ_1, ϕ_{m_0}) and setting the values of all m_0 observations to zero. We then generate the data set (x_t) by Eq. (3.2). The process starts with a delay of three at the first iteration. Two unknown parameters $(\hat{\phi}_1,\hat{\phi}_m)$ are calculated by Eq. (3.5) and the minimum error $(minE)$ is obtained. We repeat the process by increasing the delay by one. In each iteration, the minimum error $(minE)$ and minimum delay $(minD)$ are obtained. The process stops when the delay equals the prescribed delay. The process will succeed when the minimum delay equals the initial delay, i.e. the optimal delay (m_{op}) is obtained for the sample size n. This algorithm stops when the number of iterations reaches the maximum number of iterations. Finally, the accuracy of delay estimation based on the brute-force technique is computed by Eq. (3.10).

To illustrate the performance of a brute-force technique. We compute the accuracy on the simulated data set for each different sample size. The accuracy of the delay estimation is computed by

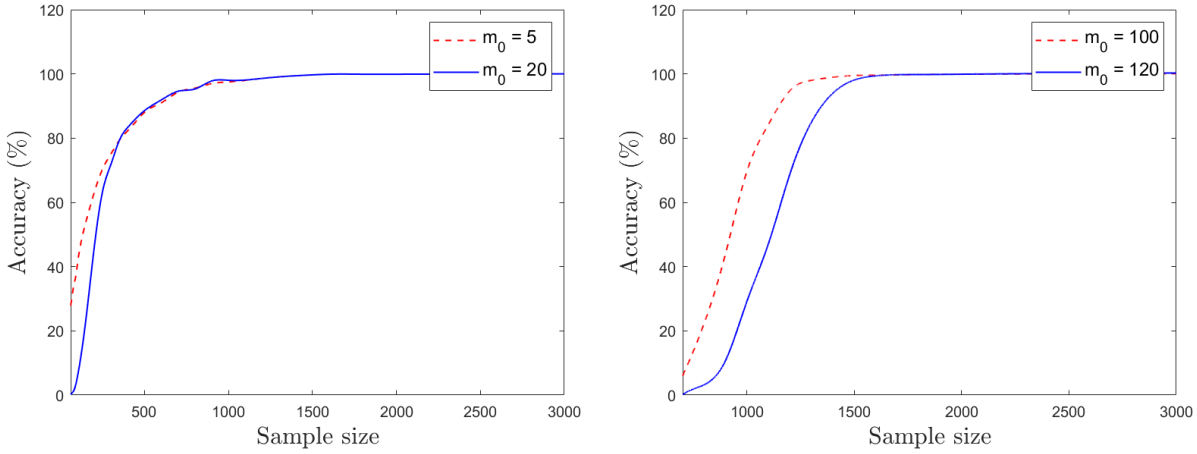
$$Accuracy(\%) = \frac{\text{Number of success outcomes}}{\text{Number of possible outcomes}} \times 100. \tag{3.10}$$

Algorithm 2

Require: sample size (n), initial delay (m_0), the m-delay AR parameter (ϕ_1, ϕ_{m_0}), count = 0, set initial iteration $(l = 0)$, the maximum number of iterations $(L = 10,000)$ and set $x_t = 0$ for t = 1,2,...,m_0

while($l < L$**)**
  for $t = m_0 + 1$ to n
  $x_t = \phi_1 x_{t-1} + \phi_m x_{t-m}$
  end for
  set $m = 3$
  compute $\hat{\phi}_1, \hat{\phi}_3$ using Eq. (3.5)
  compute $E_3 = \sqrt{\frac{\sum_{t=m+1}^{n} (\hat{x}_t - x_t)^2}{n-m}}$
  set $minE = E_3$
  for $m = m + 1$ to $\frac{n}{2} - 1$
    compute $\hat{\phi}_1, \hat{\phi}_m$ using Eq. (3.5)
    compute $\hat{x}_t = \hat{\phi}_1 x_{t-1} + \hat{\phi}_m x_{t-m}$
    $E_m = \sqrt{\frac{\sum_{t=m+1}^{n} (\hat{x}_t - x_t)^2}{n-m}}$
      if $E_m < MinE$
      $minE = E_m$
      $minD = m$
      end if
  end for
  if $minD < m_0$
  count = count +1
  $m_{op} = minD$
  end if
end while
compute the accuracy using Eq. (3.10)

Using a brute-force algorithm, our experiments show that sample size has a significant effect on the accuracy of the m-delay AR model as shown in Figure 3.4.



(a) (b)

Figure 3.4: The relationship between the percentage of accuracy and sample sizes of the m-delay approximation.

To investigate the effect of sample size on the accuracy of the m-delay approximation as revealed in Figure 3.4a, we choose 24 sample sizes between 60 and 3,000 giving the percentage of accuracy from 27.85% to 99.90% and 0.22% to 99.90% for delay 5 and 20, respectively. The percentage of accuracy with delay 100 and 120 is shown in Figure 3.4b.

We found that using sample size from 700 to 3,000 gives the percentage of accuracy 6% to 99.90% and 0.10% to 99.90%, respectively.

3.3 Empirical Study

In this section, we first test the performance of the MAR model comparing with the AR model as shown in Algorithm 3. We then apply both models to approximate the monthly mean minimum temperature in Perth, Western Australia.

3.3.1 Monte Carlo Simulations

Algorithm 3 below compares the performance of the MAR model (based on only 2 parameters namely ϕ_1 and ϕ_m) with the AR model (via m coefficient parameter, $\phi_1, \phi_2, \ldots, \phi_m$).

This algorithm starts with determining input parameters including sample size (n), actual parameters of AR with delay process (ϕ_1, ϕ_{m_0}) and setting the values of all m_0 observations to zero. We then generate the data set (x_t) by Eq. (3.2). To find the root mean square error from the AR model $(RMSE_{AR})$, we first estimate the model parameter and compute the root mean square error. For the MAR model, the computation process of the optimal delay starts at delay 3. The unknown parameters ($\hat{\phi}_1$,$\hat{\phi}_m$) are calculated by Eq. (3.5), and the minimum error $(minE)$ and the minimum delay $(minD)$ are obtained. The process is repeated until the delay reaches the prescribed delay $\frac{n}{2} - 1$. This step returns the optimum delay with the smallest value of the error for each iteration (m_{op} = minD) and we also calculate $RMSE_{MAR}$ using the optimal delay (m_{op}). This process repeats until the total iterations equal the maximum number of iterations. Finally, we compute and compare the average root mean square error obtained from both models, as shown in Table 3.2.

The experimental results shown in Table 3.2 are generated using $\phi_1 = 0.5, \phi_m = 0.3, m_0 = 10$ and $L = 10,000$. We now use the independent sample t-test to confirm that the results obtained from both models are not significantly different. The null hypothesis and alternative hypothesis are $\overline{RMSE}_{AR} = \overline{RMSE}_{MAR}$ and $\overline{RMSE}_{AR} \neq \overline{RMSE}_{MAR}$, respectively.

Algorithm 3

Require: sample size (n), initial delay (m_0), the AR parameter (ϕ_1, ϕ_{m_0}), count = 0, set initial iteration $(l = 0)$, the maximum number of iterations $(maxL)$, set $L = 0$, $sumE_{AR} = 0$, $sumE_{MAR} = 0$ and set $x_t = 0$ for t = 1,2,. . . ,m_0

while($l < maxL$**)**
  for $t = m_0 + 1$ to n
    $x_t = \phi_1 x_{t-1} + \phi_m x_{t-m}$
  end for
find $RMSE_{AR}$
  Estimate the coefficient of the AR model$(\hat{\phi}_1, \hat{\phi}_2, \ldots, \hat{\phi}_{m_0})$
  compute $RMSE_{AR}$
find $RMSE_{MAR}$
  set $m = 3$
  compute $\hat{\phi}_1, \hat{\phi}_3$ using Eq. (3.5)
  compute $E_3 = \sqrt{\frac{\sum_{t=m+1}^{n} (\hat{x}_t - x_t)^2}{n-m}}$
  set $minE = E_3$
  for $m = m + 1$ to $\frac{n}{2} - 1$
    compute $\hat{\phi}_1, \hat{\phi}_m$ using Eq. (3.5)
    compute $\hat{x_t} = \hat{\phi}_1 x_{t-1} + \hat{\phi}_m x_{t-m}$
    $E_m = \sqrt{\frac{\sum_{t=m+1}^{n} (\hat{x}_t - x_t)^2}{n-m}}$
      if $E_m < MinE$
      $minE = E_m$
      $minD = m$
      end if
  end for
  if $minD < m_0$
  count = count +1
  m_{op} = $minD$
  end if
  compute $RMSE_{MAR}$ using m_{op}
end while
Compute average root mean square error of both models ($\overline{RMSE}_{AR}$ and $\overline{RMSE}_{MAR}$

Table 3.2: Average root mean square of the AR model and the MAR model for different sample sizes.

size	AR	MAR	**size**	AR	MAR	**size**	AR	MAR
30	0.09364	0.08147	650	0.09978	0.09916	4000	0.09999	0.09989
50	0.09703	0.08897	700	0.09959	0.09929	4500	0.09996	0.09978
80	0.09834	0.09320	750	0.09983	0.09930	5000	0.09997	0.09989
100	0.09871	0.09468	800	0.09989	0.09939	5500	0.09996	0.99887
150	0.09908	0.09635	850	0.09981	0.09934	6000	0.09998	0.09991
200	0.09939	0.09737	900	0.09993	0.09948	6500	0.09999	0.09993
250	0.09953	0.09793	950	0.09987	0.09945	7000	0.09999	0.09993
300	0.09955	0.09820	1000	0.09988	0.09948	7500	0.09997	0.09992
350	0.09962	0.09847	1500	0.09990	0.09963	8000	0.09996	0.09991
400	0.09978	0.09878	2000	0.09992	0.09972	8500	0.10000	0.09995
450	0.09969	0.09880	2500	0.09994	0.09978	9000	0.09996	0.09992
500	0.09982	0.09901	3000	0.09995	0.09982	9500	0.09998	0.09994
550	0.09977	0.09905	3500	0.09995	0.09983	10000	0.09999	0.09995
600	0.09975	0.09908						

The result of hypothesis testing is demonstrated in Table 3.3.

Table 3.3: Hypothesis test using t-test.

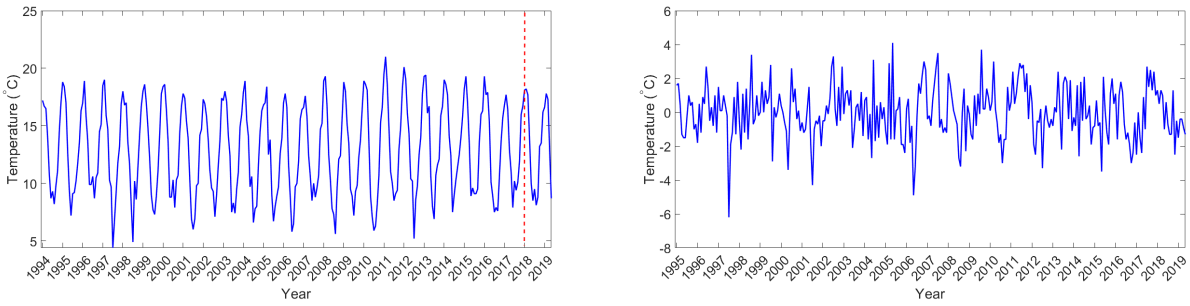
Method	N	Mean	SD	df	t-test	p-value
AR	40	0.0995	0.0011	78	-0.945	0.348
MAR	40	0.1208	0.1425			

It indicates insufficient evidence to reject the null hypothesis at a significance level of 0.05 as a p-value > 0.05. This shows no difference between the means of the two methods. However, the classical AR model requires m parameters, while the MAR method only requires two parameters. As a result, it can be helpful when m is large. Due to this advantage, our proposed MAR model is an efficient time series prediction method compared to the classical AR model.

3.3.2 Validation of the m-delay AR Model

This section concerns the prediction of the monthly mean minimum temperature in Perth, Western Australia, using the data obtained from the Bureau

of Meteorology, Australia between January 1994 and June 2019 containing 306 observations. The data are divided into two sets; training set and test set. The training set, from January 1994 to December 2017, has 288 observations. The rest, January 2018 to June 2019, is the testing set consisting of 18 observations. Figure 3.5a presents time series plot of the original observations. The dash line divided data into training and test sets. As the training data has a seasonal pattern as shown in Figure 3.5a, we firstly deseasonalize the original data by taking the seasonal difference $(x_t - x_{t-12})$. The result after transformation is shown in Figure 3.5b.



(a) (b)

Figure 3.5: Monthly mean minimum temperature in Perth from January 1994 to December 2017 : (a) Original data; (b) transformed data.

As the AR model requires stationary data, we apply the Augmented Dickey Fuller (ADF) test to test the stationarity of the monthly mean minimum temperature series. The results obtained from the original and the transformed data are displayed in Table 3.4.

Table 3.4: ADF unit root test results.

Data	ADF	p-value
Original data	-1.5547	0.1133
Transformed data	-13.6170	0.0010

The results of Table 3.4 show that the null hypothesis of non-stationarity is accepted at a 5% significance level (p-value = 0.1133 $>$ 0.05) for the original data. We then difference the data and applied the test again. The p-value of

the transformed data is lower than the significance level (p-value = 0.0010 $<$ 0.05). It indicates that the transformed data are stationary.

Next, we check the features of the transformed data: normal and independent [137]. To check whether normal distribution of data, we apply the Shapiro-Wilk test. The result is revealed in Table 3.5. Table 3.5 indicates that

Table 3.5: Normality test for transformed data.

Variable	Shapiro-Wilk Test		
	Statistics	df	p-value
Transformed data	0.993	276	0.234

transformed data are normally distributed as p-value = 0.234. We now check the independent of the transformed data by using one sample t-test. The result is shown in Table 3.6. As shown in Table 3.6, the p-value of 0.911 illustrates

Table 3.6: Independent testing for one-sample group using t-test.

Variable	n	Mean	SD	df	t	p-value
Transformed data	276	0.011	1.610	275	0.112	0.911

that the data are independent. We now can use these transformed data that are normally distributed and independent to find the suitable AR model.

Classical AR(9) Model

It appears that the transformed data are suitable for this study. From our experiment, it is found that the classical AR(9) model is applicable to this data set when $\hat{\phi}_1 = 0.1871, \hat{\phi}_2 = 0.1916, \hat{\phi}_3 = -01097, \hat{\phi}_4 = 0.0628, \hat{\phi}_5 = -0.0229, \hat{\phi}_6 = -0.0789, \hat{\phi}_7 = 0.0938, \hat{\phi}_8 = -0.0336$ and $\hat{\phi}_9 = -0.00029934$.

Table 3.7: Normality test of the residuals from the classical AR order 9.

Variable	Shapiro-Wilk Test		
	Statistics	df	p-value
Residuals	0.993	276	0.221

Table 3.8: One-sample t-test of the residuals from the classical AR order 9.

Variable	n	Mean	SD	df	t	p-value
Residuals	276	0.009	1.534	275	0.094	0.925

In Table 3.7 and Table 3.8, the residuals of the AR(9) are tested. The statistical testing results of the residuals from the classical AR(9) based on the Shapiro-Wilk test (p-value = 0.221) and one sample t-test (p-value = 0.925) demonstrate that the residuals obtained from the classical AR (9) are normally distributed and independent.

The m-delay AR (MAR) Model

From Eq.(3.5) and using the transformed data, we obtain the m-delay coefficient $\hat{\phi}_1 = 0.194561$ and $\hat{\phi}_9 = 0.027676$ which satisfy the characteristic Eq. (3.7) with inequality condition $\mid \phi_1 \mid + \mid \phi_m \mid < 1$.

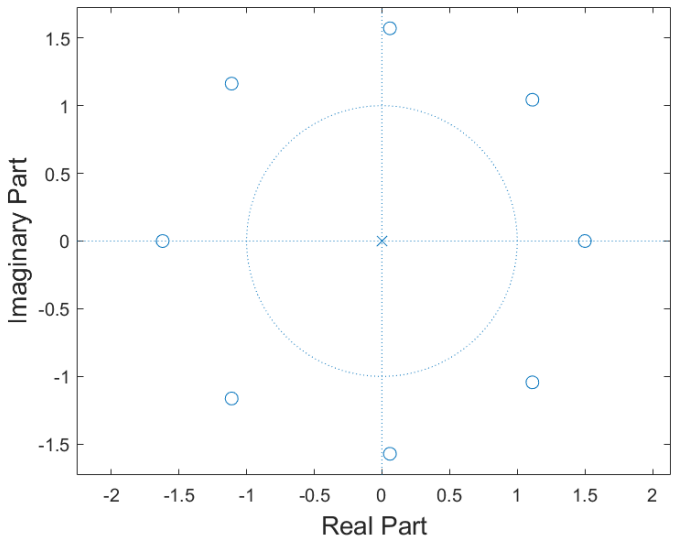


Figure 3.6: All roots of the characteristic obtained from Eq. (3.7) in the MAR model

From Figure 3.6, we can find that all roots of Eq. (3.7) lie outside the unit circle. This confirms the stationarity condition of the proposed the MAR model. The analyses of residuals obtained from the MAR model using Shapiro-Wilk

test and one sample t-test are presented in Table 3.9 and Table 3.10.

Table 3.9: Normality test of the residuals obtained from the MAR model.

Variable	Shapiro-Wilk Test		
	Statistics	df	p-value
Residuals	0.995	276	0.485

Table 3.10: One-sample t-test of the residuals from the MAR model.

Variable	n	Mean	SD	df	t	p-value
Residuals	276	0.010	1.579	275	0.106	0.916

The results from Table 3.9 and Table 3.10 illustrate that the residuals from the MAR model are normally distributed and independent with p-values of 0.485 and 0.916, respectively.

Predictive Modeling

Using the test set of data with 18 observations as shown in column 2 in Table 3.11 from January 2018 to June 2019, we apply the AR(9) model and the proposed MAR model to predict the monthly mean minimum temperature as shown in columns 3 and 4, respectively.

We then plot the forecasting monthly mean minimum temperature obtained from the MAR model and the AR model with observed data. The scatter plot is given in Figure 3.7.

As the number of predicted observations on the test set are small ($n = 18$), the difference among two methods are investigated with the Mann-Whitney U test. The null hypothesis (H_0) : there is no significant between two methods against the alternative (H_1) : there is significant between two methods [138]. The statistics testing is given in Table 3.12.

From the result as shown in Table 3.12, it indicates that there is not enough evidence to reject the null hypothesis (p-value = 1.000 $>$ 0.05). Consequently, it can be conclude that there is no significant difference in the predicted mean

Table 3.11: Forecasting monthly mean minimum temperature in Perth obtained from the MAR model and the AR(9) model.

Date	Observed	Predicted values	
		The MAR model	The AR(9) model
January, 2018	18.1	16.9	17.1
February, 2018	18.2	18.0	18.1
March, 2018	17.7	16.5	16.6
April, 2018	13.9	13.2	13.2
May, 2018	10.0	11.2	11.3
June, 2018	8.5	7.8	7.9
July, 2018	9.5	10.4	10.0
August, 2018	8.1	9.3	9.6
September, 2018	8.8	9.9	9.5
October, 2018	13.2	11.7	11.6
November, 2018	13.5	16.3	16.2
December, 2018	16.2	16.2	16.4
January, 2019	16.6	18.0	17.5
February, 2019	17.8	17.9	18.2
March, 2019	17.3	17.6	17.2
April, 2019	13.0	13.8	13.8
May, 2019	8.7	9.8	10.1
June, 2019	9.2	8.2	7.9
Root mean square error (RMSE)		1.1499	1.1282
Mean absolute deviation (MAD)		0.9667	0.9389

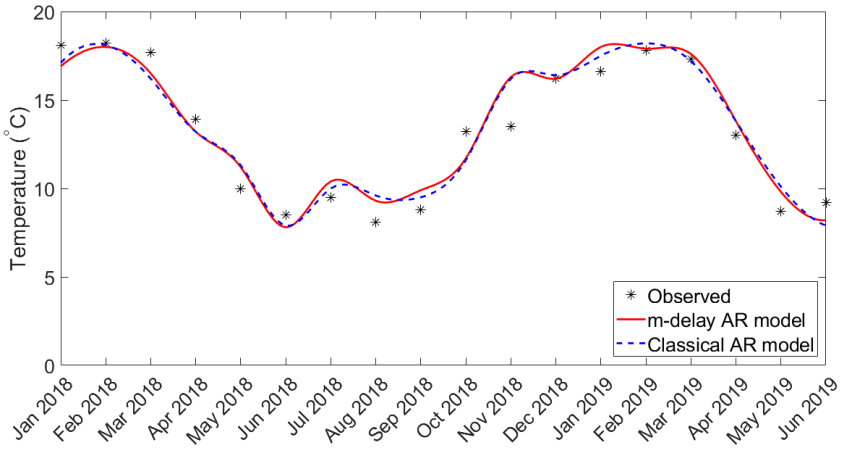


Figure 3.7: Scatter plot of observed data, and a dashed line (the MAR model) and a solid line (the AR model).

Table 3.12: Mann-Whitney U test result.

Method	Mann-Whitney U test	p-value
MAR - AR methods	162.000	1.000

minimum temperature in Perth obtained from the MAR model and the AR(9) model. Consequently, the m-delay AR model is an effective model for time series prediction because this method utilizes only two model parameters while the traditional AR order p $(AR(p))$ method requires p parameters.

3.4 Conclusion

A classical AR model with a large delay requires determining many coefficient parameters. This chapter considers a special case of the classical AR model, particularly when the delay is large. The m-delay AR model (MAR) using two parameters is proposed. We develop the least squares method for estimating two model parameters. According to simulations and case studies, there is no difference between the AR and the MAR methods. Since our proposed method only considers two model parameters, it would be beneficial when large delays are examined. We outline future developments and possible applications below.

- For parameter estimation, we only modify the least square method to approximate the coefficients of the m-delay AR model. There are several techniques to approximate these AR parameters, namely the maximum likelihood method, the Yule-Walker estimation, and the method of moment.

- Based on the simulation results in Section 3.4, the root mean square error is utilized. For future development, adding more information criteria can be useful to choose an optimal delay such as the Akaike information criteria (AIC), the Bayesian information criteria (BIC), the minimum de-

scription length (MDL) [139], the predictive least square (PLS), the predictive densities criteria (PDC), the sequentially normalized maximum likelihood (SNML) criteria, the final prediction error (FPE) [140] and the criteria autoregressive transfer function (CAT) [141].

- Our proposed model can be applied to historical financial data to estimate the volatility of the stochastic delay differential equation.

Chapter 4

Estimating Volatility of Stochastic Delay Differential Equation

This chapter attempts to estimate the parameters of the stochastic delay differential equation (SDDE) for matching the volatility obtained from the Monte Carlo simulations and the real-world volatility. The parameters of the SDDE are the drift term (λ) and the volatility (σ). We assume that the volatility is fixed in the first part, but the drift term (λ) parameter is unknown. Two parameters are unknown in the second part. The organization of this chapter is structured as follows. Section 4.1 displays the stochastic differential equation. The data visualization is shown in Sections 4.2. Section 4.3 and 4.4 demonstrate the model identification for one and two parameters together with the matching part results, respectively. Section 4.5 makes a summary of this work. Note that the work in this chapter has been published in [142] and [143].

4.1 Stochastic Differential Equation

A standard stochastic differential equation is given by [144]

$$dY(t) = a(Y(t), t)dt + b(Y(t), t)dw(t), \tag{4.1}$$

where $a(Y(t), t)$ and $b(Y(t), t)$ are drift and diffusion terms, respectively, $Y(t)$ denote the return of stock prices. In financial statistics, $b(Y(t), t)$ is termed the "volatility" [145]. We estimate the solution through discretization of SDE.

The widespread application of the SDE in financial economics has attracted numerous researchers to research and develop this model. The popular SDE models start with the Bachelier model, which defines the drift and diffusion terms as constants [146]. The general Bachelier model is described as [147]. Louis Bachelier was the first to initiate the study of continuous-time processes and introduce Brownian motion mathematically. The general Bachelier model is described as

$$dY(t) = \beta dt + \sigma dw(t), \tag{4.2}$$

where constants $\beta > 0$ and $\sigma > 0$ are called the drift and volatility, $w(t)$ is the standard Brownian motion (Wiener process) and the stock prices are $S(t) = e^{Y(t)}$.

The Ornstein-Uhlenbeck (OU) process was introduced as an improved model for physical Brownian motion, which incorporates the effect of friction [148]. The following SDE can define the OU process:

$$dY(t) = -\lambda Y(t)dt + \sigma dw(t), \tag{4.3}$$

where λ presents the rate of mean reversion, the drift coefficient here is a linear function of Y(t). For the non-linear drift term of SDE, the radial Ornstein-Uhlenbeck and the hyperbolic diffusion processes are outlined in [149]. The radial Ornstein-Uhlenbeck is defined as

$$dY(t) = (\lambda Y^{-1}(t) - Y(t))dt + \sigma dw(t). \tag{4.4}$$

The hyperbolic diffusion process has the following form:

$$dY(t) = -\lambda Y(t)\left(\frac{1}{\sqrt{1+Y^2(t)}}\right)dt + \sigma dw(t). \tag{4.5}$$

Loung and Dokuchaev [80] suggested modelling the continuous-time stock price process based on the general Ornstein-Uhlenbeck in Eq. (4.3) with add delay term. The authors confirm that the new delay term in the SDE provides a better result for matching the simulated price process volatility with the volatility from the historical data [150, 151].

In order to model the Ornstein-Uhlenbeck process on a computer, it is usual to discretize time and compute samples at discrete time steps of width Δt,

$$\begin{aligned} dY(t) &= \lambda(\mu - Y)dt + \sigma dw(t) \\ &= -\lambda(Y - \mu)dt + \sigma dw(t) \end{aligned}$$

if $\mu = 0$

$$\begin{aligned} dY(t) &= -\lambda Y(t)dt + \sigma dw(t) \\ Y(t) - Y(t-1) &= -\lambda Y(t-1)\Delta t + \sigma dw(t) \end{aligned}$$

Given a fixed time increment $\Delta t > 0$, one can easily simulate a trajectory of the Wiener process in the time interval $[t_0, T]$

$$w(\Delta t) = w(\Delta t) - w(0) \sim N(0, \Delta t) \sim \sqrt{\Delta t}N(0, 1),$$

and the same is also true for any other increment $w(t + \Delta t)$ - $w(t)$; i.e.,

$$w(t + \Delta t) - w(t) \sim N(0, \Delta t) \sim \sqrt{\Delta t}N(0, 1).$$

Let δ = Δt and $\xi = N(0, 1)$. Therefore,

$$Y(t) - Y(t-1) = -\lambda[Y(t-1) - Y(t-m)]\Delta t + \sigma dw(t)$$
$$Y(t) = Y(t-1) - \lambda[Y(t-1) - Y(t-m)]\delta + \sigma\sqrt{\delta}\xi(t)$$

Finally, the SDDE developed by Luong and Dokuchaev [80], is defined by

$$Y(t) = (1 - \lambda\delta)Y(t-1) + \lambda\delta Y(t-m) + \sigma\sqrt{\delta}\xi(t), \tag{4.6}$$

where $Y(t)$ are the returns of the stock price at certain time t, $\sigma > 0$ is the stock volatility, m presents a time delay, δ be sampling frequency, and $\xi(t_k)$ be independent and identically distributed random variables from the standard normal distribution.

Here, the drift parameter is defined as

$$a(Y(t), t) = -\lambda(Y(t) - Y(t-m)). \tag{4.7}$$

In their model, model parameters were turned via trial-and-error. To the best of our knowledge, there is no attempt to estimate two unknown parameters (λ, σ) of SDDE in Eq. (4.6). This motivated us to estimate two model parameters of the SDDE.

4.2 Data

The intraday closing prices, which are recorded with the frequency of every 5 minutes and 15 minutes, are extracted from the Thomson Reuters database in 2008 totally of 19,750 and 6,750 prices, respectively [152]. We selected some of the top stocks in the New York Stock Exchange (NYSE), including the International Business Machines Corporation (IBM), the Microsoft Corporation (MSFT), the Standard and Poor's 500 (S&P 500) and the Standard and Poor's

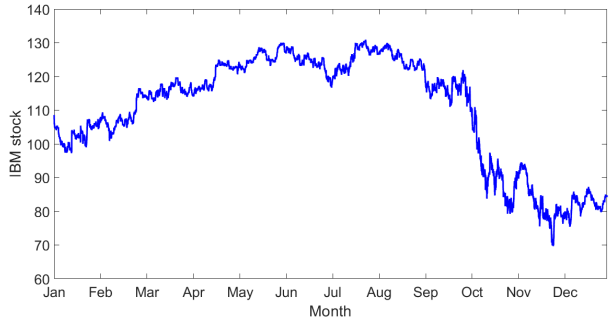
100 (S&P 100). The NYSE is an American stock exchange in the Financial District of Lower Manhattan in New York City, traded between 9.30 am. and 4.00 pm without lunch break on weekdays. The descriptive statistics in 2008, including mean, median, minimum, maximum and standard deviation of four stocks, are shown in Table 4.1.

Table 4.1: Descriptive statistical analysis results of four original stock prices.

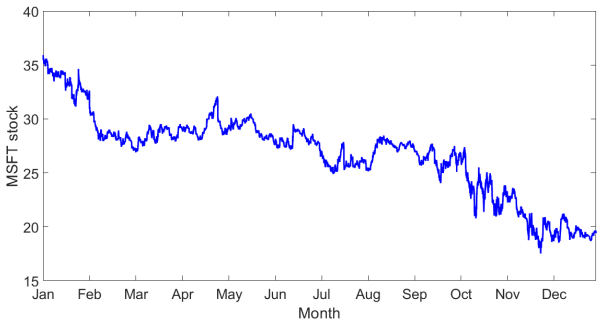
Index	Mean	Median	Minimum	Maximum	Standard Deviation
• **5 minutes**					
IBM	110.26	116.01	69.65	130.92	16.24
MSFT	26.74	27.67	17.52	35.92	3.81
S&P 500	1,223.07	1,291.69	743.79	1,459.38	190.01
S&P 100	568.59	596.80	362.62	681.23	80.60
• **15 minutes**					
IBM	110.26	116	69.65	130.92	16.24
MSFT	26.74	27.68	17.54	35.9	3.81
S&P 500	1,223.06	1,291.56	746.56	1,458.36	189.99
S&P 100	568.59	596.78	363.80	680.83	80.58

The descriptive statistics summaries about the sample and measures of the data. It consists of two primary measures: measures of central tendency (mean, median, mode) and measures of variability (minimum, maximum and standard deviation variables). As shown in Table 4.1 for 5 minutes data, the mean price of IBM is 110.26 with a median of 116.01 USD. The lowest, highest, and standard deviation prices are 69.65,130.92 and 16.24, respectively. The average MSFT stock price is 26.74, and the median is 27.67 USD for MSFT stock. The price range and the standard deviation are 18.40 and 3.81, respectively. The mean of S&P 500 stock and the median are 1,223.07 and 1,291.69 USD, respectively. The S&P 500 prices drooped to 743.79 USD and peaked at 1,459.38 USD. For S&P 100 stock, 568.59 USD is the average price, and 596.80 USD is the median price. The price range and the standard deviations are 318.61 and 80.60 USD, respectively.

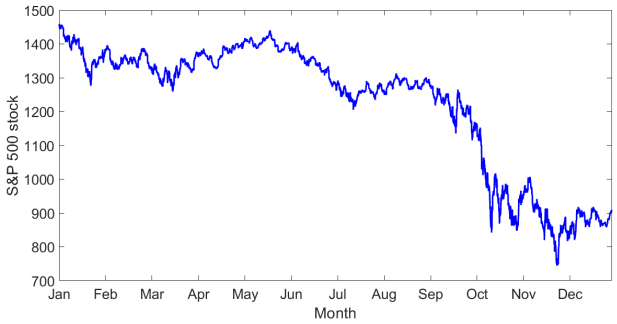
Time series plots of observed price every 5 minutes and 15 minutes time interval are displayed in Figures 4.1 - 4.2.



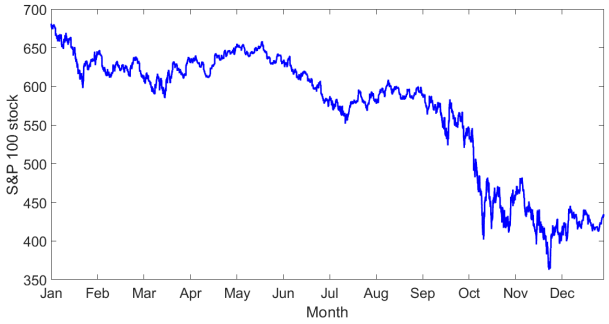
(a) International Business Machines Corporation (IBM)



(b) Microsoft Corporation (MSFT)

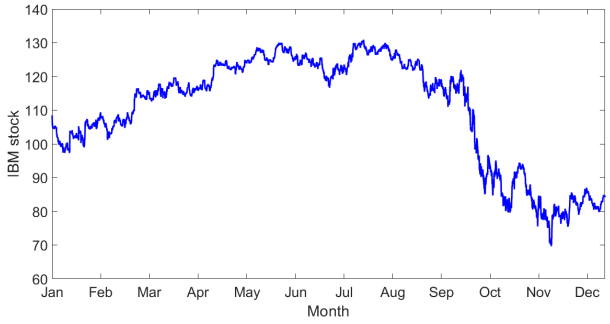


(c) Standard and Poor's 500 (S&P 500)

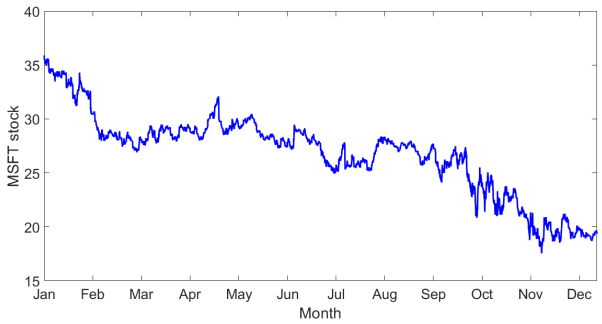


(d) Standard and Poor's 100 (S&P 100)

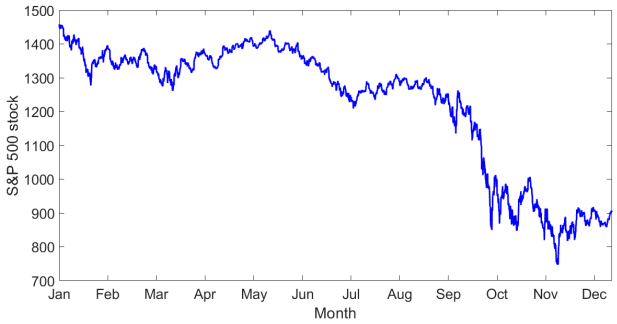
Figure 4.1: Four time series plot at 5 minutes stock prices in 2008



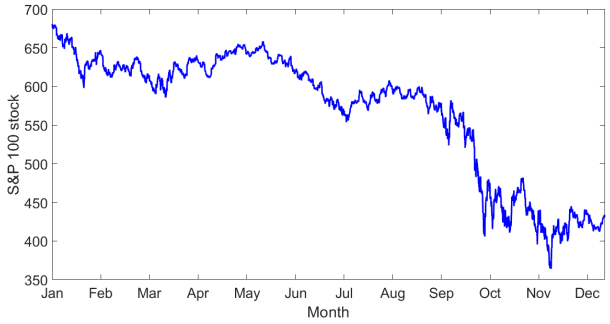
(a) International Business Machines Corporation (IBM)



(b) Microsoft Corporation (MSFT)



(c) Standard and Poor's 500 (S&P 500)



(d) Standard and Poor's 100 (S&P 100)

Figure 4.2: Four time series plot at 15 minutes stock prices in 2008

As shown from Figure 4.1 and Figure 4.2, we study the pattern of stock prices during the 2008 global financial crisis and the great recession. It can be noticed that all sample stock prices significant drop at the end of 2008 [153].

The number of observation per each month at 5 minutes and 15 minutes time interval is shown in Table 4.2.

Table 4.2: The number of observation at 5 minutes and 15 minutes of four original stock prices for each month in 2008.

Frequency	January	February	March	April	May	June
5 minutes	1,659	1,580	1,580	1,738	1,659	1,659
15 minutes	567	540	540	594	567	567
Frequency	July	August	September	October	November	December
5 minutes	1,659	1,659	1,659	1,817	1,422	1,659
15 minutes	567	567	567	621	486	567

4.3 Model Identification for One Parameter

In this section, the main aim is to match the estimated volatility for one unknown parameter (λ) of the SDDE in Eq. (4.6). To achieve the aim, we propose the m-delay Autoregressive coefficients (ARC) algorithm and compare it with the standard differential evolution algorithm for matching the estimated volatility from the Monte Carlo simulations and actual data set.

4.3.1 Research Methods

In this section, we present how to estimate the parameter of the drift term (λ) in the SDDE model Eq. (4.6). The stock price may be determined by

$$S(t_k) = e^{Y(t_k)}. \tag{4.8}$$

For notation simplicity, we let $Y_k = Y(t_k)$. As Eq. (4.6) is a linear autoregressive with delay m, we then obtain

$$Y_k = \epsilon_k + \phi_1 Y_{k-1} + \phi_m Y_{k-m}, \tag{4.9}$$

where ϕ_1 and ϕ_m are computed by

$$\begin{cases} \phi_1 &= \dfrac{\sum\limits_{t=m+1}^{n} Y_t Y_{t-1} \sum\limits_{t=m+1}^{n} Y_{t-m}^2 - \sum\limits_{t=m+1}^{n} Y_{t-1} Y_{t-m} \sum\limits_{t=m+1}^{n} Y_t Y_{t-m}}{\sum\limits_{t=m+1}^{n} Y_{t-1}^2 \sum\limits_{t=m+1}^{n} Y_{t-m}^2 - \left[\sum\limits_{t=m+1}^{n} Y_{t-1} Y_{t-m}\right]^2}, \\ \phi_m &= \dfrac{\sum\limits_{t=m+1}^{n} Y_{t-1}^2 \sum\limits_{t=m+1}^{n} Y_t Y_{t-m} - \sum\limits_{t=m+1}^{n} Y_t Y_{t-1} \sum\limits_{t=m+1}^{n} Y_{t-1} Y_{t-m}}{\sum\limits_{t=m+1}^{n} Y_{t-1}^2 \sum\limits_{t=m+1}^{n} Y_{t-m}^2 - \left[\sum\limits_{t=m+1}^{n} Y_{t-1} Y_{t-m}\right]^2}. \end{cases} \tag{4.10}$$

We call ϕ_1 and ϕ_m as the two m-delay AR coefficients which is based on the m-delay AR model [135].

AR characteristic equation is

$$1 - \phi_1 L - \phi_m L^m = 0. \tag{4.11}$$

Let $\{a_1, a_2, \ldots, a_m\}$ be the m roots of the characteristic polynomial. As one of its roots is 1, the series $\{Y_k\}_{k=1}^{\kappa}$ does not rely on stationarity condition. Hence, we define a_1 equal 1 and the others $\{a_2, a_3, \ldots, a_m\}$ are greater than 1. We then apply Eq. (4.6) to obtain the estimated volatility.

$$\hat{\sigma} = \sqrt{\hat{\sigma}^2} = \left[\frac{1}{\Delta t} \sum_{k=2}^{n} (\bar{Y}_m - Y_k)^2\right]^{\frac{1}{2}}, \tag{4.12}$$

where $\bar{Y}_m$ is the average of all sample return from the stock prices $S(t_k)$, $Y_k = \log S(t_k) - \log S(t_{k-1})$ and $\Delta t = (n-1)\delta$.

In order to demonstrate the effectiveness of the proposed model, the evaluation metrics , namely the root mean square error (RMSE), the mean absolute

error (MAE) and the mean absolute percentage error (MAPE) are employed. The definition of three evaluation metrics are as follows:

$$RMSE = \frac{1}{n}\sqrt{\sum_{k=1}^{n}(y(k) - \hat{y}(k))^2}, \tag{4.13}$$

$$MAE = \frac{\sum_{k=1}^{n}|y(k) - \hat{y}(k)|}{n}, \tag{4.14}$$

$$MAPE = \frac{\sum_{k=1}^{n}|y(k) - \hat{y}(k)|\,/y(k)}{n} \times 100, \tag{4.15}$$

where $y(k)$ and $\hat{y}$(k) denote observed data and predicted data, respectively.

The ranges of MAPE define the model performance as follows [154]:

- highly accurate if: MAPE $\leq$ 10 %
- good accurate if: 11% $\leq$ MAPE $\leq$ 20 %
- reasonable accurate if: 21 % $\leq$ MAPE $\leq$ 50 %

To match the estimated volatility obtained from the Monte Carlo simulations and the historical financial data, we propose the ARC algorithm as follows:

The ARC algorithm (Algorithm 4) begins with fixing the sample size, the delay, the sampling frequency, and the constant term. We then computed the monthly volatility of IBM, MSFT, S&P 500 and S&P 100 by Eq. (4.12). To estimate the volatility in the simulation part, we start with setting the volatility from historical data equal to the volatility of the Monte Carlo simulations. The initial return set is computed by setting the parameter of the drift term (λ) of Eq. (4.6) to zero. The two m-delay AR coefficients are calculated using Eq. (4.10) to obtain $(\hat{\lambda}^{(1)})$. We then apply $(\hat{\lambda}^{(1)})$ to generate the next return set by

Algorithm 4 Pseudocode for the ARC algorithm

Require: n : number of sample size, m : delay, δ : sampling frequency, TOL : constant term

Compute the volatility ($\sigma_{historical}$) at two different sampling frequencies σ_{5min} and σ_{15min} of IBM, MSFT, S&P 500 and S&P 100 by Eq. (4.12).
Set $\sigma = \sigma_{historical}$
Set $j = 1$
Initial return
for $t = 1$ to n **do**
  $y_t = \sigma\sqrt{\delta}\xi_t$
end for
Compute ϕ_1, ϕ_m from Eq. (4.10) to obtain $\hat{\lambda}_1$
while ($|\hat{\lambda}_j - \hat{\lambda}_{j-1}| \geq TOL$) **do**
  for $t = m + 1$ to n
    $y_t = (1 - \delta\hat{\lambda}_j)y_{t-1} + (\delta\hat{\lambda}_j)y_{t-m} + \sigma\sqrt{\delta}\xi_t$
  end for
  $j = j + 1$
  Compute ϕ_1, ϕ_m from Eq. (4.10) to obtain $\hat{\lambda}_j$
end while
Compute the estimated volatility Eq. (4.12)

Eq. (4.6) and the volatility is calculated using Eq. (4.12). The difference between two steps $\hat{\lambda}$ is computed. The process stops when the difference value is less than the constant term. Finally, the estimated volatility of the Monte Carlo simulations part is computed using the last set of returns.

The differential evolution (DE) algorithm is used to compare the ARC algorithm's performance. The DE algorithm is a competitive tool for identifying the model parameters. It was introduced by Storn and Price in 1996 [155]. The DE is a stochastic population-based metaheuristic algorithm that is easy to understand, simple to implement, has better performance and is reliable [101]. DE algorithm has been applied to many fields. This algorithm has successfully solved various engineering problems [156, 157], such as mechanical engineering design problems, chemical engineering design, and biomedical engineering design problems. In finance, the DE technique has been used to estimate trading strategies to maximise trading profit [158, 159]. The idea of the DE algorithm starts with prescribing the number of population (NP), the crossover

ratio (CR), the number of model parameters (D), the mutation ratio (F), the delay (m), the sampling frequency (δ), tolerance term (TOL), the data size (n), upper bound (U) and lower bound (L). The initial population is chosen randomly with uniform distribution in the user-defined bounds. The DE optimisation process has three operators: mutation, crossover and selection. Trial vectors are generated using mutation and crossover operators. The selection operator is applied to select which vectors are survived to be a member of the next generation. The cycle of the three operators is repeated until the predetermined convergence criterion is satisfied.

The pseudocodeto compute the volatility obtained from the DE algorithm is displayed in Algorithm 5.

The Algorithm 5 starts with defining the number of population size (NP), the crossover ratio (CR), the mutation ratio (F), delay (m), the sampling frequency (δ), constant term (TOL) is 0.01, the upper bound (U) and the lower bound (L). The first step of simple DE starts with the initialization stage. An initial population is generated using a uniform distribution on the interval [0,1], where λ^L and λ^U are the lower and upper bounds for the decision parameter, respectively. In the mutation process, three individuals $(\lambda_{r1}, \lambda_{r2}, \lambda_{r3})$ are selected in the population set of NP elements, which $r1 \neq r2 \neq r3 \neq j$. F is a user-defined constant known as the mutation factor, $F \in [0, 1]$. We then apply all values to obtain the mutant vectors (v_j).

The trial vectors (u_j) are generated by mixing the parameters of the target vectors (λ_j) with the mutant vectors (v_j) according to a selected crossover probability (CR). The selection scheme is applied in the DE algorithm for the next step. The historical data are imported to compute the fitness of the target vectors and the fitness of trial vectors. The best solution is chosen by comparing the fitness of the trial vectors and the corresponding target vectors. We then calculate the predicted stock prices $(\hat{y})$ of SDDE function using Eq. (4.6). The error between actual stock prices and predicted stocked prices are calculated.

Algorithm 5 Pseudocode for the classical differential evolution for one model parameter

Require: NP : number of population size, CR : crossover probability, F : mutation factor,m : delay, δ : sampling frequency, TOL : constant term, U : upper bound, L : lower bound

Initialisation

Generate the initial population
for $j = 1$ to NP **do**
  $\lambda_j = \lambda^L + rand \cdot (\lambda^U - \lambda^L), \quad j = 1, 2, \ldots, NP$
end for
for $j = 1$ to NP **do**
  randomly select $r_1, r_2, r_3 \in 1, 2, \ldots, NP$ where $r_1 \neq r_2 \neq r_3 \neq j$
  Mutation
  $v_j = \lambda_{r1} + F \cdot (\lambda_{r2} - \lambda_{r3})$
  Crossover
  if $rand(0, 1) < CR$ **then**
    $u_j = v_j$
  else
    $u_j = \lambda_j$
  end if
  Selection
  if $f(u_j) < f(\lambda_j)$ **then**
    $\hat{\lambda}_j = u_j$
  else
    $\hat{\lambda}_j = \lambda_j$
  end if
end for
Import real stock prices
while $(MAPE > TOL)$ **do**
  for $j = 1$ to NP
    for $t = m + 1$ to NP
    $\hat{y}_t = (1 - \delta\hat{\lambda}_j)y_{t-1} + (\delta\hat{\lambda}_j)y_{t-m} + \sigma\sqrt{\delta}noise(0, 1)$
    $e_t = \left|\frac{y_t - \hat{y}_t}{y_t}\right|$
    end for
  end for
$MAPE = \frac{1}{NP-m}\sum_{t=m+1}^{NP} e_t \times 100$
end while
Compute the estimated volatility Eq. (4.12)

The mean absolute percentage error (MAPE) is applied to seek an optimal parameter $(\hat{\lambda})$. The optimization process is stopped whenever the MAPE is less than a constant term (TOL). Finally, the last set of data is applied in Eq. (4.12) to obtain the estimated volatility.

For choosing the number of population sizes (NP), Storn and Price [155] suggested a reasonable choice for NP between 5D and 10D (D represents the problem size). In this thesis, we choose 10D as the number of populations. Therefore, the number of populations used in Section 4.3 and Section 5.4.1 (Algorithm 7) is equal to 10 because there is one unknown parameter for estimation. For two unknown parameters determined based on Section 4.4, the number of populations equals 20. Generally, F and CR fall within the range of [0, 2] and [0, 1], respectively. The sizes $F = 0.2$, $CR = 0.6$ have been used in [160–162]. Consequently, we chose a mutation ratio $(F) = 0.2$ and a crossover rate $(CR) = 0.6$ for the entire experiment.

4.3.2 Numerical Results

Monte Carlo Simulations of Stock Volatility

The estimated Volatility $(\hat{\sigma})$ in Eq. (4.12) is obtained by the Monte Carlo simulations in Eq. (4.6) using 22 trading days in a month, 6.5 trading-hours on a regular day, 12 samples per hour (5-minute interval) and 4 samples per hour (15-minute interval). The value of parameters used in the simulation are as follows:

- $\delta = 1/(22 \times 6.5 \times 12)$ for 5 minutes interval
- $\delta = 1/(22 \times 6.5 \times 4)$ for 15 minutes interval
- $m = 3, 10, 20, 50$ and $\lambda \in [-2000, 0) \cup (0, 20000]$
- $\sigma = 0.3$

We obtain the estimated volatility $(\hat{\sigma})$ with its mean and standard deviation (SD) for 10,000 iterations as shown in Table 4.3.

Table 4.3: Mean and standard deviation of estimated volatility $(\hat{\sigma})$ for each triplet (λ, δ, m)

λ	δ	$m = 3$		$m = 10$		$m = 20$		$m = 50$	
		Mean	SD	Mean	SD	Mean	SD	Mean	SD
-1.716	5-min	0.2999	0.0051	0.2998	0.0051	0.2999	0.0052	0.2999	0.0051
-0.572	15-min	0.2999	0.0089	0.2999	0.0089	0.3000	0.0089	0.3000	0.0089
-4.290	5-min	0.2998	0.0051	0.2998	0.0051	0.2999	0.0052	0.2999	0.0052
-1.430	15-min	0.3005	0.0089	0.3004	0.0087	0.3005	0.0089	0.3005	0.0089
-17.160	5-min	0.2999	0.0051	0.2999	0.0052	0.3002	0.0051	0.3000	0.0053
-5.720	15-min	0.3026	0.0089	0.3028	0.0090	0.3033	0.0090	0.3059	0.0095
1.716	5-min	0.2998	0.0051	0.2998	0.0051	0.2998	0.0051	0.2999	0.0051
0.572	15-min	0.2993	0.0090	0.2994	0.0088	0.2993	0.0088	0.2994	0.0089
4.290	5-min	0.2998	0.0051	0.2999	0.0051	0.2999	0.0051	0.2999	0.0050
1.430	15-min	0.2988	0.0089	0.2989	0.0088	0.2989	0.0088	0.2990	0.0088
17.160	5-min	0.3000	0.0051	0.3001	0.0052	0.3001	0.0051	0.3004	0.0052
5.720	15min	0.2967	0.0089	0.2971	0.0088	0.2974	0.0089	0.2985	0.0089

As shown from Table 4.3, it can be noted that

- When $\lambda > 0$, the estimated volatility decreases as the sampling frequency decreases.
- When $\lambda < 0$, decreasing sampling frequency leads to increasing of the estimated volatility.

Case Studying Volatility

We now demonstrate the volatility from the high-frequency data of the IBM, MSFT, S&P 500 and S&P 100. These data were traded between 09.30 am and 04.00 pm from Thomson Reuters database in 2008 [152].

The compromised procedure to compute the volatility is represented in Figure 4.3

The flow chart starts with importing all stocks from the Thomson Reuters database. We then clean up errors and fill any missing data using a splines interpolation method. Next, we take a logarithm to obtain the return and the

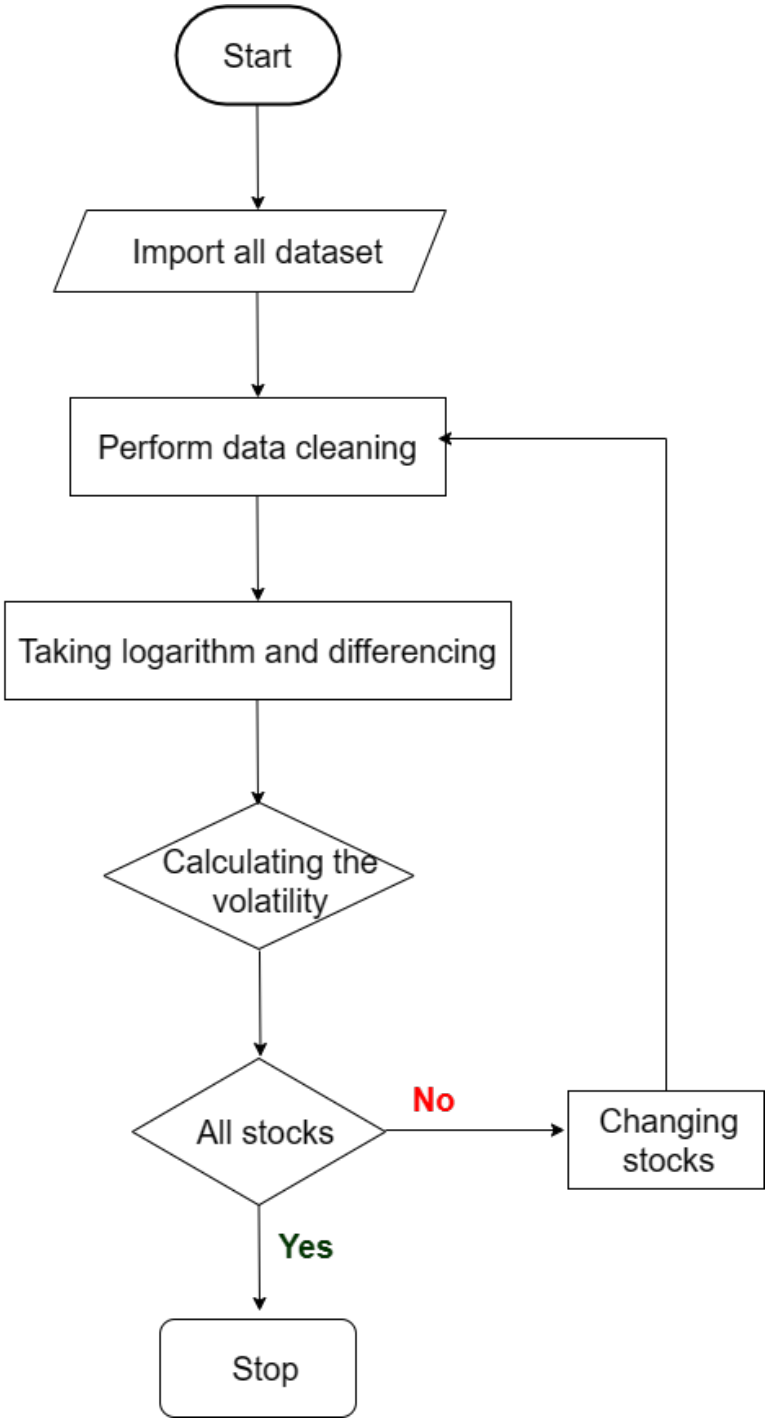


Figure 4.3: Flow chart how to analyze data sets.

difference. The volatility is computed in the next step. This process is stopped when the volatility is computed for all stocks.

The volatility for the IBM, MSFT, S&P 500 and S&P 100 at two different sampling frequencies in 2008 are displayed in Table 4.4.

Table 4.4: Actual volatility of four stock indexes with two different sampling frequencies in 2008

Month	Data	Stock			
		IBM	MSFT	S&P 500	S&P 100
January	5-min	0.128238	0.114330	0.071822	0.070590
	15-min	0.111404	0.100703	0.069563	0.067839
February	5-min	0.076853	0.076580	0.050601	0.049385
	15-min	0.075787	0.076787	0.049258	0.048145
March	5-min	0.070163	0.081282	0.061975	0.060435
	15-min	0.070075	0.083488	0.063043	0.061576
April	5-min	0.059146	0.087885	0.040261	0.039062
	15-min	0.056966	0.081779	0.038989	0.038107
May	5-min	0.050221	0.069760	0.032422	0.032110
	15-min	0.045363	0.063007	0.031329	0.031074
June	5-min	0.062038	0.079216	0.045904	0.045730
	15-min	0.058700	0.078077	0.046096	0.045950
July	5-min	0.076856	0.104967	0.054959	0.055746
	15-min	0.073569	0.105926	0.054173	0.055108
August	5-min	0.058617	0.076249	0.040903	0.042842
	15-min	0.052804	0.071691	0.040682	0.042794
September	5-min	0.148514	0.137254	0.101656	0.105564
	15-min	0.138231	0.128023	0.101457	0.104143
October	5-min	0.258056	0.251875	0.191253	0.188473
	15-min	0.235271	0.260465	0.198555	0.195551
November	5-min	0.175672	0.171256	0.144492	0.142027
	15-min	0.176757	0.168367	0.147318	0.146257
December	5-min	0.121681	0.132251	0.102008	0.097710
	15-min	0.098087	0.111982	0.081194	0.077879

4.3.3 Matching the Volatility

Using the ARC Algorithm, the estimated volatility and the actual volatility of each stock (IBM, MSFT, S&P 500 and S&P 100) with two sampling frequencies at 5 minutes and 15 minutes time intervals are shown in Tables 4.5 - 4.8.

Table 4.5: The estimated volatility obtained from the ARC algorithm at each sampling frequency of IBM stock with four different delays in 2008

.

Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.128238	0.131111	0.129565	0.129179	0.128859
	15-min	0.111404	0.108429	0.113126	0.112521	0.111389
Feb	5-min	0.076853	0.080075	0.081295	0.074475	0.077382
	15-min	0.075787	0.080538	0.078856	0.074063	0.075016
Mar	5-min	0.070163	0.072932	0.071561	0.069850	0.070021
	15-min	0.070075	0.077505	0.072946	0.071310	0.070044
Apr	5-min	0.059146	0.062684	0.060983	0.060286	0.059933
	15-min	0.056966	0.059282	0.059059	0.055561	0.056249
May	5-min	0.050221	0.051809	0.051648	0.050058	0.050264
	15-min	0.045363	0.048115	0.047032	0.046082	0.045943
Jun	5-min	0.062038	0.067260	0.063685	0.062104	0.062053
	15-min	0.058700	0.064242	0.060154	0.057161	0.058913
Jul	5-min	0.076856	0.080862	0.078190	0.077318	0.076888
	15-min	0.073569	0.077617	0.076374	0.074069	0.073213
Aug	5-min	0.058617	0.062752	0.059873	0.059973	0.058260
	15-min	0.052804	0.055270	0.054887	0.053794	0.052518
Sep	5-min	0.148514	0.150890	0.150781	0.150321	0.149687
	15-min	0.138231	0.142430	0.142361	0.140593	0.138394
Oct	5-min	0.258056	0.263987	0.261178	0.252206	0.259156
	15-min	0.235271	0.252664	0.229766	0.233416	0.235047
Nov	5-min	0.175672	0.182943	0.170993	0.176974	0.174726
	15-min	0.176757	0.167162	0.181122	0.180348	0.177175
Dec	5-min	0.121681	0.124719	0.123943	0.119639	0.121554
	15-min	0.098087	0.105139	0.095712	0.096469	0.098808

Table 4.6: The estimated volatility obtained from the ARC algorithm at each sampling frequency of MSFT stock with four different delays in 2008

.

Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.114330	0.120710	0.118066	0.116465	0.115147
	15-min	0.100703	0.107346	0.104115	0.101948	0.100253
Feb	5-min	0.076580	0.080133	0.079551	0.077411	0.076622
	15-min	0.076787	0.091659	0.080123	0.075353	0.076738
Mar	5-min	0.081282	0.085889	0.079921	0.082993	0.081079
	15-min	0.083489	0.089207	0.086797	0.081199	0.083016
Apr	5-min	0.087885	0.090029	0.089167	0.088229	0.087684
	15-min	0.081779	0.087890	0.085526	0.084595	0.081464
May	5-min	0.069760	0.072065	0.071534	0.070488	0.069721
	15-min	0.063008	0.073474	0.066920	0.064605	0.063650
Jun	5-min	0.079216	0.084214	0.083314	0.078727	0.079318
	15-min	0.078077	0.085512	0.082536	0.075281	0.077459
Jul	5-min	0.104967	0.109782	0.108305	0.107205	0.104367
	15-min	0.105926	0.111484	0.108232	0.106219	0.105661
Aug	5-min	0.076249	0.080315	0.079202	0.077964	0.076533
	15-min	0.071691	0.078712	0.074466	0.070298	0.071322
Sep	5-min	0.137254	0.140612	0.138302	0.136621	0.138098
	15-min	0.128023	0.136635	0.131371	0.130904	0.128997
Oct	5-min	0.251875	0.246393	0.249783	0.252092	0.251800
	15-min	0.260465	0.268659	0.267803	0.257771	0.260186
Nov	5-min	0.171256	0.173374	0.172964	0.170524	0.171154
	15-min	0.168367	0.160969	0.173211	0.169196	0.168990
Dec	5-min	0.132252	0.139208	0.135359	0.133803	0.131968
	15-min	0.111982	0.118533	0.117688	0.112505	0.111379

Table 4.7: The estimated volatility obtained from the ARC algorithm at each sampling frequency of S&P 500 stock with four different delays in 2008

.

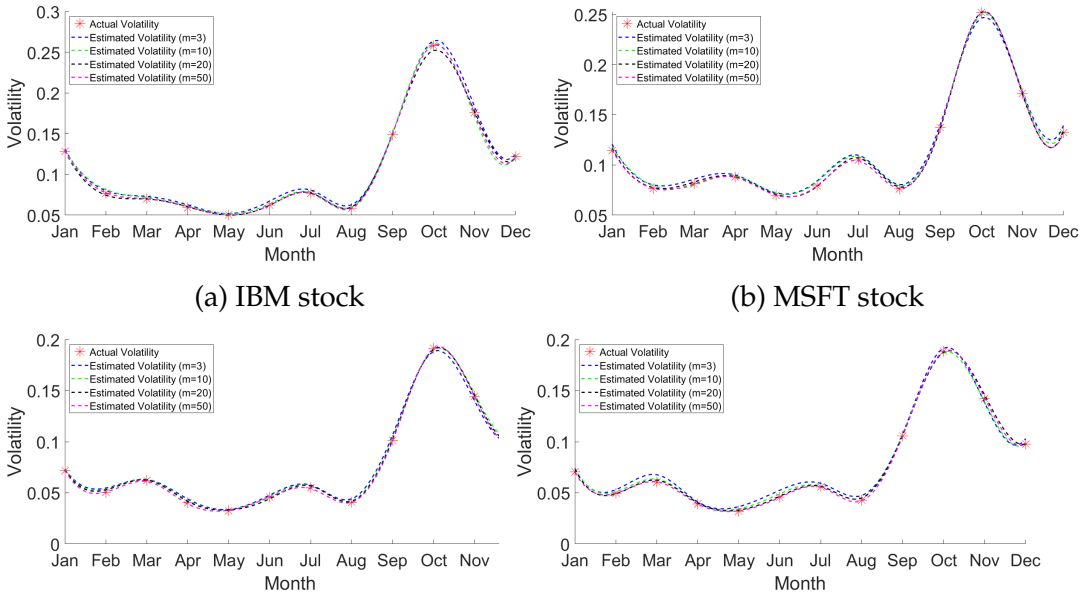
Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.071822	0.073439	0.073288	0.072551	0.072058
	15-min	0.069563	0.078433	0.077231	0.072319	0.068499
Feb	5-min	0.050601	0.054654	0.053574	0.052854	0.050754
	15-min	0.049258	0.054913	0.052228	0.048052	0.049474
Mar	5-min	0.061976	0.063146	0.063050	0.062586	0.061181
	15-min	0.063043	0.080765	0.076418	0.065197	0.063799
Apr	5-min	0.040261	0.043961	0.042338	0.041620	0.039222
	15-min	0.038989	0.042411	0.040339	0.039282	0.039099
May	5-min	0.032422	0.033721	0.033494	0.033198	0.033012
	15-min	0.031329	0.034780	0.034023	0.032934	0.031955
Jun	5-min	0.045904	0.47792	0.046439	0.044256	0.045913
	15-min	0.046096	0.049677	0.047986	0.045045	0.046363
Jul	5-min	0.054960	0.057626	0.057089	0.056876	0.054206
	15-min	0.054173	0.059272	0.058179	0.052458	0.054894
Aug	5-min	0.040903	0.044130	0.042311	0.041671	0.040758
	15-min	0.040682	0.048047	0.044737	0.041774	0.040670
Sep	5-min	0.101656	0.106742	0.103841	0.102507	0.101053
	15-min	0.101457	0.105499	0.103348	0.102332	0.101709
Oct	5-min	0.191253	0.188387	0.190046	0.190584	0.191385
	15-min	0.198555	0.201774	0.199449	0.199300	0.198109
Nov	5-min	0.144492	0.139871	0.147037	0.145167	0.144179
	15-min	0.147319	0.150720	0.149076	0.148845	0.143339
Dec	5-min	0.102008	0.105906	0.103528	0.103493	0.102275
	15-min	0.081194	0.088884	0.077286	0.079781	0.081188

Table 4.8: The estimated volatility obtained from the ARC algorithm at each sampling frequency of S&P 100 stock with four different delays in 2008

.

Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.070590	0.074387	0.073980	0.073126	0.069935
	15-min	0.067839	0.077363	0.072550	0.068619	0.067222
Feb	5-min	0.049385	0.053102	0.051048	0.048980	0.049559
	15-min	0.048145	0.054077	0.050828	0.049135	0.049063
Mar	5-min	0.060436	0.067768	0.063437	0.062435	0.060885
	15-min	0.061576	0.072895	0.065535	0.062738	0.062018
Apr	5-min	0.039062	0.041113	0.040629	0.040200	0.039174
	15-min	0.038107	0.043842	0.040281	0.037707	0.038539
May	5-min	0.032110	0.036498	0.034326	0.033117	0.032426
	15-min	0.031074	0.036509	0.034318	0.030385	0.031410
Jun	5-min	0.045730	0.052540	0.049090	0.046614	0.045474
	15-min	0.045950	0.059899	0.048014	0.044822	0.045059
Jul	5-min	0.055746	0.059336	0.056797	0.056433	0.055536
	15-min	0.055108	0.058433	0.057463	0.056922	0.055661
Aug	5-min	0.042842	0.047380	0.045036	0.044949	0.042122
	15-min	0.042794	0.049124	0.047512	0.040176	0.042465
Sep	5-min	0.105564	0.107561	0.107263	0.106174	0.105472
	15-min	0.104144	0.108984	0.107181	0.105062	0.104503
Oct	5-min	0.188473	0.191560	0.186986	0.187632	0.188488
	15-min	0.195552	0.202384	0.199650	0.198137	0.196001
Nov	5-min	0.142027	0.138092	0.138948	0.145661	0.142942
	15-min	0.146258	0.158020	0.149901	0.143770	0.146082
Dec	5-min	0.097710	0.102960	0.098505	0.097654	0.097661
	15-min	0.077879	0.090654	0.081065	0.078752	0.077648

Using the numerical results obtained from Tables 4.5 - 4.8, we plotted the actual volatility and its estimated volatility obtained from the simulation part with four different delays at 5 minutes and 15 minutes time intervals in 2008, as shown in Figures 4.4 - 4.5.



(a) IBM stock

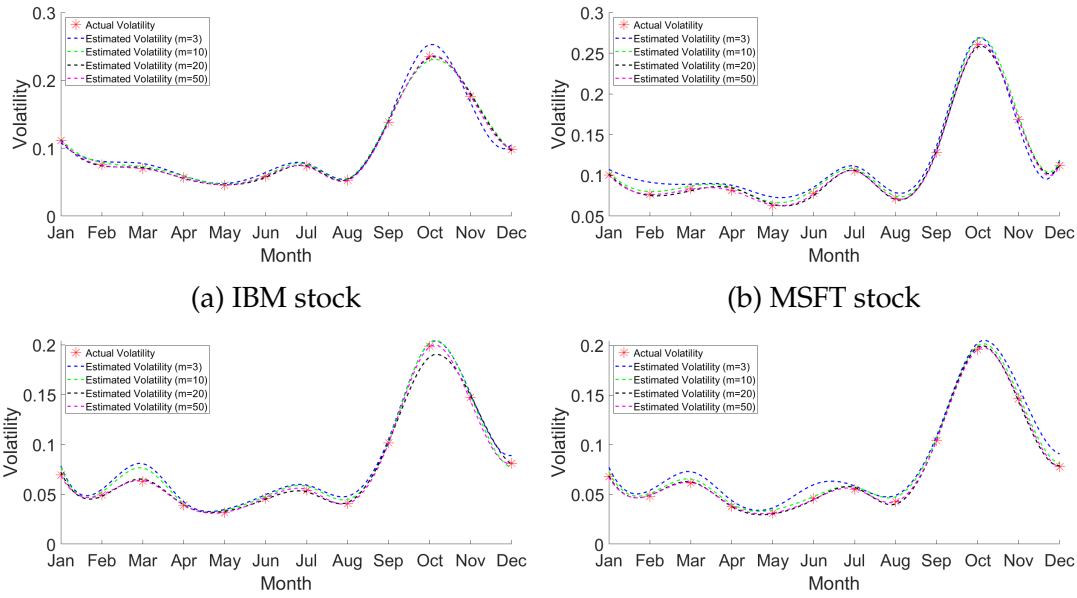
(b) MSFT stock

(c) S&P 500 stock

(d) S&P 100 stock

Figure 4.4: Actual volatility of four stocks (asterisk mark) and its estimates provided by Monte Carlo simulations of the ARC model with four delays at 5 minutes time interval.

Figure 4.4 compares the actual volatility and estimated volatility obtained from the ARC algorithm with four different delays at 5 minutes time intervals in 2008. It is seen that the estimated volatility for all stocks quite fits the actual volatility. Furthermore, the estimated volatility is better with a large delay than a slight delay for all four stocks. The estimated volatility at 15 minutes time interval is illustrated in Figure 4.5. It can be noticed that the gap between the actual and its estimated volatility is larger than the results obtained from 5 minutes, particularly in October, due to the volatility reach to a peak this month. The estimated volatility becomes more accurate when the delay is increased.



(a) IBM stock

(b) MSFT stock

(c) S&P 500 stock

(d) S&P 100 stock

Figure 4.5: Actual volatility of four stocks (asterisk mark) and its estimates provided by Monte Carlo simulations of the ARC model with four delays at 15 minutes time interval.

We then present the DE algorithm for parameter estimation and compare the estimated volatility with our proposed algorithm. For the DE algorithm, we first estimate the parameter of the drift term using the classical DE algorithm. Then, the estimated drift parameter is utilised in Eq. (4.6) and calculates the estimated volatility. The volatility and the actual volatility of each stock (IBM, MSFT, S&P 500 and S&P 100) at 5 and 15 minutes are shown in Tables 4.9 - 4.12.

The comparison of the estimated volatility obtained from the DE algorithm and its actual volatility of each stock (IBM, MSFT, S&P 500 and S&P 100) with four delays at 5 and 15 minutes time interval are shown in Figures 4.6 - 4.7.

Table 4.9: The estimated volatility obtained from the DE algorithm at each sampling frequency of IBM stock with four different delays in 2008

.

Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.128238	0.119657	0.124191	0.124949	0.127253
	15-min	0.111404	0.105038	0.107051	0.107979	0.108939
Feb	5-min	0.076853	0.065082	0.071724	0.073882	0.074111
	15-min	0.075787	0.062718	0.065644	0.068260	0.070996
Mar	5-min	0.070163	0.075156	0.074016	0.073630	0.073510
	15-min	0.070075	0.058058	0.059353	0.064967	0.068159
Apr	5-min	0.059146	0.055986	0.057970	0.057669	0.057890
	15-min	0.059966	0.051570	0.052286	0.052904	0.053772
May	5-min	0.050221	0.046642	0.048957	0.049264	0.049316
	15-min	0.045363	0.041697	0.041760	0.042055	0.042308
Jun	5-min	0.062038	0.058017	0.058713	0.059980	0.061297
	15-min	0.058700	0.049068	0.051974	0.054531	0.056472
Jul	5-min	0.076856	0.070758	0.073988	0.074316	0.074995
	15-min	0.073569	0.067920	0.068428	0.069239	0.070483
Aug	5-min	0.058617	0.053744	0.054281	0.057201	0.057593
	15-min	0.052804	0.044885	0.047454	0.047613	0.049980
Sep	5-min	0.148514	0.139200	0.144234	0.146532	0.146862
	15-min	0.138231	0.129209	0.131454	0.133337	0.134881
Oct	5-min	0.258056	0.239978	0.244892	0.250916	0.255460
	15-min	0.235271	0.210368	0.218745	0.226247	0.229019
Nov	5-min	0.175672	0.165518	0.170610	0.171879	0.172907
	15-min	0.176757	0.192479	0.189313	0.184587	0.181497
Dec	5-min	0.121681	0.117214	0.119057	0.119816	0.120027
	15-min	0.098087	0.115524	0.114389	0.109043	0.100885

Table 4.10: The estimated volatility obtained from the DE algorithm at each sampling frequency of MSFT stock with four different delays in 2008

Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.114330	0.104255	0.106272	0.111968	0.112793
	15-min	0.100703	0.089566	0.092589	0.095037	0.096466
Feb	5-min	0.076580	0.070891	0.070952	0.073941	0.074076
	15-min	0.076787	0.093114	0.087099	0.085662	0.082189
Mar	5-min	0.081282	0.077821	0.079137	0.079945	0.080389
	15-min	0.083489	0.064409	0.071256	0.073095	0.074307
Apr	5-min	0.087885	0.081894	0.085430	0.086813	0.087387
	15-min	0.081779	0.074551	0.075185	0.076580	0.077014
May	5-min	0.069760	0.065622	0.066902	0.067546	0.068315
	15-min	0.063008	0.050334	0.057647	0.058245	0.059699
Jun	5-min	0.079216	0.073967	0.074242	0.077882	0.078451
	15-min	0.078077	0.061910	0.063765	0.066593	0.073734
Jul	5-min	0.104967	0.098744	0.100007	0.100273	0.102197
	15-min	0.105926	0.095941	0.095977	0.101652	0.102463
Aug	5-min	0.076249	0.072981	0.073524	0.073739	0.075068
	15-min	0.071691	0.060714	0.063170	0.066195	0.067089
Sep	5-min	0.137254	0.130426	0.133169	0.135560	0.136988
	15-min	0.128023	0.118407	0.121234	0.122544	0.125413
Oct	5-min	0.251875	0.246007	0.249582	0.250159	0.250432
	15-min	0.260465	0.241803	0.244454	0.252475	0.257118
Nov	5-min	0.171256	0.165104	0.168016	0.170272	0.170382
	15-min	0.168367	0.160232	0.162285	0.164363	0.164989
Dec	5-min	0.132252	0.122661	0.128500	0.129048	0.131492
	15-min	0.111982	0.120473	0.119138	0.116320	0.113292

Table 4.11: The estimated volatility obtained from the DE algorithm at each sampling frequency of S&P 500 stock with four different delays in 2008

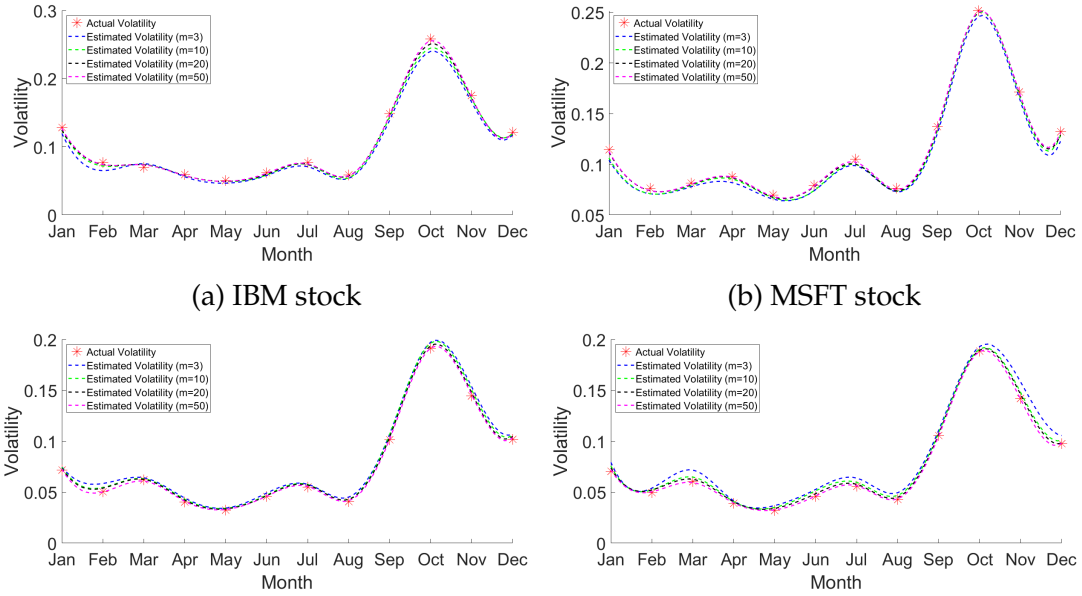
.

Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.071822	0.074850	0.074573	0.073508	0.072849
	15-min	0.069563	0.085551	0.078416	0.076930	0.075155
Feb	5-min	0.050601	0.058551	0.054652	0.054002	0.050264
	15-min	0.049258	0.055888	0.052312	0.051170	0.048196
Mar	5-min	0.061976	0.064171	0.063241	0.062713	0.060897
	15-min	0.063043	0.096521	0.086828	0.077756	0.061037
Apr	5-min	0.040261	0.044271	0.043039	0.042125	0.041080
	15-min	0.038989	0.043799	0.041611	0.040351	0.039328
May	5-min	0.032422	0.034438	0.034168	0.033607	0.033045
	15-min	0.031329	0.036148	0.034392	0.033666	0.030108
Jun	5-min	0.045904	0.049216	0.047276	0.046579	0.046157
	15-min	0.046096	0.066056	0.059790	0.054957	0.049789
Jul	5-min	0.054960	0.057797	0.057562	0.057075	0.055636
	15-min	0.054173	0.068994	0.064312	0.059102	0.052745
Aug	5-min	0.040903	0.045155	0.043128	0.042927	0.041653
	15-min	0.040682	0.048383	0.045401	0.043415	0.041035
Sep	5-min	0.101656	0.107222	0.106714	0.104237	0.102208
	15-min	0.101457	0.112052	0.107089	0.105261	0.104008
Oct	5-min	0.191253	0.197290	0.196454	0.193776	0.191517
	15-min	0.198555	0.224755	0.207418	0.204282	0.202066
Nov	5-min	0.144492	0.153618	0.150557	0.147613	0.145617
	15-min	0.147319	0.155982	0.152046	0.150149	0.145699
Dec	5-min	0.102008	0.107388	0.105864	0.104753	0.103223
	15-min	0.081194	0.104013	0.103989	0.099683	0.094588

Table 4.12: The estimated volatility obtained from the DE algorithm at each sampling frequency of S&P 100 stock with four different delays in 2008

.

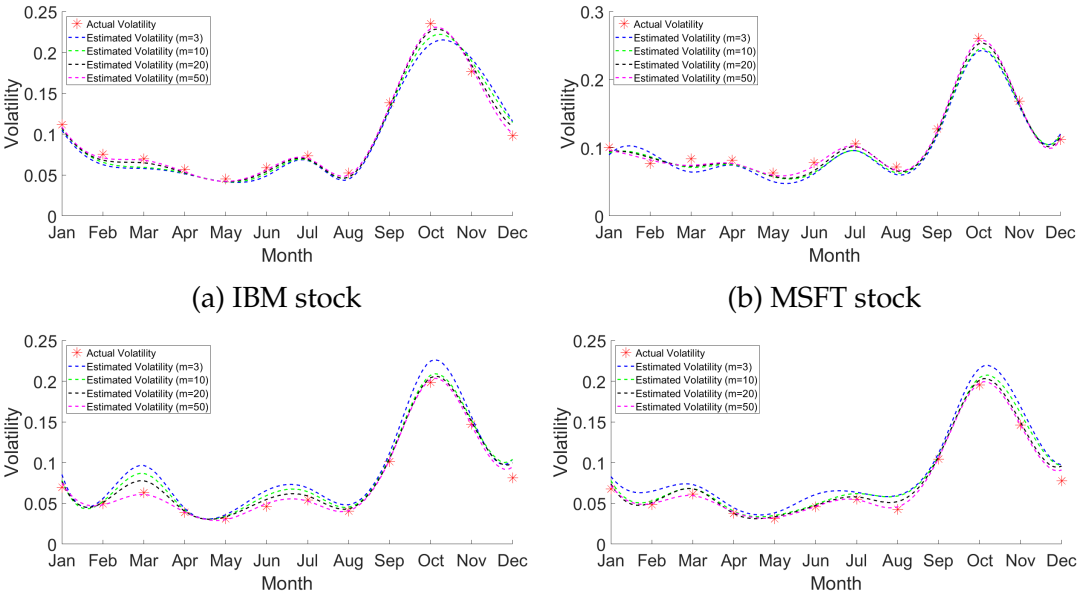
Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.070590	0.079385	0.075823	0.073898	0.072213
	15-min	0.067839	0.082973	0.077999	0.074014	0.072144
Feb	5-min	0.049385	0.053968	0.052226	0.052208	0.050437
	15-min	0.048145	0.064909	0.053113	0.051059	0.049869
Mar	5-min	0.060436	0.071740	0.064882	0.062712	0.059523
	15-min	0.061576	0.073130	0.068053	0.067917	0.059571
Apr	5-min	0.039062	0.041455	0.040731	0.040493	0.039463
	15-min	0.038107	0.044670	0.040425	0.037667	0.039393
May	5-min	0.032110	0.036923	0.035112	0.034193	0.032663
	15-min	0.031074	0.038387	0.035321	0.033731	0.032476
Jun	5-min	0.045730	0.052853	0.051430	0.048607	0.046298
	15-min	0.045950	0.061016	0.048853	0.047506	0.046290
Jul	5-min	0.055746	0.063853	0.059658	0.057757	0.056357
	15-min	0.055108	0.063631	0.062088	0.057989	0.054614
Aug	5-min	0.042842	0.049493	0.046917	0.045025	0.043504
	15-min	0.042794	0.059914	0.059057	0.052232	0.045782
Sep	5-min	0.105564	0.110800	0.108382	0.106969	0.105169
	15-min	0.104144	0.111829	0.108695	0.107634	0.106130
Oct	5-min	0.188473	0.193059	0.190473	0.189568	0.188130
	15-min	0.195552	0.215987	0.204513	0.201451	0.197871
Nov	5-min	0.142027	0.158215	0.147940	0.145961	0.141178
	15-min	0.146258	0.168589	0.160377	0.149581	0.145843
Dec	5-min	0.097710	0.105548	0.101042	0.098260	0.098094
	15-min	0.077879	0.097275	0.096594	0.095704	0.090943



(a) IBM stock (b) MSFT stock

(c) S&P 500 stock (d) S&P 100 stock

Figure 4.6: Actual volatility of four stocks (asterisk mark) and its estimates provided by Monte Carlo simulations of the DE model with four delays at 5 minutes.



(a) IBM stock (b) MSFT stock

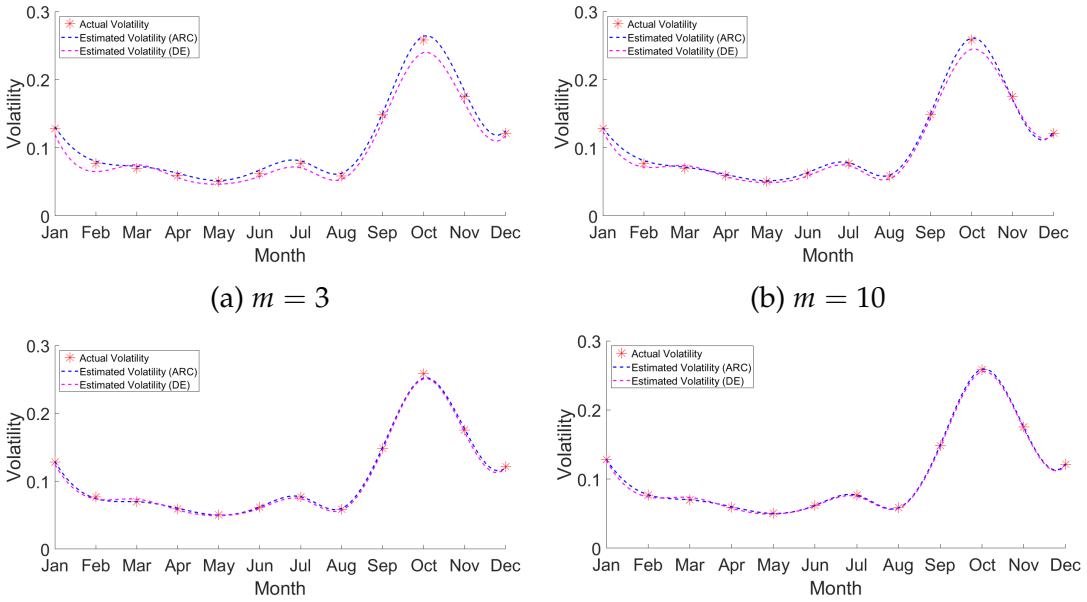
(c) S&P 500 stock (d) S&P 100 stock

Figure 4.7: Actual volatility of four stocks (asterisk mark) and its estimates provided by Monte Carlo simulations of the DE model with four delays at 15 minutes.

It can be observed from Figures 4.6 - 4.7 that the estimated volatility obtained from the DE algorithm provides similar results to the ARC algorithm in terms of sampling frequency and delays. The estimated volatility follows a similar movement to the actual volatility for all stocks. Increasing the delay provides better results in the matching part as the estimated volatility is close to the actual volatility. Using 5 minutes data has significantly outperformed the 15 minutes data.

4.3.4 The Comparison of Estimated Volatility Obtained from the ARC and Classical DE Algorithm

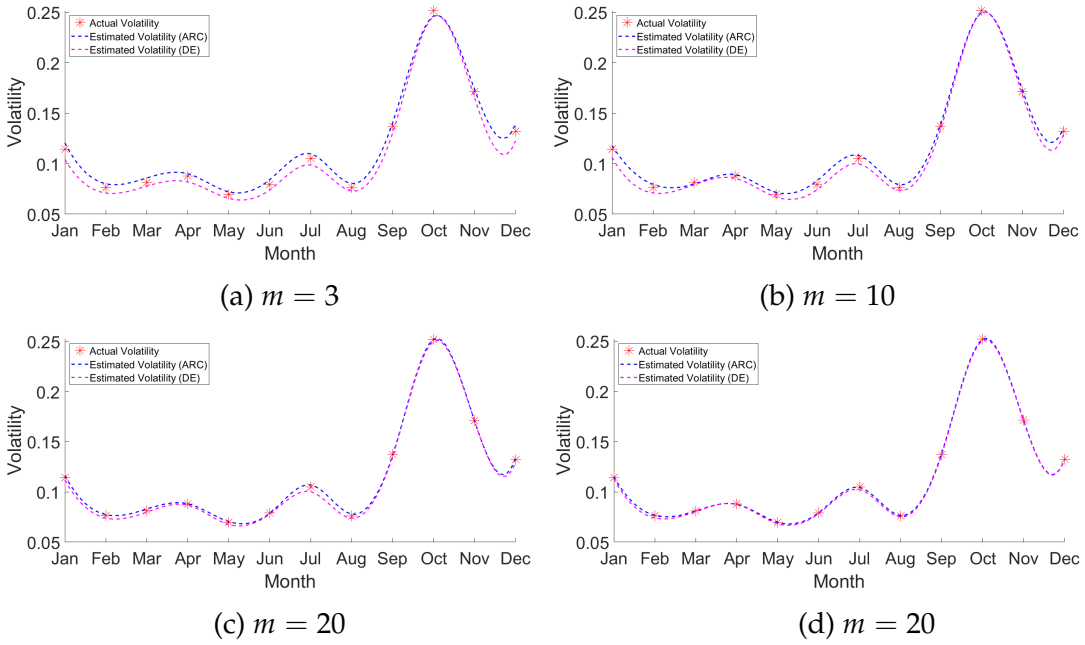
To compare the performance of the ARC algorithm with the classical differential evolution (DE), the IBM, MSFT, S&P 500 and S&P 100 stocks at 5 minutes and 15 minutes time intervals in 2008 are used. The comparisons of the estimated volatility obtained from the ARC and DE algorithms with four delays at 5 minutes time intervals are displayed in Figures 4.8 - 4.11.



(a) $m = 3$ (b) $m = 10$

(c) $m = 20$ (d) $m = 20$

Figure 4.8: Actual volatility of 5 minutes IBM stock (asterisk mark) and its estimates provided by the ARC and the DE algorithms with four delays.



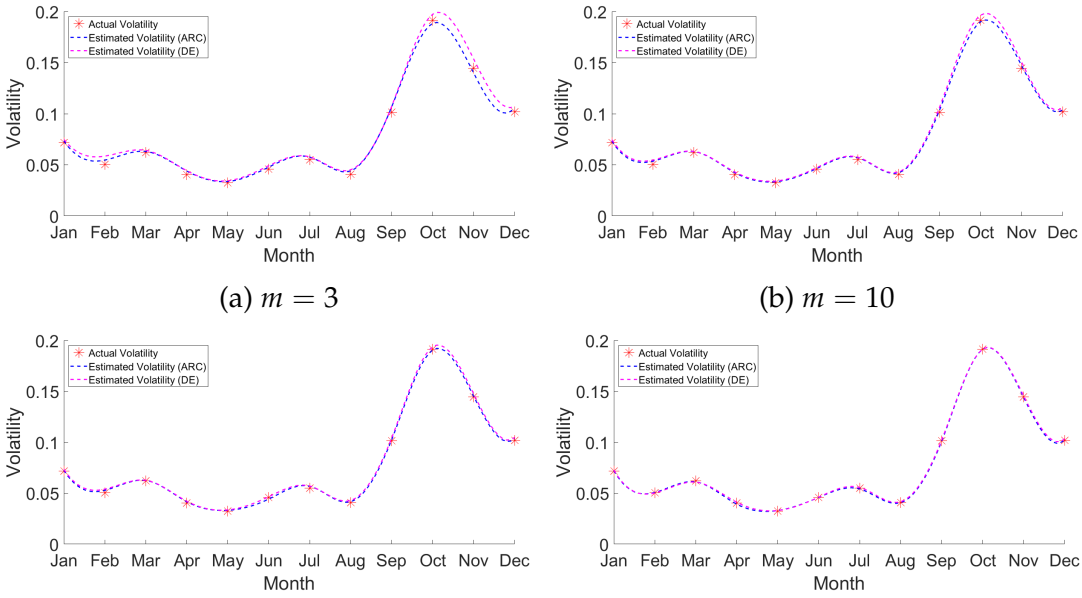
(a) $m = 3$ (b) $m = 10$

(c) $m = 20$ (d) $m = 20$

Figure 4.9: Actual volatility of 5 minutes MSFT stock (asterisk mark) and its estimates provided by the ARC and the DE algorithms with four delays.

The comparisons of the estimated volatility obtained from the ARC and DE algorithms with four delays at 15 minutes time intervals are displayed in Figures 4.12 - 4.15.

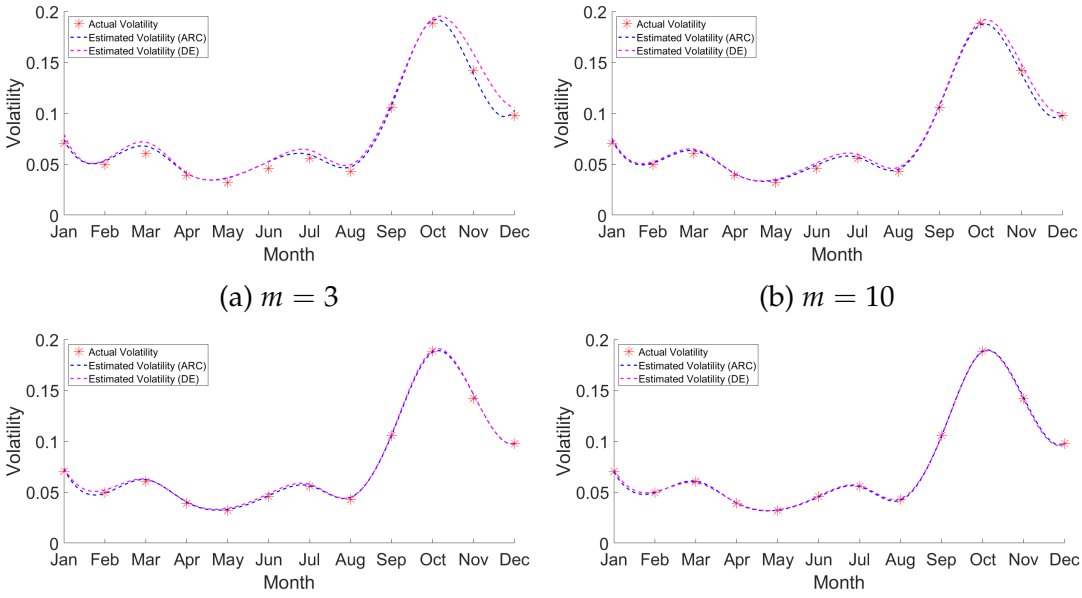
As we can see from Figures 4.8 - 4.15, it can be noticed that the patterns of the estimated volatility from both algorithms are similar to the actual volatility. The ARC algorithm outperforms the basic DE algorithm as the estimated volatility is close to the actual volatility for all stocks with four various delays at 5 minutes and 15 minutes. In addition, it is obviously seen that larger delays provide better results than small delays. To investigate the performance of different delays, the evaluation metrics, namely the root mean square error (RMSE), the mean absolute error (MAE), and the mean absolute percentage error (MAPE), are employed. The RMSE, MAE, and MAPE obtained from all stocks at two different time intervals with four delays are shown in Figures 4.16 - 4.17.



(a) $m = 3$ (b) $m = 10$

(c) $m = 20$ (d) $m = 20$

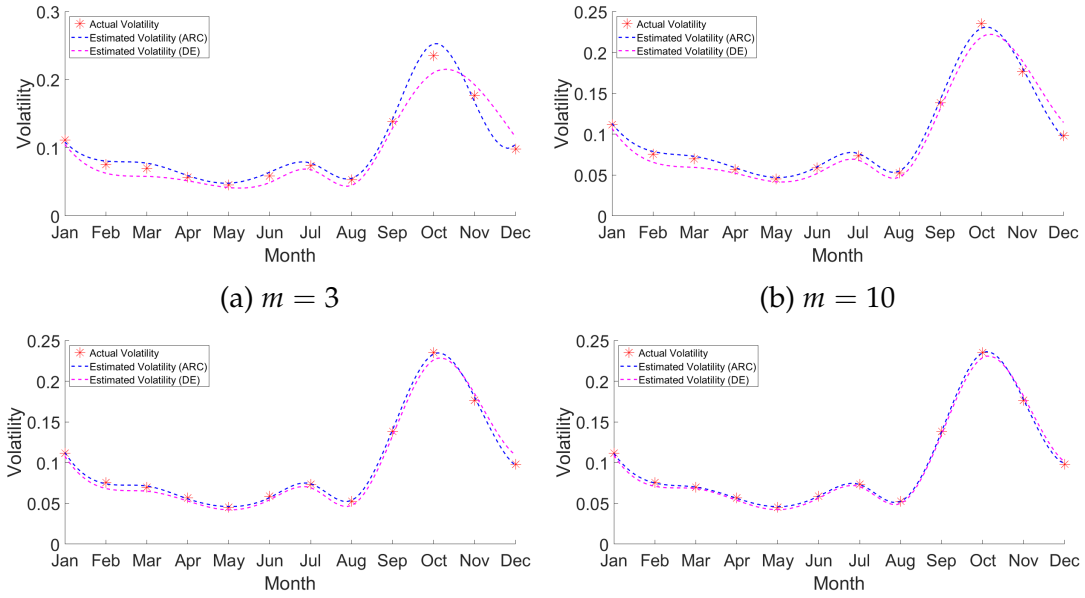
Figure 4.10: Actual volatility of 5 minutes S&P 500 stock (asterisk mark) and its estimates provided by the ARC and the DE algorithms with four delays.



(a) $m = 3$ (b) $m = 10$

(c) $m = 20$ (d) $m = 20$

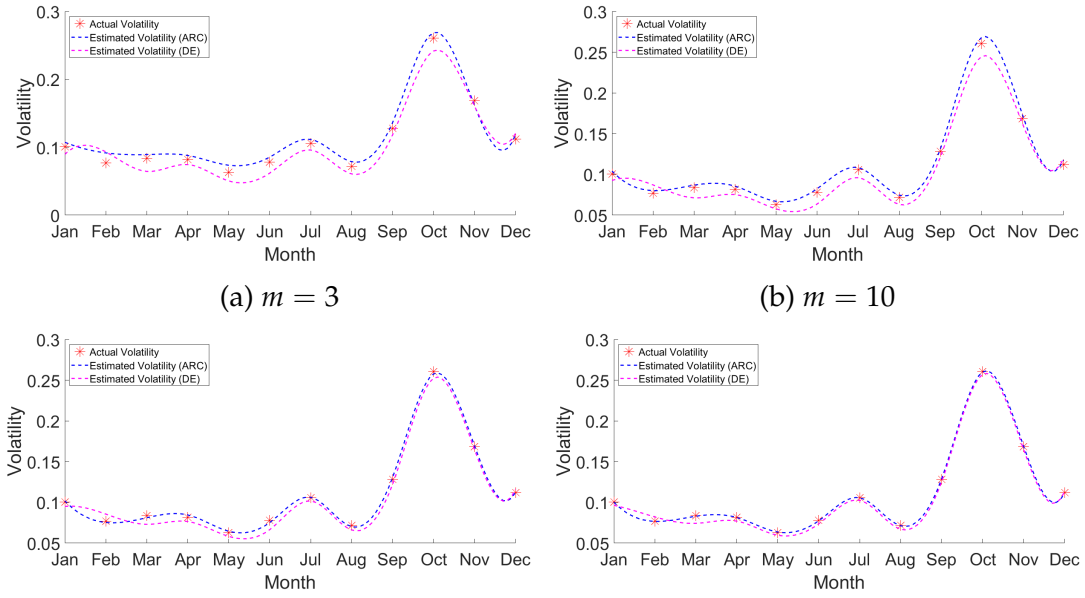
Figure 4.11: Actual volatility of 5 minutes S&P 100 stock (asterisk mark) and its estimates provided by the ARC and the DE algorithms with four delays.



(a) $m = 3$ (b) $m = 10$

(c) $m = 20$ (d) $m = 20$

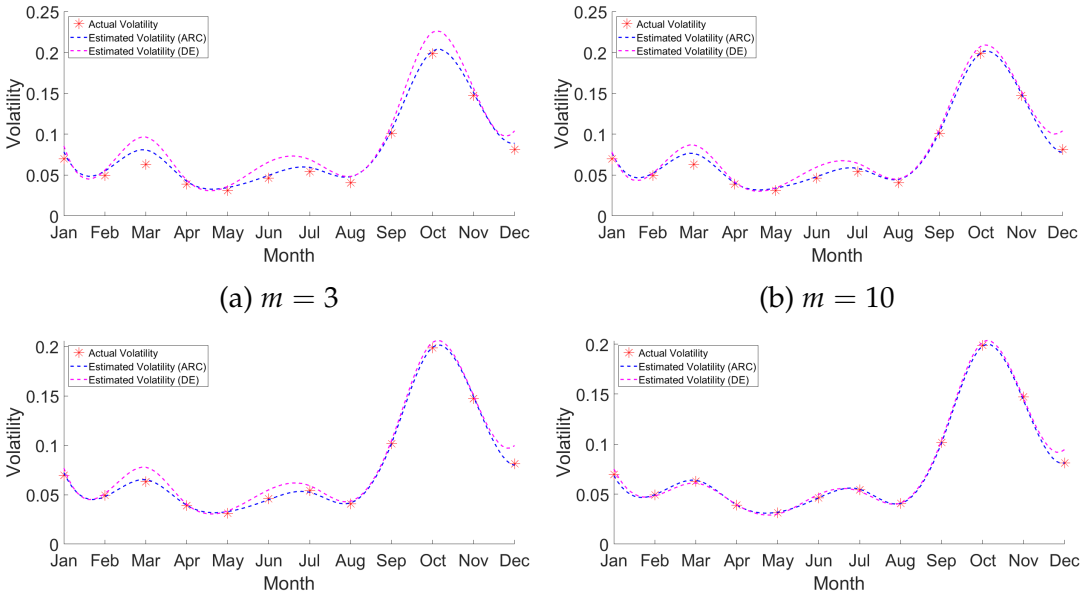
Figure 4.12: Actual volatility of 15 minutes IBM stock (asterisk mark) and its estimates provided by the ARC and the DE algorithms with four delays.



(a) $m = 3$ (b) $m = 10$

(c) $m = 20$ (d) $m = 20$

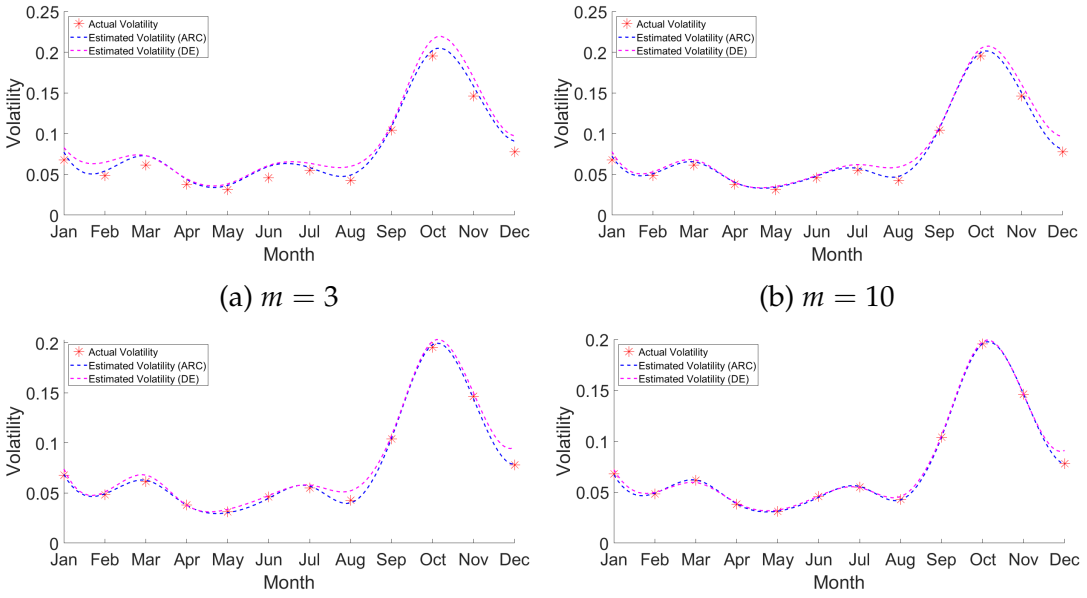
Figure 4.13: Actual volatility of 15 minutes MSFT stock (asterisk mark) and its estimates provided by the ARC and the DE algorithms with four delays.



(a) $m = 3$ (b) $m = 10$

(c) $m = 20$ (d) $m = 20$

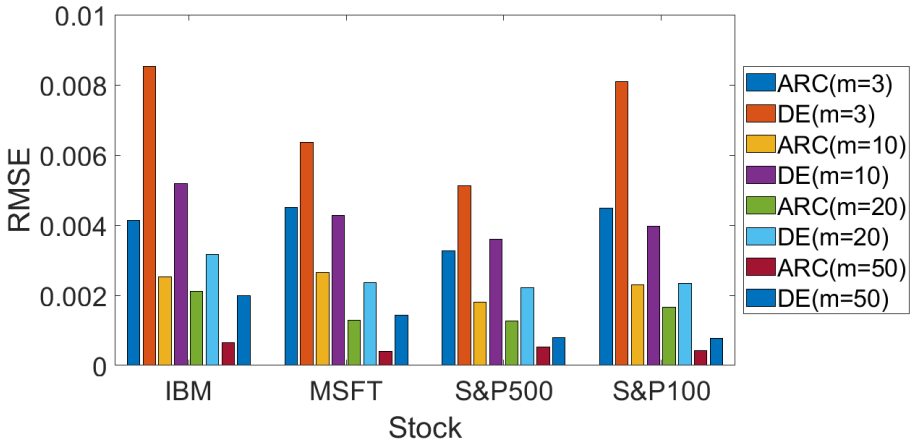
Figure 4.14: Actual volatility of 15 minutes S&P 500 stock (asterisk mark) and its estimates provided by the ARC and the DE algorithms with four delays.



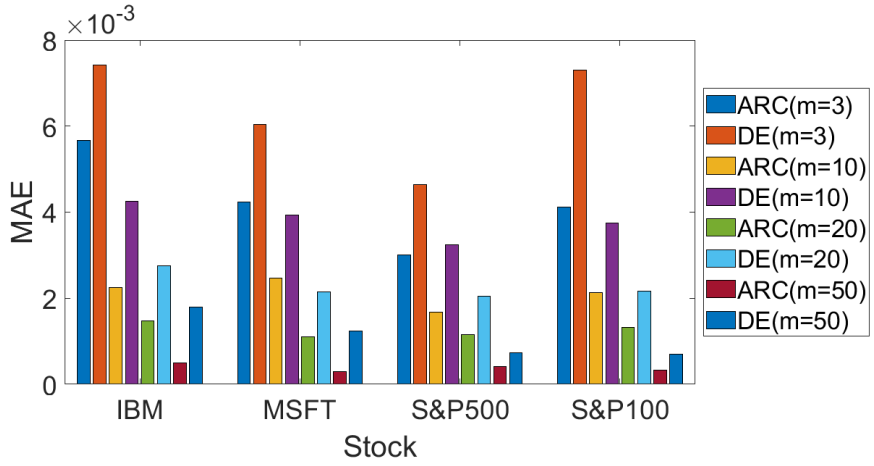
(a) $m = 3$ (b) $m = 10$

(c) $m = 20$ (d) $m = 20$

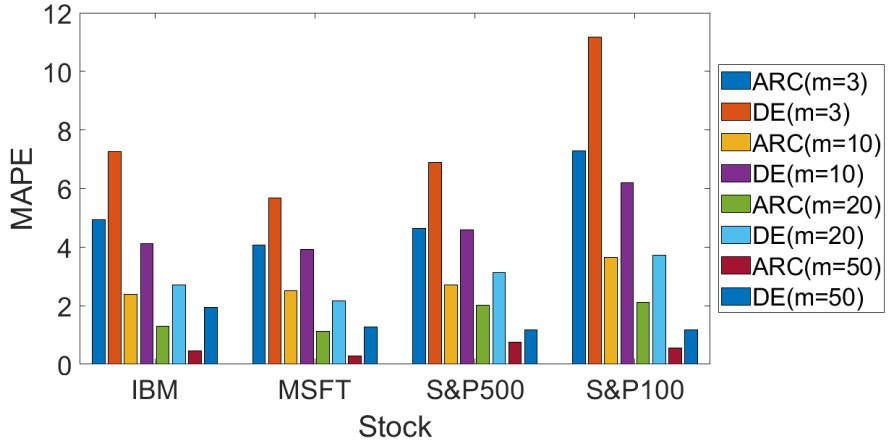
Figure 4.15: Actual volatility of 15 minutes S&P 100 stock (asterisk mark) and its estimates provided by the ARC and the DE algorithms with four delays.



(a) RMSE

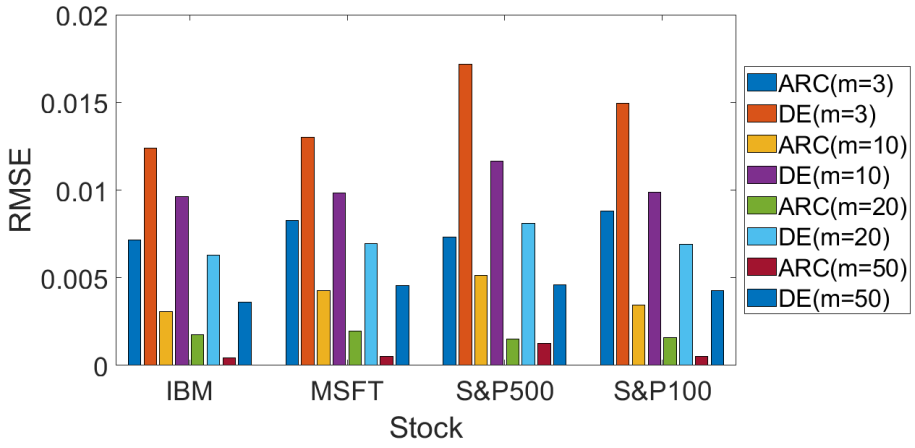


(b) MAE

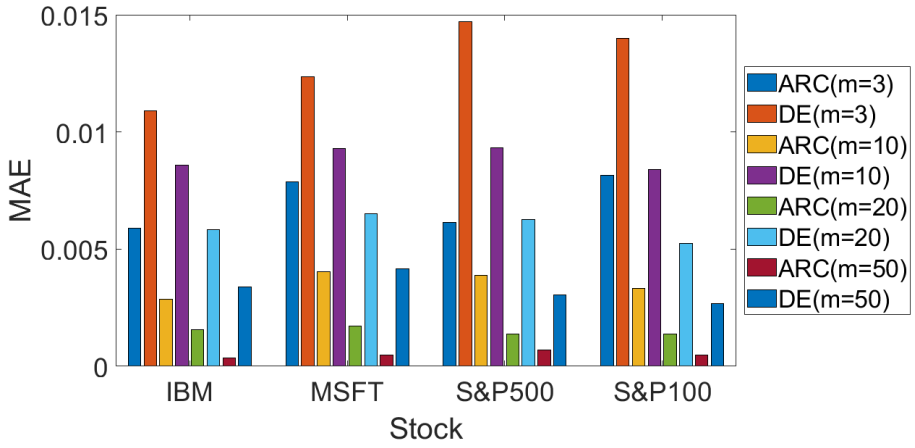


(c) MAPE

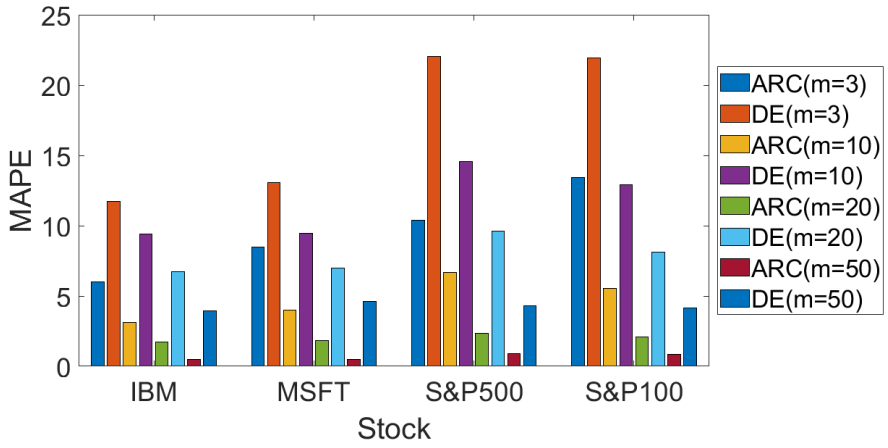
Figure 4.16: The RMSE, MAE and MAPE of four stocks at 5 minutes time interval



(a) RMSE



(b) MAE



(c) MAPE

Figure 4.17: The RMSE, MAE and MAPE of four stocks at 15 minutes time interval

Figures 4.16 - 4.17 present the comparison of the evaluation metrics between the ARC and the DE algorithms. Our proposed (ARC) algorithm significantly outperforms the DE algorithm in terms of RMSE, MAE, and MAPE for all four stocks with four different delays.

4.4 Model Identification for Two Parameters

In this section, we assume that the two model parameters, namely the drift term parameter (λ) and the volatility of the SDDE in Eq. (4.12), are unknown. The differential evolution (DE) algorithm is used to estimate two parameters to match the volatility. The pseudocode of the DE algorithm is displayed in Algorithm 6.

The idea of Algorithm 6 is similar to Algorithm 5 except the problem size (D) because this section considers two unknown model parameters $(D = 2)$ while the previous section works with one unknown parameter $(D = 1)$. After the SDDE model parameters are computed, we calculate the estimated volatility for each month. Finally, the actual volatility and the estimated volatility at 5 minutes and 15 minutes intervals of the IBM, MSFT, S&P 500, and S&P 100 stocks with four delays are presented in Tables 4.13 - 4.16.

As shown from Tables 4.13 - 4.16, the estimated volatility obtained from the DE algorithm provides a similar pattern to the actual volatility. The estimated volatility using a large delay provides better results than a small delay for four stocks. The comparison between the estimated volatility obtained from SDDE and the real-world volatility is displayed in Figures 4.18 - 4.19.

It can be observed from Figures 4.18 - 4.19 that the estimated volatility follows a similar movement to the actual volatility for all stocks. Increasing the delay provides better results in the matching part as the estimated volatility is close to the actual volatility.

Algorithm 6 DE Pseudocode for two-parameter identification

Require: $NP = 20$, $CR = 0.6$, $D = 2$, $F = 0.2$, $m = 3, 10, 20, 50$, $\delta = 1/n$, $TOL = 0.01$, $X_1^L = -200$, $X_1^U = 200$, $X_2^L = 0.01$, $X_2^U = 1$, $MAPE_{min} = 100$, $maxG = 1000$ and $maxI = 12$.

Import the stock price $\{y_k\}_{k=1}^n$
while($I < maxI$**)**
Generate the initial population : set $(I = 0, G = 0)$
    $x_{i,j}^G = L_j + rand(0,1) \cdot (x_j^U - x_j^L), \quad i = 1, \ldots, NP \quad j = 1, \ldots, D$
    while($MAPE_{min} > TOL$ **or** $G < maxG$**)**
        for $i = 1$ to NP
        Randomly select $r_1, r_2, r_3 \in 1, 2, \ldots, NP$ where $r_1 \neq r_2 \neq r_3 \neq i$
        Mutation Step
            $v_{i,j}^G = x_{r1,j}^G + F \cdot (x_{r2,j}^G - x_{r3,j}^G)$
            $j_{rand} = [rand(0,1) \cdot D]$
        Crossover
            for $j = 1$ to 2
                if $rand(0,1) <= CR \;\; or \;\; j = j_{rand}$
                $u_{i,j}^G = v_{i,j}^G$
                else
                $u_{i,j}^G = x_{i,j}^G$
                end if
            end for
        Selection: $sumE_m = 0, sumE = 0$
            for $k = m + 1$ to n
                $f_i^m(k) = (1 - \delta \cdot u_{i,1}^G) y_{k-1} + (\delta \cdot u_{i,1}^G) y_{k-m} + \sqrt{\delta} \cdot u_{i,2}^G \cdot randn(0,1)$
                $f_i(k) = (1 - \delta \cdot x_{i,1}^G) y_{k-1} + (\delta \cdot x_{i,1}^G) y_{k-m} + \sqrt{\delta} \cdot x_{i,2}^G \cdot randn(0,1)$
                $e^m(k) = \left| f_i^m(k) - y(k) \right| / y(k)$
                $e(k) = \left| f_i(k) - y(k) \right| / y(k)$
                $sumE_m = sumE_m + e^m(k)$
                $sumE = sumE + e(k)$
            end for
            $MAPE_m = sumE_m \times 100/(n - m)$
            $MAPE = sumE \times 100/(n - m)$
                if $MAPE_m < MAPE$
                $x_{i,1}^{G+1} = u_{i,1}^G;$
                $x_{i,2}^{G+1} = u_{i,2}^G;$
                $MAPE_i = MAPE_m$
                else
                $x_{i,1}^{G+1} = x_{i,1}^G$
                $x_{i,2}^{G+1} = x_{i,2}^G$
                $MAPE_i = MAPE$
                end if
        end for
        $MAPE_{min} = min(MAPE_i)$
        $G = G + 1$
        Compute the volatility (Eq. (4.12)) using the optimal $x_{i,1}^{G+1} and x_{i,2}^{G+1}$
    end while
    $I = I + 1$
end while

Table 4.13: The estimated volatility obtained from the DE algorithm at each sampling frequency of IBM stock with four different delays in 2008

.

Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.128238	0.24534	0.125516	0.126534	0.127103
	15-min	0.111404	0.102988	0.106083	0.108231	0.109040
Feb	5-min	0.076853	0.073234	0.074196	0.074272	0.075339
	15-min	0.075787	0.066058	0.070553	0.072653	0.073225
Mar	5-min	0.070163	0.067790	0.068883	0.069330	0.069693
	15-min	0.070075	0.061653	0.064439	0.068148	0.069927
Apr	5-min	0.059146	0.056139	0.057500	0.057910	0.058013
	15-min	0.056966	0.049258	0.052996	0.054893	0.055336
May	5-min	0.050221	0.047354	0.048995	0.049035	0.049258
	15*min	0.045363	0.039585	0.040919	0.042136	0..042607
Jun	5-min	0.062038	0.059267	0.060818	0.061291	0.061389
	15-min	0.058700	0.049129	0.054984	0.056855	0.057032
Jul	5-min	0.076856	0.073512	0.073706	0.075086	0.075885
	15-min	0.073569	0.068534	0.070190	0.070948	0.071210
Aug	5-min	0.058617	0.055527	0.056125	0.057020	0.057693
	15-min	0.052804	0.047701	0.048808	0.049389	0.051193
Sep	5-min	0.148514	0.138606	0.144517	0.145096	0.147641
	15-min	0.138231	0.123459	0.132753	0.134970	0.135104
Oct	5-min	0.258056	0.248788	0.249065	0.254946	0.255797
	15-min	0.235271	0.205623	0.226104	0.228266	0.232342
Nov	5-min	0.175672	0.167558	0.171442	0.173507	0.174353
	15-min	0.176757	0.168604	0.172557	0.173290	0.174163
Dec	5-min	0.121681	0.118064	0.119937	0.120230	0.120947
	15-min	0.098087	0.116222	0.113490	0.111033	0.108649

Table 4.14: The estimated volatility obtained from the DE algorithm at each sampling frequency of MSFT stock with four different delays in 2008

.

Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.114330	0.104788	0.110747	0.111965	0.113142
	15-min	0.100703	0.088190	0.094377	0.097476	0.098074
Feb	5-min	0.076580	0.072917	0.073019	0.074512	0.074669
	15-min	0.076787	0.067055	0.068514	0.072037	0.073590
Mar	5-min	0.081282	0.077285	0.077580	0.079546	0.080533
	15-min	0.083489	0.078907	0.079913	0.080037	0.081053
Apr	5-min	0.087885	0.083991	0.085171	0.086724	0.087319
	15-min	0.081779	0.076551	0.078392	0.078914	0.079846
May	5-min	0.069760	0.063677	0.066851	0.068001	0.068739
	15-min	0.063008	0.056786	0.057166	0.059682	0.060008
Jun	5-min	0.079216	0.074309	0.076855	0.077579	0.078512
	15-min	0.078077	0.072225	0.074672	0.076406	0.076637
Jul	5-min	0.104967	0.101422	0.102646	0.103056	0.103717
	15-min	0.105926	0.098479	0.100773	0.102298	0.103390
Aug	5-min	0.076249	0.070168	0.074443	0.074916	0.075074
	15-min	0.071691	0.064247	0.065166	0.067167	0.068029
Sep	5-min	0.137254	0.127751	0.132470	0.136644	0.136899
	15-min	0.128023	0.121539	0.122622	0.125971	0.126749
Oct	5-min	0.251875	0.243867	0.247414	0.250824	0.251905
	15-min	0.260465	0.248352	0.252651	0.258612	0.260509
Nov	5-min	0.171256	0.164481	0.166776	0.169422	0.171122
	15-min	0.168367	0.162475	0.164643	0.165397	0.166593
Dec	5-min	0.132252	0.124107	0.129175	0.130823	0.131422
	15-min	0.111982	0.124265	0.121940	0.115305	0.115516

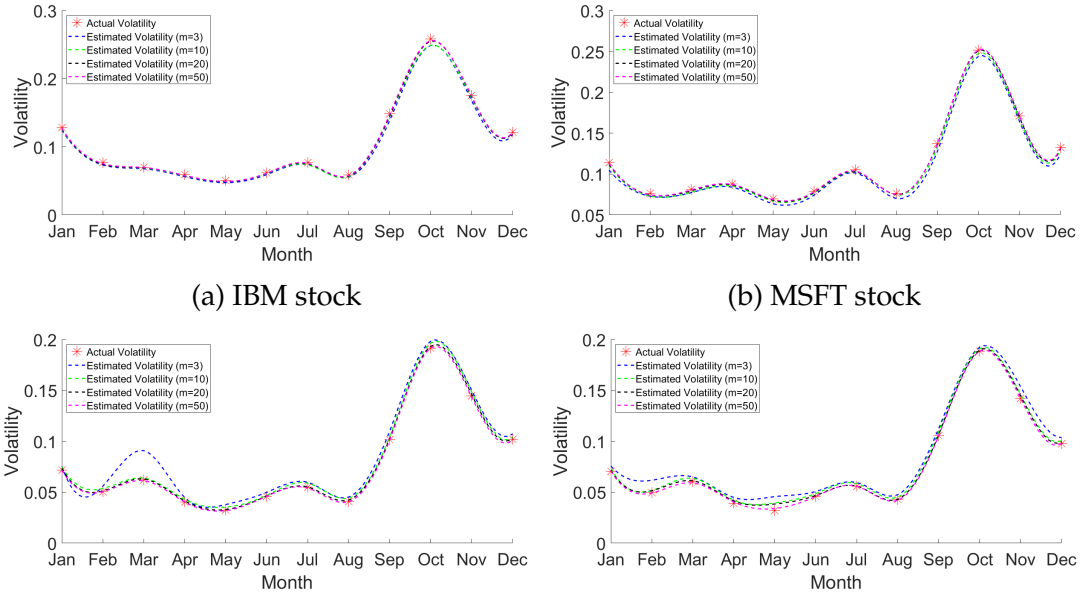
Table 4.15: The estimated volatility obtained from the DE algorithm at each sampling frequency of S&P 500 stock with four different delays in 2008

Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.071822	0.075590	0.075011	0.073630	0.071483
	15-min	0.069563	0.084017	0.076522	0.071371	0.068735
Feb	5-min	0.050601	0.055916	0.053631	0.051194	0.050527
	15-min	0.049258	0.053071	0.051652	0.051641	0.048771
Mar	5-min	0.061976	0.090971	0.063883	0.063011	0.061376
	15-min	0.063043	0.073444	0.066146	0.061822	0.062017
Apr	5-min	0.040261	0.044748	0.043036	0.041363	0.040298
	15-min	0.038989	0.050395	0.045584	0.040225	0.037657
May	5-min	0.032422	0.037906	0.035362	0.032971	0.031932
	15-min	0.031329	0.042935	0.034608	0.034578	0.030921
Jun	5-min	0.045904	0.049937	0.047025	0.046362	0.046013
	15-min	0.046096	0.052598	0.050704	0.048221	0.045092
Jul	5-min	0.054960	0.059773	0.058876	0.055093	0.054137
	15-min	0.054173	0.070077	0.063354	0.052763	0.054137
Aug	5-min	0.040903	0.045243	0.044240	0.042361	0.040536
	15-min	0.040682	0.060255	0.048910	0.043746	0.039016
Sep	5-min	0.101656	0.109990	0.104733	0.102721	0.101035
	15-min	0.101457	0.112502	0.106917	0.105077	0.100179
Oct	5-min	0.191253	0.198176	0.196477	0.193212	0.191422
	15-min	0.198555	0.208681	0.207043	0.200071	0.197199
Nov	5-min	0.144492	0.150168	0.148467	0.146072	0.143644
	15-min	0.147319	0.154971	0.151584	0.145758	0.146120
Dec	5-min	0.102008	0.107367	0.104171	0.103065	0.101387
	15-min	0.081194	0.102250	0.099773	0.098842	0.090866

Table 4.16: The estimated volatility obtained from the DE algorithm at each sampling frequency of S&P 100 stock with four different delays in 2008

.

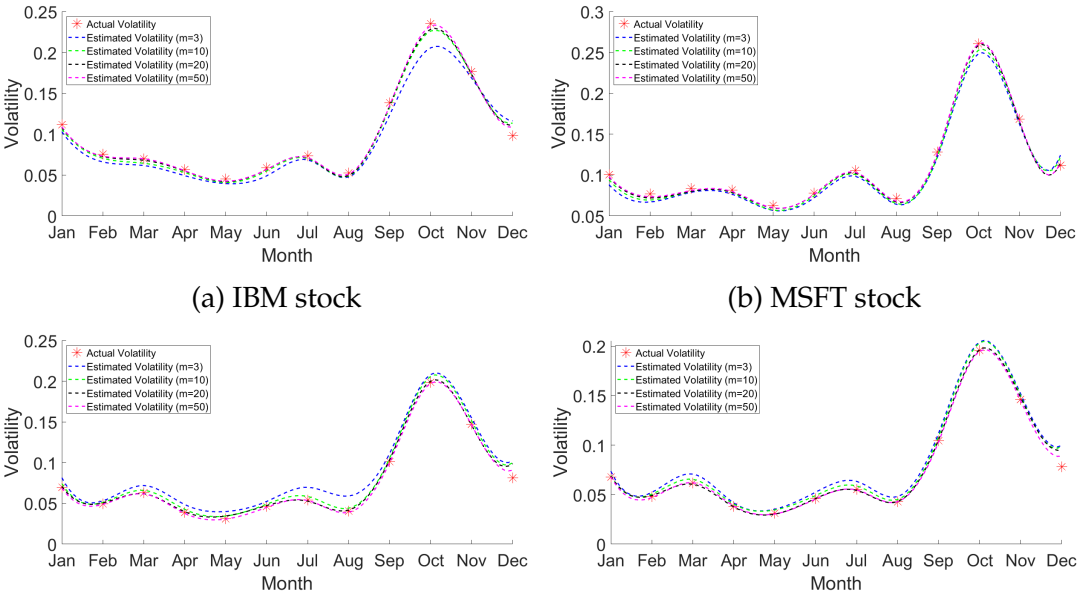
Month	Time	σ	$\hat{\sigma}_{m=3}$	$\hat{\sigma}_{m=10}$	$\hat{\sigma}_{m=20}$	$\hat{\sigma}_{m=50}$
Jan	5-min	0.070590	0.075687	0.073539	0.071494	0.070818
	15-min	0.067839	0.073600	0.069362	0.068863	0.068612
Feb	5-min	0.049385	0.061755	0.051767	0.051452	0.049284
	15-min	0.048145	0.052288	0.050533	0.049135	0.047061
Mar	5-min	0.060436	0.065020	0.063688	0.061243	0.059220
	15-min	0.061576	0.070731	0.065452	0.060507	0.062018
Apr	5-min	0.039062	0.044834	0.042138	0.041518	0.040311
	15-min	0.038107	0.041610	0.040418	0.037575	0.038539
May	5-min	0.032110	0.045690	0.039531	0.038416	0.033969
	15-min	0.031074	0.035311	0.034545	0.030350	0.030441
Jun	5-min	0.045730	0.050694	0.048718	0.046840	0.045791
	15-min	0.045950	0.052535	0.049169	0.046174	0.045615
Jul	5-min	0.055746	0.059506	0.058621	0.056018	0.055959
	15-min	0.055108	0.063336	0.058711	0.054273	0.054726
Aug	5-min	0.042842	0.047555	0.045163	0.043048	0.042104
	15-min	0.042794	0.048118	0.045086	0.043007	0.042898
Sep	5-min	0.105564	0.112497	0.109613	0.106575	0.105433
	15-min	0.104144	0.112197	0.108861	0.105429	0.103403
Oct	5-min	0.188473	0.192281	0.191103	0.189607	0.188347
	15-min	0.195552	0.204228	0.203197	0.196433	0.194705
Nov	5-min	0.142027	0.153560	0.145886	0.144671	0.141921
	15-min	0.146258	0.151676	0.149740	0.148211	0.144825
Dec	5-min	0.097710	0.103688	0.100751	0.099383	0.097990
	15-min	0.077879	0.099670	0.098753	0.095619	0.089140



(a) IBM stock (b) MSFT stock

(c) S&P 500 stock (d) S&P 100 stock

Figure 4.18: Actual volatility of four stocks (asterisk mark) and its estimates provided by Monte Carlo simulations of the DE algorithm with four delays at 5 minutes time intervals.



(a) IBM stock (b) MSFT stock

(c) S&P 500 stock (d) S&P 100 stock

Figure 4.19: Actual volatility of four stocks (asterisk mark) and its estimates provided by Monte Carlo simulations of the DE algorithm with four delays at 15 minutes time intervals.

The evaluation metrics of the estimated volatility on the IBM, MSFT, S&P 500 and S&P 100 at 5 minutes and 15 minutes time intervals are presented in Table 4.17.

Table 4.17: The evaluation metrics on four stocks at 5 minutes and 15 minutes time intervals in 2008

Stock	Delay	5 minutes			15 minutes		
		RMSE	MAE	MAPE	RMSE	MAE	MAPE
IBM	3	0.005332	0.004640	4.476460	0.012796	0.010872	11.489900
	10	0.003602	0.002946	2.746930	0.006665	0.005829	6.743270
	20	0.001991	0.001817	1.874120	0.004996	0.004008	4.512810
	50	0.001168	0.001079	1.171890	0.003764	0.002859	3.291120
MSFT	3	0.006538	0.006179	5.746870	0.008455	0.007983	7.995730
	10	0.003435	0.003313	3.113090	0.006023	0.005699	5.841640
	20	0.001641	0.001575	1.616360	0.003270	0.003139	3.439050
	50	0.000969	0.000826	0.930076	0.002500	0.002288	2.615260
S&P500	3	0.009830	0.007279	11.621300	0.011964	0.010944	18.746500
	10	0.003218	0.003055	4.818490	0.006662	0.005122	7.650710
	20	0.001195	0.001065	1.562580	0.005338	0.002863	4.340250
	50	0.000505	0.000425	0.642891	0.003008	0.001769	2.623520
S&P100	3	0.007692	0.006924	12.362200	0.008889	0.007573	11.695900
	10	0.003646	0.003404	6.128120	0.007057	0.004950	7.271250
	20	0.002327	0.001716	3.437930	0.005210	0.002289	3.110570
	50	0.000778	0.000525	1.192670	0.003322	0.001501	2.071710

The results from Figures 4.18 - 4.19 and Table 4.17 confirm that the estimated volatility obtained from the DE algorithm for two model parameters follows a similar movement to the actual volatility. The evaluation metrics obtained from 5 minutes stock prices are lower than 15 minutes stock prices. Consequently, using 5 minutes data has significantly outperformed the 15 minutes data.

4.5 Conclusions

This chapter focuses on the matching volatility obtained from the SDDE and the real-world stock prices. First, we divided the matching part into two cases: one unknown and two unknown parameters. For one parameter estimation,

the autoregressive coefficients (ARC) algorithm is proposed to identify the parameter of the drift term (λ) while assuming the volatility is known. Next, we select the closing prices, the final price on the trading day, every 5 minutes and 15 minutes from the Thomson Reuters database during the 2008 global financial crisis and Great Recession of the IBM, MSFT, S&P 500 and S&P 100 stocks. The performance of the ARC algorithm is compared with the DE algorithm for all four stocks with two different sampling frequencies. The experimental results suggest that the estimated volatility obtained from the proposed ARC algorithm performs better than the classical DE algorithm in terms of the RMSE, MAE, and MAPE. For two parameters, the DE algorithm is utilized to estimate two unknown model parameters. The empirical results obtained from the DE algorithm indicate that the estimated volatility of each stock has a similar pattern to the actual price. We then investigate the performance of the delay. The results obtained from one and two model identification at two sampling frequencies confirm that the evolution metrics decrease as the delay increases. Moreover, the financial time series at 5 minutes time intervals provide smaller evolution metrics than 15 minutes data. In conclusion, the study becomes more accurate when the delay is increased, and a higher sampling frequency leads to higher estimation accuracy.

Chapter 5

A Two-Delay Combination Model for Stock Price Prediction

This chapter presents a new technique of the combination method to increase the prediction accuracy of the individual deep learning model using a two-delay combination method. We also present the new weight of the combination model using the differential evolution algorithm. The chapter is structured as follows. Section 5.1 presents the deep learning methods. Section 5.2 displays the overfitting and underfitting with deep learning models. Section 5.3 represents the data visualisation. The ideas to estimate weights and hyperparameter optimisation are shown in Section 5.4. The empirical results are represented in Section 5.5 for univariate and multivariate time series. Finally, Section 5.6 makes a summary of this work. Note that the work in this chapter has been published in [163].

5.1 Deep Learning Methods

Deep learning techniques (DL) have been widely studied in stock price prediction in recent years because the nature of stock price time series is usually non-linear, non-parametric, and chaotic, which is highly difficult to fit

models using traditional statistics procedures. Three well-known DLs used in this chapter are the multilayer perceptron (MLP), the convolutional neural network (CNN) and the long short-term memory (LSTM). Moreover, combination methods have become widespread in stock price prediction, and many more suitable models for stock prediction have been proposed. The linear combination method is one of the ways for many scholars to predict the stock price more accurately. This technique was introduced by Bates, and Granger [164]. The main objective of this approach is to combine the benefits of different single forecasting models. Several studies have confirmed that the combination forecast has superior efficiency to the individual method [165]. However, weight selection is a significant challenge of the linear combination forecast method. To our knowledge, there is no literature on combining different delays for stock price prediction. Moreover, little work in the literature was studied on the delay (look back) period [129]. This motivates us to combine two delay forecasts and estimate the weights using the differential evolution algorithm. This chapter presents the two-delay combination techniques and estimates the hybrid model's weight using the differential evolution (DE) algorithm. The experiment in this chapter consists of seven parts: data collecting, data pre-possessing, hyperparameter optimization, model training, model saving, model testing, and out-of-sample prediction results. The flowchart of the deep learning (DL) process is shown in Figure 5.1. [3]

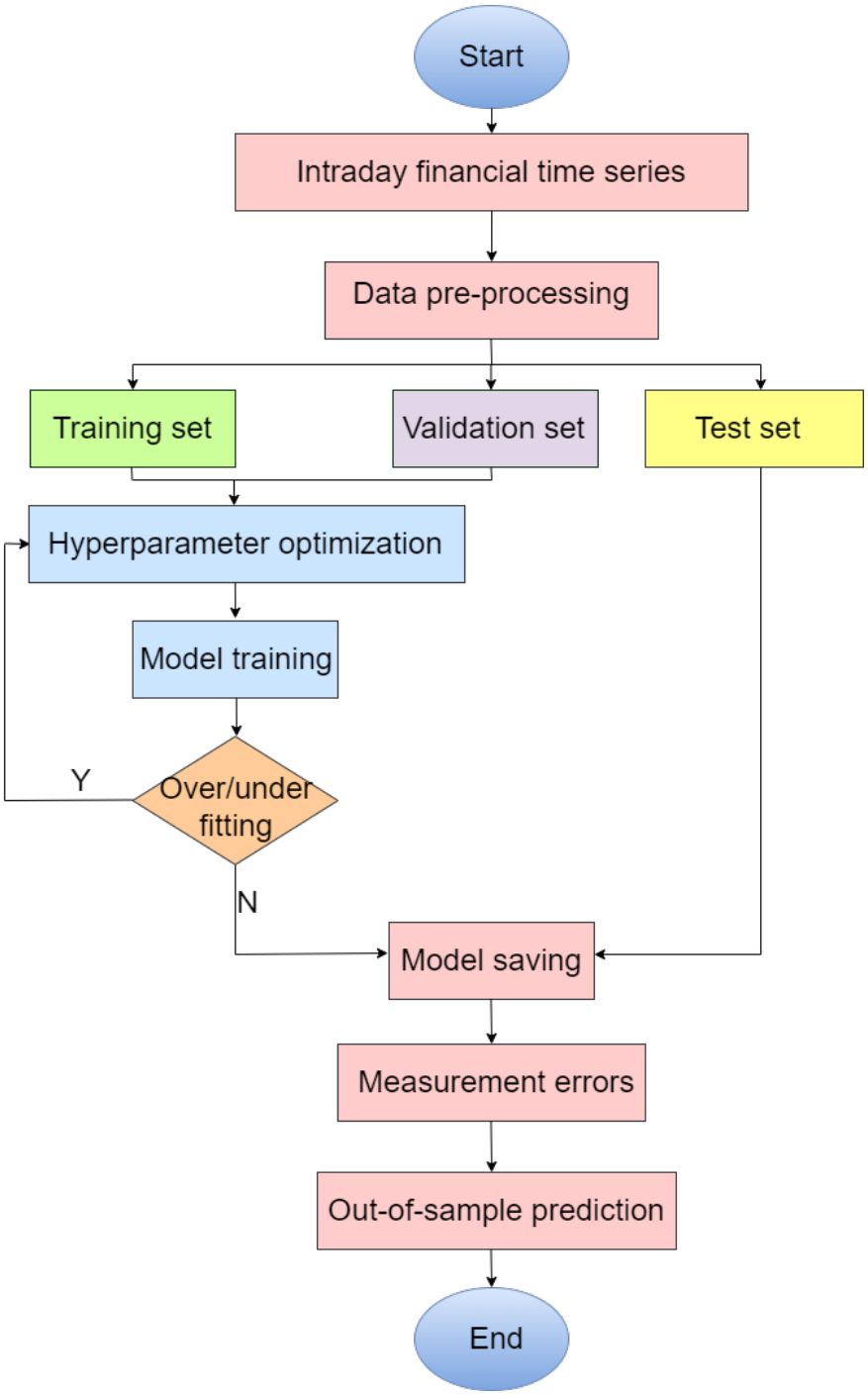


Figure 5.1: The flowchart of the deep learning process.

5.2 Overfitting and Underfitting with Deep Learning Models

Overfitting and underfitting are widespread problems in machine learning. An extensive range of literature has been dedicated to studying methods for preventing these problems. A loss curve can present both against epochs [166]. Overfitting is one of the most significant problems in training neural networks. This occurs when the model performs well on training data but generalizes poorly to unseen data (the training loss is significantly lower than the validation loss) [167–171]. Underfitting is a model that fails to learn the problem sufficiently and performs poorly on both training and validation datasets (high loss in both training and validation sets) [170, 172]. In addition, it cannot correctly handle other data not contained in the training data. An excellent fit model means the model that suitably learns the training dataset and generalizes well to the holdout dataset (a training and validation loss decreases to stability with a minimal gap between the two final loss values) [173]. Underfitting is often not discussed as it is easy to detect given a good performance metric. The machine learning approach has obtained an excellent predictive performance if the values of both training and testing data are almost similar [174].

Regularization is a technique used for preventing neural networks from overfitting. The most common regularization methods are dropout, weight regularization, and early stopping. Dropout was used as a powerful method to prevent overfitting [175]. The term dropout refers to dropping out units (hidden and visible) in a neural network. Dropout can be used on the input or/and hidden layer, but it is not used on the output layer. Weight regularization provides an approach to reduce the overfitting of a neural network model on the training data and improve the model's performance on new data, such as the holdout test set. There are multiple types of weight regularization, such

as L1 (sum of the absolute weights), L2 (sum of the squared weights), and L1L2 (sum of the absolute and the squared weights) [173]. Early stopping is a simple, effective, and widely used approach to training neural networks. The key idea with early stopping is to keep track of the model parameters that give the best performance over the validation set and then to stop the training after this best performance so far over the validation set does not improve over a predefined number of training steps [176].

5.3 Data

In this chapter, the raw data are collected from the Thomson Reuters database every 5 minutes between January 2021 and January 2022 totally of 21,409 observations [152] to predict the closing stock prices. For univariate data, only the closing price is an input variable. In contrast, the five inputs for the multivariate data set are the opening price, the highest price, the lowest price, the closing price, and the trading volume (OHLCV). We selected four big companies' stocks on the New York Stock Exchange (NYSE), which operates on a weekday from 9.30 am. to 4.00 pm. without a lunch break, including Apple Inc. (AAPL), Adobe Inc. (ADBE), Devon Energy Corporation (DVN), and Moderna, Inc. (MRNA) stocks. The summary statistics of the original closing prices from four stocks, including mean, median, maximum, minimum, standard deviation, skewness and kurtosis, are presented in Table 5.1.

Table 5.1: Descriptive statistics.

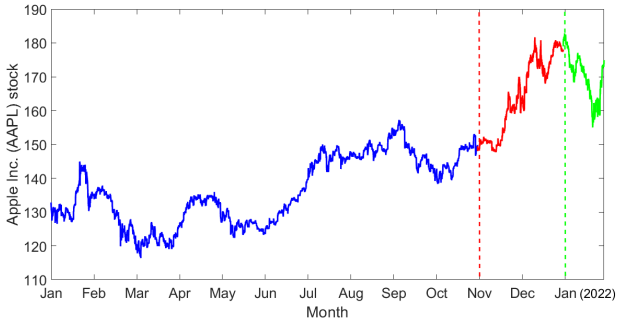
Statistics	AAPL	ADBE	DVN	MRNA
Mean	143.0679	557.2368	30.6329	242.8652
Median	142.8700	554.2500	28.0600	28.0600
Maximum	182.8400	699.2300	54.1300	492.8100
Minimum	116.3500	421.3700	15.7600	104.1700
Standard deviation	16.0622	73.7783	9.4154	96.8969
Skewness	2.8282	2.8247	2.8284	2.8267
Kurtosis	7.9990	7.9844	7.9999	7.9928

The data used in this chapter are the univariate and multivariate time series. We present only the characteristics of the closing prices in Table 5.1 because we are focusing on the closing price prediction for both cases. Apple Inc. (AAPL) is a specialised company in consumer electronics, software and online services. The average price is 143.0679 USD, and the standard deviation is 16.0622. Adobe Inc. (ADBE) is an American multinational computer software company incorporated in Delaware. The price peaked at 699.23 USD and dropped to the lowest at 421.37 USD. Devon Energy Corporation (DVN) is an energy company engaged in hydrocarbon exploration in the USA. The mean of DVN stock and the standard deviation are 30.6329 and 9.4154 USD, respectively. Moderna, Inc (MRNA) is an American pharmaceutical and biotechnology company. The price range is between 104.17 and 492.81 USD.

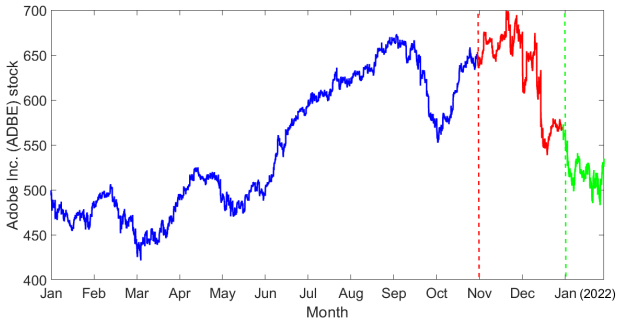
Time series plots of observed price along with training set (blue line), validation set (red line) and test set (green line) are displayed in Figure 5.2.

We then focus on the data pre-processing step before applying it for model training and testing. After obtaining data from the Thomson Reuters database, we clean up any errors and fill in any missing data using a splines interpolation method. The historical financial data series were divided into three sets, namely training, validation, and test sets. For each stock, 19,829 data points from January to December 2021 were used for training and validating the deep learning models. 83 % (January - October 2021) were used for training (learning parameters), and the remaining 17 % (November - December 2021) to validate network performance and avoid overfitting. The remaining one month (January 2022) was used to assess the model performance (hold-out sample).

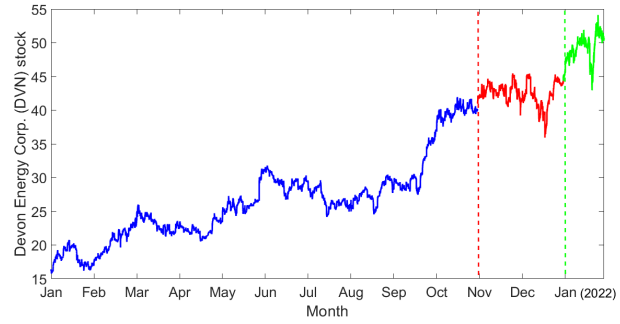
All experiments in this study are performed by using Python programming language, which is running on Google Colaboratory (also known as Google Colab). Finally, the data are normalized using MinMaxScaler. The MinMaxS-



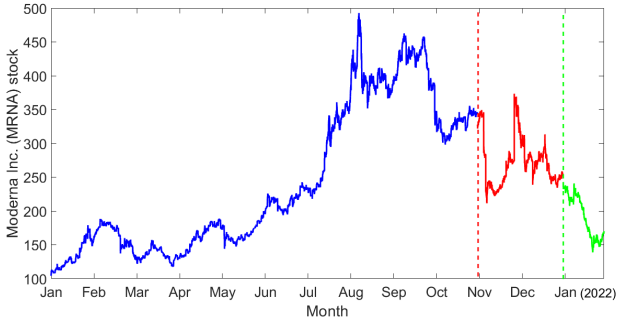
(a) Apple Inc.



(b) Adobe Inc.



(c) Devon Energy Corporation



(d) Moderna Inc.

Figure 5.2: Original AAPL, ADBE, DVN, and MRNA stock prices time series with five minutes from January 2021 to January 2022.

caler formula is as follow [177].

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}, \tag{5.1}$$

where x' is the value in the interval [0,1], x presents the observed price, x_{min}, x_{max} represent the minimum and the maximum of the set of stock price, respectively. The inverse normalization function is

$$\hat{y}_i = y_i \cdot (x_{max} - x_{min}) + x_{min}. \tag{5.2}$$

Here $\hat{y}_i$ is the predicted value and y_i presents the network output value.

5.4 Hybrid Model

Time series prediction has been a challenging task for over five-decade. A large amount of literature has focused on the method to get accurate forecasts in numerous practical applications. In general, two main ways can improve the accuracy of forecasting results: (1) developing and proposing new forecasting models and (2) hybridization of existing forecasting models. In this chapter, we focus on only the hybrid model. The famous hybrid structure consists of the parallel and the series hybrid structures [4]. The framework of two classifications of hybrid structures is shown in Figure 5.3.

From Figure 5.3, the linear combination of parallel structures is a very challenging problem because the main aim of this area is to seek the optimal weight of each forecast. However, we work with delays in time series, and there is no literature on combining different delays for forecasting. Moreover, only very little work in the literature studied the delay (look back) period (see [129,178]). This motivates us to combine two delay forecasts and estimate the weights using the differential algorithm.

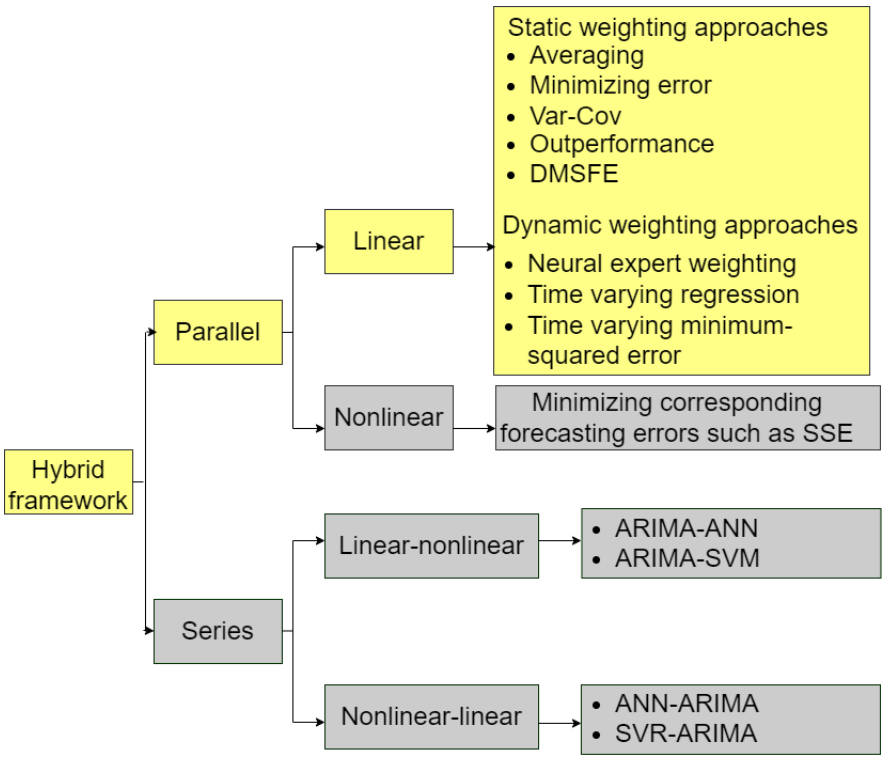


Figure 5.3: The schematic framework of the hybrid process [4]

The delay (m) in this chapter refers to the network's input length (look back). For example, the look back at four means we allow our network to look back at four data to predict the next timestamp. Each prediction model was trained utilizing two different delays predicted values, i.e., 43 (half-day trading delay) and 79 (one-day trading delay). The idea of delay (look back) is displayed in Table 5.2.

Table 5.2: How the first 4 samples for open, high, low, volume and close prices would generate the 5th sample.

Datetime	Open	High	Low	Volume	Close
4/01/2021 9:30:00 AM	222.53	223.00	220.59	1404934	220.72
4/01/2021 9:35:00 AM	220.70	220.75	219.72	731172	219.84
4/01/2021 9:40:00 AM	219.85	220.06	219.28	762643	220.03
4/01/2021 9:45:00 AM	220.02	220.40	219.76	406992	220.05
					220.13

5.4.1 A Two-delay Combination Forecast

Three general deep learning (DL) methods, namely the multilayer perceptron (MLP), the convolutional neural network (CNN) and the long short-term memory (LSTM) network, are applied to predict future stock prices. The linear combination forecast is presented to improve the single DL forecast accuracy. The idea of the combination forecasts was first introduced by Bates, and Granger [164]. Many researchers have studied this topic and built several combination forecast techniques to improve the forecast accuracy (see, for example, [165,179,180]). In this chapter, we proposed the two-delay combination method to predict future closing prices for each stock. Different stock prices are in different ranges, and we expect our proposed model to work well on all of these stocks. The general two-delay combination method can be defined as [181]

$$\hat{y}_c = \sum_{i=1}^{2} w_i \hat{y}_i, \tag{5.3}$$

where $\hat{y}_c$ is the combined forecast, w_1, w_2 are the weights obtained from the DE algorithm, $w_1 + w_2 = 1$, $\hat{y}_i$ is the i^{th} point forecast.

The structure of the linear combination forecast is presented in Figure 5.4.

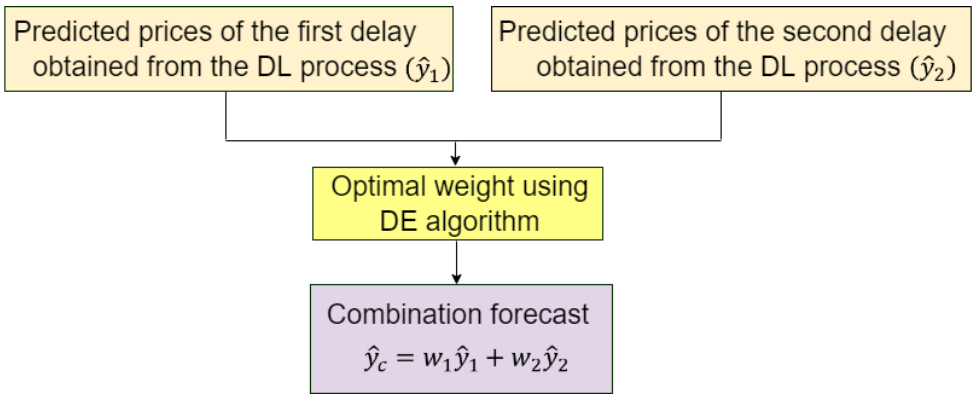


Figure 5.4: The structure of the two-delay linear combination method

This structure starts with importing a predicted price obtained from the

first and the second delay on each DL technique. We then combine the forecast results using the linear combination method. Next, the weights are computed using the DE algorithm. Finally, the evaluation indicators are calculated to compare the forecasting performance obtained by the individual and the combination of forecast techniques. The pseudocode of the DE algorithm is displayed in Algorithm 7.

The Algorithm 7 starts with defining some parameters such as the number of population size (NP), the crossover ratio (CR), the mutation ratio (F), and the constant term (TOL). Then, an initial population is generated using a uniform distribution on the interval [0,1], where x^L and x^U are the lower and upper bounds for the decision parameter, respectively. In the mutation process, three individuals (x_{r1}, x_{r2}, x_{r3}) are selected in the population set of NP elements, which $r1 \neq r2 \neq r3 \neq i$. F is a user-defined constant known as the mutation factor, $F \in [0, 1]$. We then apply all values to obtain the mutant vectors (v_i). The trial vectors (u_i) are generated by mixing the parameters of the target vectors (x_i) with the mutant vectors (v_i) according to a selected crossover probability (CR). The selection scheme is applied in the DE algorithm for the next step. The predicted stock prices of the test set are imported to compute the fitness of the target vectors and the fitness of trial vectors. The best solution is chosen by comparing the fitness of the trial vectors and the corresponding target vectors. We then calculate the predicted stock prices (f_i) of each DL model. Next, the observed prices are imported to compute the error measurement. The errors between the observed and predicted stock prices are calculated. The mean absolute percentage error (MAPE) is applied to seek an optimal weight (x_i). The optimization process stops whenever the MAPE is less than a constant term (TOL) or the number of iterations reaches the limit. Finally, the weight is applied in the hybrid predictive model Eq. (5.3) to obtain the combination forecast. The parameters setting used in this section are NP = 10, F = 0.2, and CR = 0.6 based on Section 4.3.

Algorithm 7 DE Pseudocode for weight identification

Require: $NP = 10$, $CR = 0.6$, $F = 0.2$, $TOL = 0.01$, $x^L = -5$, $x^U = 5$, $MAPE_{\min} = 100$, $maxG = 1000$ and $G = 0$.

Generate the initial population

$x_i^G = x_i^L + rand(0,1) \cdot (x_i^U - x_i^L), \quad i = 1, \ldots, NP$

While ($MAPE_{\min} > TOL$ or $G < maxG$)

for $i = 1$ to NP

Randomly select $r_1, r_2, r_3 \in 1, 2, \ldots, NP$ where $r_1 \neq r_2 \neq r_3 \neq i$

Mutation

$v_i^G = x_{r1}^G + F \cdot (x_{r2}^G - x_{r3}^G)$

Crossover

if $rand(0,1) <= CR$

$u_i^G = v_i^G$

else

$u_i^G = x_i^G$

end if

Selection: $sumE_m = 0, sumE = 0$

Import the predicted stock prices $\left\{\hat{y}_k^{(m_1)}\right\}_{k=1}^n$, $\left\{\hat{y}_k^{(m_2)}\right\}_{k=1}^n$ and the observed stock prices $\{Y_k\}_{k=1}^n$

for $k = 1$ to n

$f_i^m(k) = u_i^G \hat{y}_k^{(m_1)} + (1 - u_i^G)\hat{y}_k^{(m_2)}$

$f_i^m(k) = x_i^G \hat{y}_k^{(m_1)} + (1 - x_i^G)\hat{y}_k^{(m_2)}$

$e^m(k) = \left| f_i^m(k) - Y(k) \right| / Y(k)$

$e(k) = |f_i(k) - Y(k)| / Y(k)$

$sumE_m = sumE_m + e^m(k)$

$sumE = sumE + e(k)$

end for

$MAPE_m = sumE_m \times 100/n$

$MAPE = sumE \times 100/n$

if $MAPE_m < MAPE$

$x_i^{G+1} = u_i^G$

$MAPE_i = MAPE_m$

else

$x_i^{G+1} = x_i^G$

$MAPE_i = MAPE$

end if

end for

$MAPE_{min} = \min(MAPE_i)$

$G = G + 1$

end while

Compute the predicted combination forecast model using the optimal weight

5.4.2 Evaluation Metrics

To evaluate the performance of DL techniques and our proposed method, the statistical criteria, including the mean absolute error (MAE), the mean absolute percentage error (MAPE), and the root mean square percentage error (RMSPE), are applied in this chapter. The method which provides the smaller values of MAE, MAPE, and RMSPE (close to zero) is the better forecasting method. The definition of the evaluation metrics (also called forecast errors) are as follows [20]:

$$RMSPE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left[\frac{y_t - \hat{y}_t}{y_t}\right]^2} \times 100, \tag{5.4}$$

here y_t is the observed price, $\hat{y}_t$ is the predicted price at time t ,and n is the total number of observation.

To evaluate the amount of error reductions, the percentage improvement (PI) is presented. This value is defined as follows:

$$PI = \frac{FE_{best} - FE_c}{FE_{best}} \times 100, \tag{5.5}$$

where FE_{best} is the best single model in terms of forecast error $(MAE, MAPE, RMSPE)$ and FE_c is the forecast error obtained from the combination method.

5.4.3 Hyperparameter Optimization

Deep learning algorithm has several variables, known as hyperparameter. It is challenging for the researcher to choose an optimal hyperparameter before training data. Trial-and-error is one popular technique for choosing the model's variables, but it takes a long time. In this work, SHERPA, a Python library for hyperparameter tuning machine learning models, is applied to select hyperparameter for each model in each stock [182]. Some articles apply the SHERPA algorithm for hyperparameter optimization in neural network fields.

Beucler et al. [183] introduce a systematic way of enforcing nonlinear analytic constraints in neural networks via constraints in the architecture or the loss function. They implement the three NN types and use SHERPA for hyperparameter optimizations in each NN type. Ott et al. [184] introduce a software library, the Fortran-Keras Bridge (FKB), which is used for computing large-scale scientific projects and integrated with modern deep learning methods. SHERPA algorithm is the tool to optimize hyperparameter for each candidate neural network model. Lu et al. [185] proposed a novel deep sparse autoregressive model (SARM) to generate data via a tractable likelihood technique. The hyperparameter of the SARM model, including a number of hidden layers structure, the size of the central area for the two-stage approach and the size of the intermediate upsampling layer, are selected by using the SHERPA algorithm.

Before applying SHERPA to our model, it is a necessary stage to define the parameter search space. For the whole experiment, some parameters are set as follows: the optimizer is stochastic gradient descent (SGD), and the loss function is MSE. In addition, we added L2 regularization with a coefficient of 0.1 to penalize weight parameters for solving the overfitting problem. For the parameter search space used in DL methods, the choice of the activation function is linear, relu, tanh, softmax, and sigmoid, the choice of batch size is 32,64,128, 256 and 512, and the learning rates are fallen between 0.001 and 0.1. The hyperparameter search space of the MLP, LSTM and CNN methods for univariate and multivariate data are shown in Table 5.3.

Table 5.3: Hyperparameter search space of three deep learning models.

Model	Hyperparameter search space values		
	Name	Range	Parameter type
MLP	Hidden unit1	[32,256]	Discrete
	Hidden unit1	[32,256]	Discrete
CNN	Conv1D	[10,32,64,128]	Choice
	kernel size	[1,2,3,4,5]	Choice
	Hidden unit	[32,256]	Discrete
LSTM	LSTM(layer1)	[32,256]	Discrete
	LSTM(layer2)	[32,256]	Discrete

5.5 Empirical Results

The experimental results are divided into three sections. The univariate and multivariate data results are presented in Sections 5.5.1 and 5.5.2, respectively. The comparison of the predicted closing prices obtained from two different data types is shown in Section 5.5.3. In all the experiments, the MLP, CNN and LSTM models were evaluated by running them and observing the plot for the training and validation loss during training. The model compilation is done by calling the python fit() function.

5.5.1 Univariate Data

After the hyperparameter search space are set, the optimal hyperparameter of three stocks obtained from SHERPA algorithm with two delays for univariate time series are represented in Table 5.4.

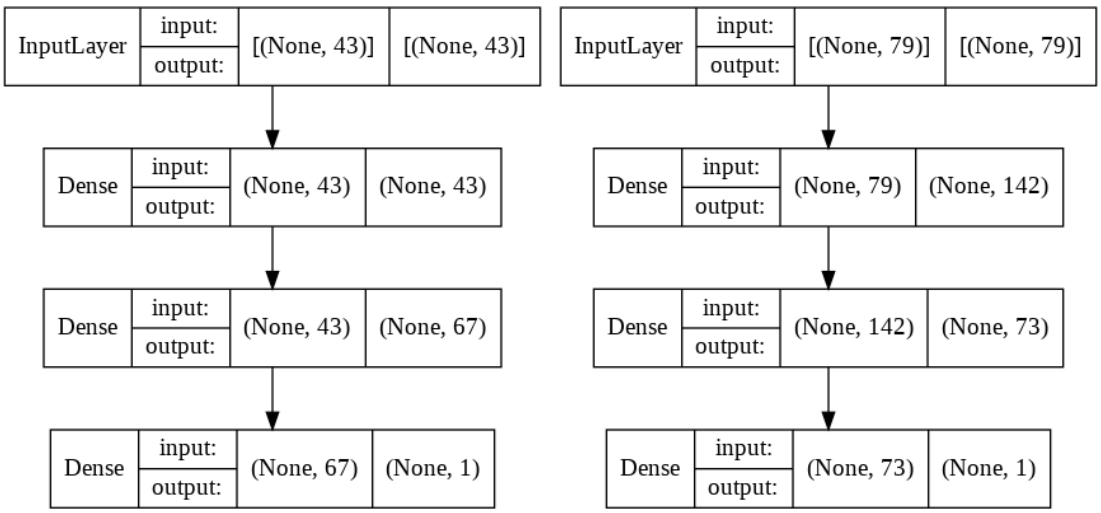
We then applied the optimal hyperparameter into the training set of each stock. The sample MLP, CNN and LSTM architectures used in the training sets of ADBE stock are shown in Figures 5.5 - 5.7. The rest of DL architectures are displayed in Appendix A.1 - A.9.

As shown in Figure 5.5, the details of the design of each layer and the overall architecture of the MLP model are as follows.

For Figure 5.5a, the shape of the input data to the networks input layer is

Table 5.4: Optimal hyperparameter of three deep learning models on the univariate data.

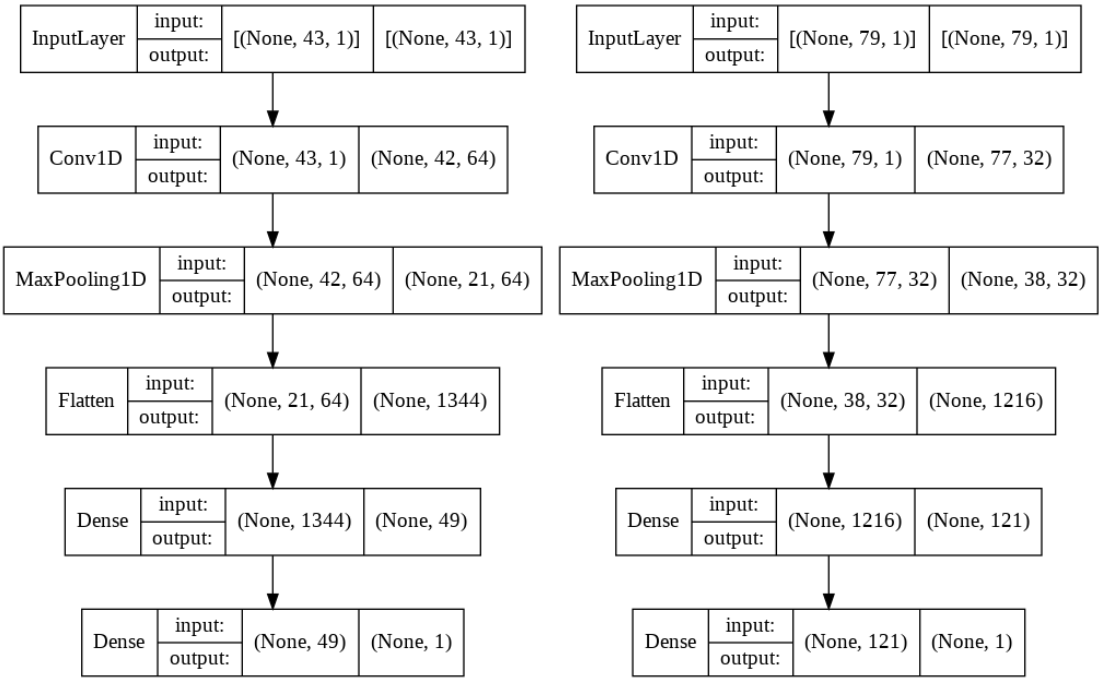
Model	Optimal hyperparameter							
	Half-day trading delay ($m = 43$)				One-day trading delay ($m = 79$)			
	AAPL	ADBE	DVN	MRNA	AAPL	ADBE	DVN	MRNA
MLP								
•Hidden unit1	152	46	43	185	186	102	142	194
•Hidden unit2	211	60	67	209	207	193	73	165
•Activation	tanh	relu	tanh	relu	tanh	relu	relu	relu
•Learning rate	0.001780	0.00390	0.002834	0.002890	0.001185	0.001847	0.004580	0.002876
CNN								
•Conv1D	64	64	32	10	128	32	10	32
•Kernel size	4	2	1	1	4	1	3	1
•Hidden unit1	51	49	120	126	115	121	123	112
•Activation	tanh	relu	tanh	tanh	tanh	tanh	tanh	relu
•Learning rate	0.005496	0.007075	0.006432	0.005296	0.005090	0.009131	0.020399	0.008197
LSTM								
•LSTM(layer1)	236	46	251	194	229	161	59	221
•LSTM(layer2)	146	141	205	248	189	160	228	210
•Activation	tanh	tanh	tanh	tanh	tanh	relu	tanh	relu
•Learning rate	0.001071	0.009802	0.001897	0.001058	0.002735	0.002045	0.002744	0.004583



(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.5: The architecture of the univariate MLP model on ADBE stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)

(None,43), indicating that the previous 43 observations (i.e., half-day trading delay) of the time series are used as the input. This architecture consists of two hidden layers. The first and the second layers have 43 and 67 neurons, respectively. The 67 nodes at the output produce the predicted closing prices for the next timestamp. Figure 5.5b is also a univariate model that uses the previous one-day closing values as the input and yields the forecast for the next timestamp. The description of the model remains identical to those of the MLP model. The only change is the number of neurons in each hidden layer. From figure 5.5b, the number of neurons are 142 and 73 from the first and the second layers, respectively.



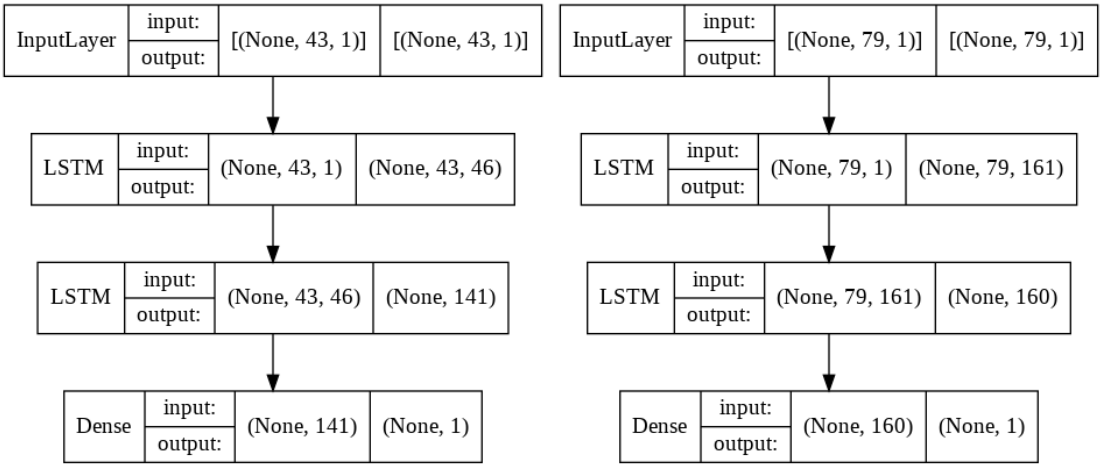
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.6: The architecture of the univariate CNN model on ADBE stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)

The details of the CNN architecture of ADBE stock (Figure 5.6) are as follows. For Figure 5.6a, it indicates that we input only one variable (the closing price), and the previous 43 closing prices (half-day delay) of stock prices time series are considered. The CNN model consists of only one convolution layer that extracts 64 feature maps from the input data with a kernel of size 2. The output shape of the third layer is (21,64) because the max-pooling layer reduces the dimension of the data by a factor of $1/2$ [186]. Next, a flattening operation is used for transformation of the max-pooling layer into one-dimension array ($21 \times 64 = 1344$). After that, the one-dimensional vector is passed through a dense layer and sent to the final output layer. In a one-day trading delay as shown in Figure 5.6b, the CNN architecture process is similar to a half-day trading delay, except the previous one-day closing values (79 observations) are added to the model. The CNN model consists of only one convolution layer that extracts 32 feature maps from the input data with a kernel of size 3. After the max-pooling step, the size of the feature maps to 38 [187]. The output flatten later is (None, 1216) shows that we flatten the output of the convolutional layers to create a one-dimension vector ($38 \times 32 = 1216$). Next, the number of neurons in the dense layer is 121. Finally, the closing prices for the next timestamp are computed using the 121 nodes at the output of the dense layer.

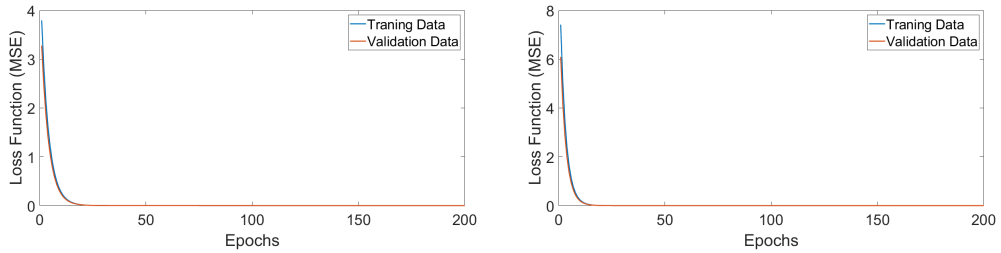
From Figure 5.7a, the shape (43, 1) of the input prices to the network refers to only one feature, i.e., the closing prices of the previous half-day trading ($m = 43$). The first hidden layer of LSTM, having 46 nodes, receives that data from the input layer [188]. The second hidden layer of LSTM consist of 141 nodes. Then, the 141 nodes are passed on to a dense layer. Finally, the output layer containing only one node receives the output of the LSTM layer.

After that, we consider the convergence plot of the loss function. The comparison of the loss function between the training and the validation set of the ADBE stock for all three DL models are displayed in Figures 5.8 - 5.10. The rest of the loss function are presented in Appendix A.10 - A.18.



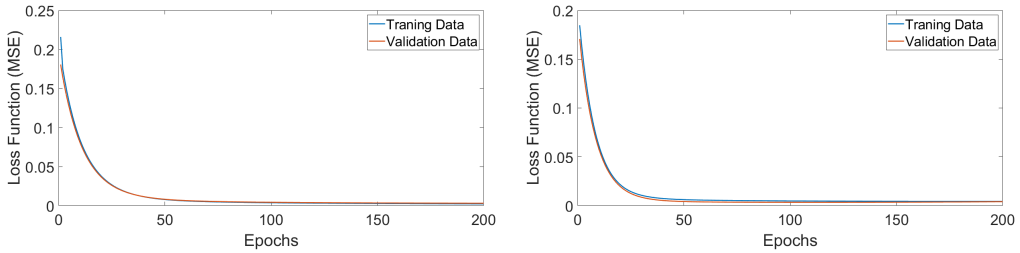
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.7: The architecture of the univariate LSTM model on ADBE stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



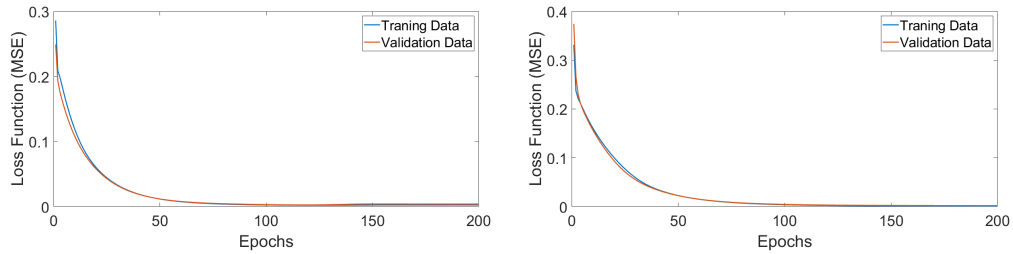
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.8: The convergence plot between the loss function of the training and validation sets while training the MLP for the univariate ADBE stock



(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.9: The convergence plot between the loss function of the training and validation sets while training the CNN for the univariate ADBE stock



(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.10: The convergence plot between the loss function of the training and validation sets while training the LSTM for the univariate ADBE stock

Figures 5.8 - 5.10 show that the training and validation loss decreased and stabilized around the very near points to each other. That means our model architecture can be used for future closing price prediction. We then compute and compare the evaluation metrics between the training and validation sets for other stocks. The MAE, MAPE and RMSPE obtained from the MLP, CNN and LSTM models over the training and validation set are shown in Table 5.5.

From Table 5.5, the MAE, MAPE, and RMSPE obtained from the training and validation sets is almost similar, resulting in further confirmation of no overfitting. This indicates that the performance of all individual DL models behaved well on both the training and validation sets and can be applied to predict out-of-sample data sets.

After receiving the individual forecast obtained from two different delays, the optimal weights are chosen using the DE algorithm (Algorithm 7). We then compute the predicted closing prices obtained from our proposed model and compare them with the individual forecast for each stock. The prediction models and the evaluation metrics of the test set of the MLP, CNN and LSTM models are shown in Tables 5.6 - 5.7, respectively.

The comparison performances of the evaluation metrics obtained from different models on case study are illustrated in Figures 5.11 - 5.14.

Table 5.7 and Figures 5.11 - 5.14 display the evaluation metrics obtained from two individual delays (half-day delay and one-day delay) and two-delay

Table 5.5: MAE, MAPE, and RMSPE of three deep learning models obtained over the training set and the validation set

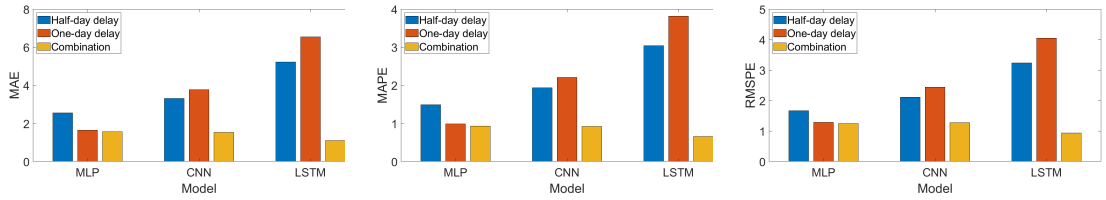
Model	Delay	Stock	Training set			Validation set		
			MAE	MAPE	RMSPE	MAE	MAPE	RMSPE
MLP	43	AAPL	2.6642	1.2805	1.6115	2.1408	1.2551	1.5481
		ADBE	6.9718	1.3692	1.6414	6.2282	1.0016	1.4172
		DVN	0.8611	3.6660	4.3484	0.4859	1.1470	1.5396
		MRNA	9.3266	5.1559	6.3266	6.9875	2.6181	3.8597
	79	AAPL	1.0059	0.7586	1.0121	1.5939	0.9463	1.1987
		ADBE	9.6217	1.8748	2.1904	8.3746	1.3695	1.9541
		DVN	1.1071	4.6694	5.4514	0.8003	1.8750	2.2757
		MRNA	14.9244	7.4049	8.6824	11.4377	4.3190	5.6957
CNN	43	AAPL	1.5561	1.1926	1.5132	2.8286	1.6648	1.9627
		ADBE	10.3001	2.0613	2.5233	8.5650	1.0723	1.5543
		DVN	1.5597	4.7325	4.8005	0.9850	2.3019	2.5770
		MRNA	10.4116	5.8310	7.2900	7.9378	2.9897	4.2478
	79	AAPL	2.4175	1.8634	2.3445	3.2890	1.9270	2.3223
		ADBE	15.0778	2.9025	3.1997	12.7104	2.0649	2.5963
		DVN	2.7887	10.2477	10.6485	2.6076	6.1728	6.4843
		MRNA	15.8038	8.9881	11.2649	12.3861	5.6948	8.1689
LSTM	43	AAPL	2.9796	2.3046	2.8790	4.1680	2.4328	2.9260
		ADBE	15.2277	3.0024	3.6045	13.1939	2.0407	2.3060
		DVN	1.5866	4.6532	5.7066	1.4976	3.5011	3.6627
		MRNA	7.5097	4.0692	5.0593	6.4402	2.4090	3.6164
	79	AAPL	5.4446	3.4379	4.2511	5.2228	3.0407	3.6875
		ADBE	13.5212	2.7037	3.2645	7.8086	1.2880	1.8136
		DVN	1.0278	4.3750	5.1417	0.6272	1.4690	1.8185
		MRNA	9.0678	5.0472	6.3214	7.2767	2.7381	3.9050

Table 5.6: Predictive model of the two-delay combination model on the univariate stock price

Model	Stock	Two-delay combination model
MLP	AAPL	$\hat{y}_c = -0.4077\hat{y}_{(m=43)} + 1.4077\hat{y}_{(m=79)}$
	ADBE	$\hat{y}_c = 3.5225\hat{y}_{(m=43)} - 2.5225\hat{y}_{(m=79)}$
	DVN	$\hat{y}_c = 1.3229\hat{y}_{(m=43)} - 0.3229\hat{y}_{(m=79)}$
	MRNA	$\hat{y}_c = 2.6822\hat{y}_{(m=43)} - 1.6822\hat{y}_{(m=79)}$
CNN	AAPL	$\hat{y}_c = 2.5787\hat{y}_{(m=43)} - 1.5787\hat{y}_{m=79)}$
	ADBE	$\hat{y}_c = 3.5091\hat{y}_{(m=43)} - 2.5091\hat{y}_{(m=79)}$
	DVN	$\hat{y}_c = 0.5036\hat{y}_{(m=43)} + 0.4964\hat{y}_{(m=79)}$
	MRNA	$\hat{y}_c = 3.0584\hat{y}_{(m=43)} - 2.0584\hat{y}_{(m=79)}$
LSTM	AAPL	$\hat{y}_c = 4.6489\hat{y}_{(m=43)} - 3.6489\hat{y}_{(m=79)}$
	ADBE	$\hat{y}_c = 4.8893\hat{y}_{(m=43)} - 3.8893\hat{y}_{(m=79)}$
	DVN	$\hat{y}_c = -1.3912\hat{y}_{(m=43)} + 2.3912\hat{y}_{(m=79)}$
	MRNA	$\hat{y}_c = 4.1401\hat{y}_{(m=43)} - 3.1401\hat{y}_{(m=79)}$

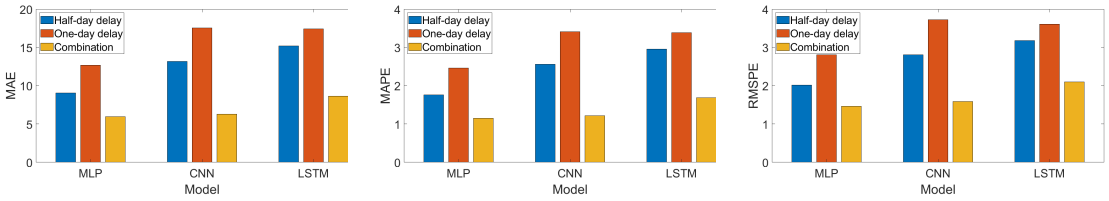
Table 5.7: The comparison of the evaluation metrics together with the percentage improvement over the best single model of the test set on the univariate data sets

Model	Stock	MAE				MAPE				RMSPE			
		$m = 43$	$m = 79$	Hybrid	PI (%)	$m = 43$	$m = 79$	Hybrid	PI (%)	$m = 43$	$m = 79$	Hybrid	PI (%)
MLP	AAPL	2.5407	1.6185	***1.5554***	3.90	1.4870	0.9571	***0.9244***	3.42	1.6598	1.2814	***1.2455***	2.80
	ADBE	9.0591	12.6891	***5.9506***	34.31	1.7588	2.4626	***1.1525***	34.47	2.0146	2.8099	***1.4649***	27.29
	DVN	0.9195	2.0318	***0.6265***	31.86	1.8604	4.0847	***1.2760***	31.41	2.1925	4.3846	***1.6902***	22.91
	MRNA	10.5215	15.3308	***3.7782***	64.09	5.8211	8.4239	***2.1234***	63.52	6.4104	9.0157	***2.8679***	55.26
CNN	AAPL	3.3066	3.7723	***1.5325***	53.65	1.9348	2.2046	***0.9160***	52.66	2.1101	2.4343	***1.2746***	39.59
	ADBE	13.1851	17.5634	***6.2687***	52.46	2.5595	3.4069	***1.2141***	52.56	2.8093	3.7271	***1.5869***	43.51
	DVN	2.0325	2.8229	***0.9443***	53.54	4.0995	4.9415	***1.3255***	67.67	4.3008	5.3660	***1.8648***	56.64
	MRNA	12.3283	19.4479	***4.2438***	65.58	6.8228	10.7736	***2.3195***	66.00	7.4497	11.6383	***2.9843***	59.94
LSTM	AAPL	5.2144	6.5450	***1.0964***	78.97	3.0424	3.8160	***0.6552***	78.46	3.2337	4.0515	***0.9357***	71.06
	ADBE	15.2090	17.4345	***8.6542***	43.10	2.9567	3.3823	***1.6891***	42.87	3.1786	3.6055	***2.0975***	34.01
	DVN	2.6990	1.6363	***0.6634***	59.46	5.4421	3.2923	***1.3620***	58.63	5.5681	3.5132	***1.9846***	43.51
	MRNA	8.7494	10.9644	***3.5757***	59.13	4.8098	5.9085	***1.9157***	60.17	5.3198	6.4621	***2.4313***	54.30



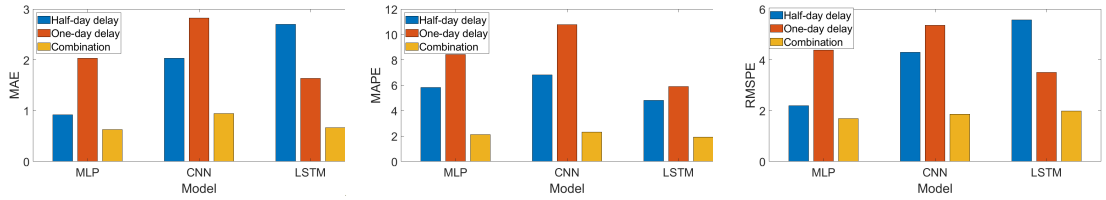
(a) (b) (c)

Figure 5.11: The evaluation metrics of forecasting results obtained from different models on AAPL stock: (a) MAE; (b) MAPE; (c) RMSPE



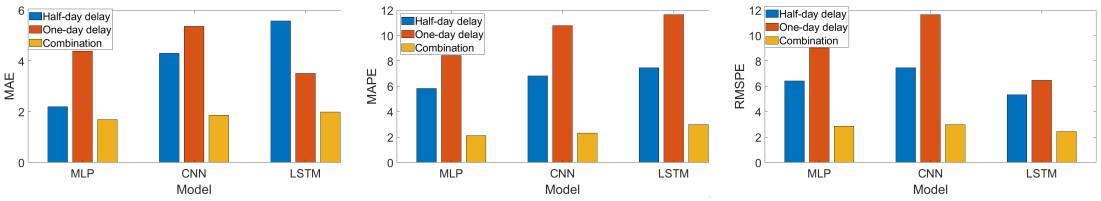
(a) (b) (c)

Figure 5.12: The evaluation metrics of forecasting results obtained from different models on ADBE stock: (a) MAE; (b) MAPE; (c) RMSPE



(a) (b) (c)

Figure 5.13: The evaluation metrics of forecasting results obtained from different models on DVN stock: (a) MAE; (b) MAPE; (c) RMSPE

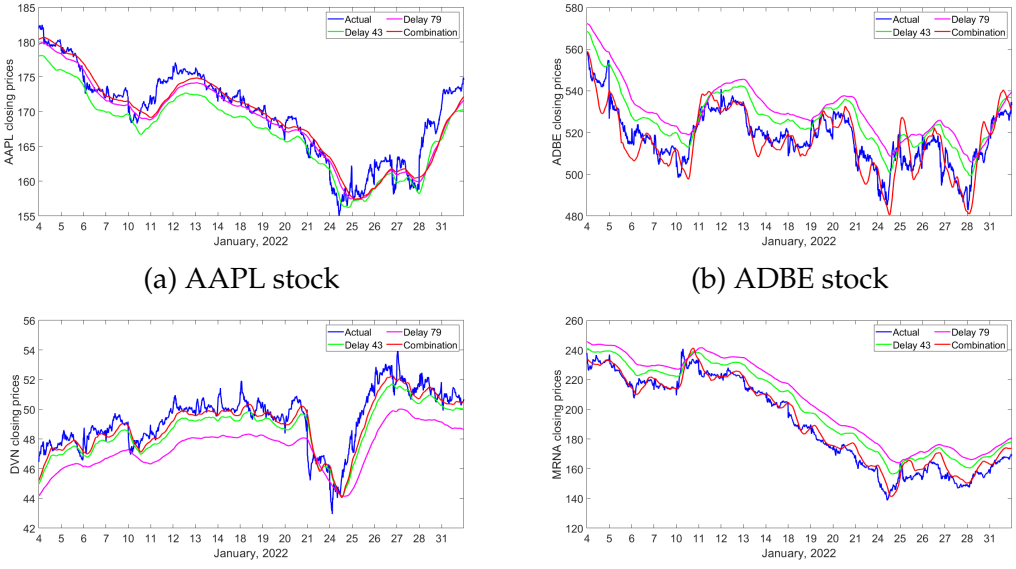


(a) (b) (c)

Figure 5.14: The evaluation metrics of forecasting results obtained from different models on MRNA stock: (a) MAE; (b) MAPE; (c) RMSPE

combination forecasting models on four stocks with each deep learning procedure. It found that the combination model between half-day delay and one-day delay provided the lowest evaluation metrics for all deep learning models and stocks.

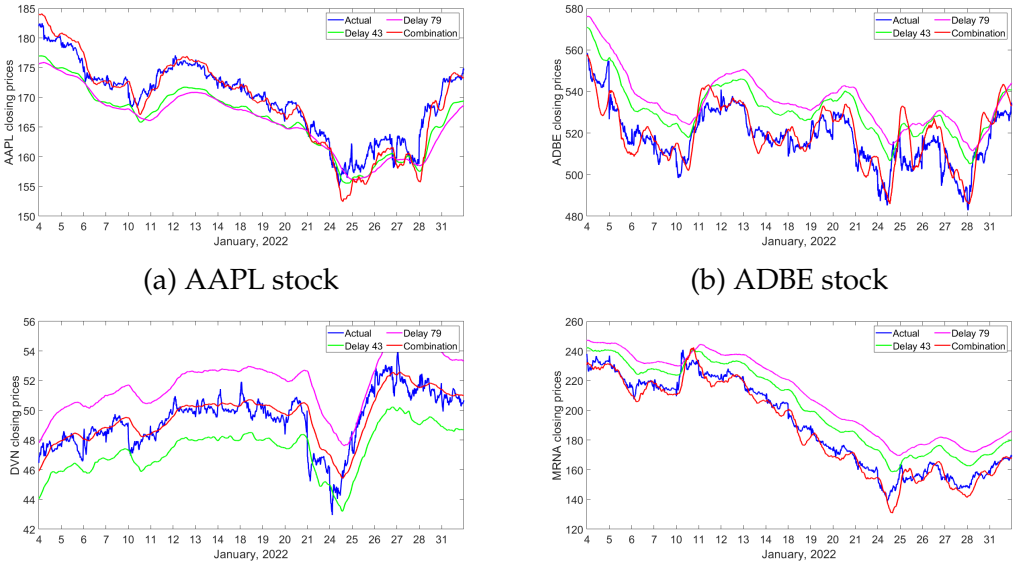
The comparison of the prediction curves between the individual deep learning model and the combination model and its actual price of four stocks obtained from the MLP, CNN and LSTM models are illustrated in Figures 5.15 - 5.17. respectively.



(a) AAPL stock (b) ADBE stock

(c) DVN stock (d) MRNA stock

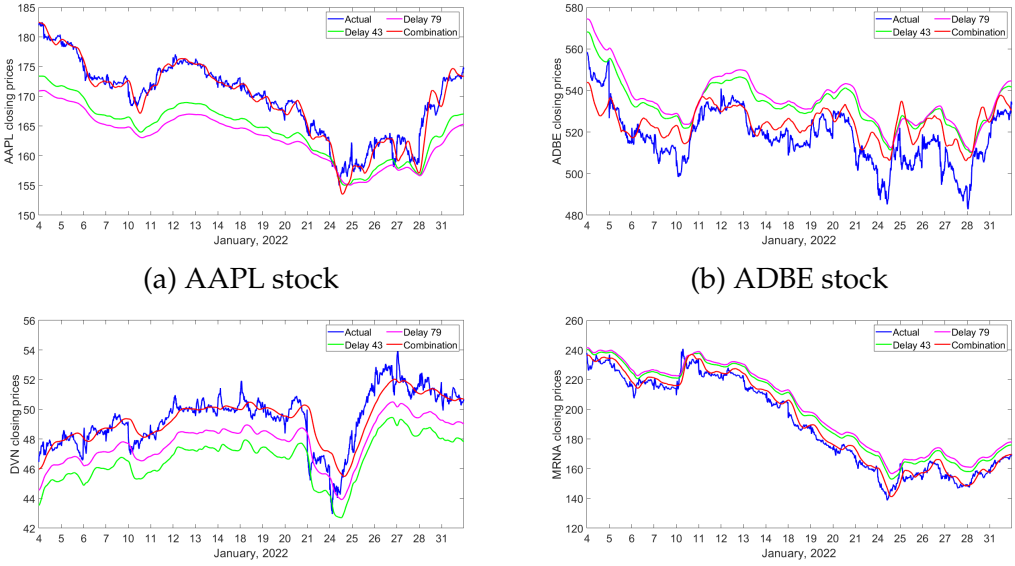
Figure 5.15: The forecasting curve for actual data versus the individual and combination methods on four stock obtained from MLP model.



(a) AAPL stock (b) ADBE stock

(c) DVN stock (d) MRNA stock

Figure 5.16: The forecasting curve for Actual data versus the individual and combination methods on four stock obtained from CNN model.



(a) AAPL stock (b) ADBE stock

(c) DVN stock (d) MRNA stock

Figure 5.17: The forecasting curve for Actual data versus the individual and combination methods on four stock obtained from LSTM model.

Figures 5.15 - 5.17 present the observed univariate stock prices compared to the predicted prices obtained from the individual delay and combination models of deep learning techniques in January 2022. For all Figures, the blue lines display the actual stock prices, and the green and pink lines represent the predicted closing prices obtained from half-day and one-day delays, respectively. The red lines show the predicted closing prices obtained from the two-delay combination models. The predicted stock prices obtained from the half-day delay, one-day delay and two-delay combination predictive models show similar stock price patterns to the actual stock prices. It is obviously seen that our proposed combination model performed better than the individual forecast for all stocks and deep learning models because our proposed model provides the lowest evaluation metrics for all stocks. .

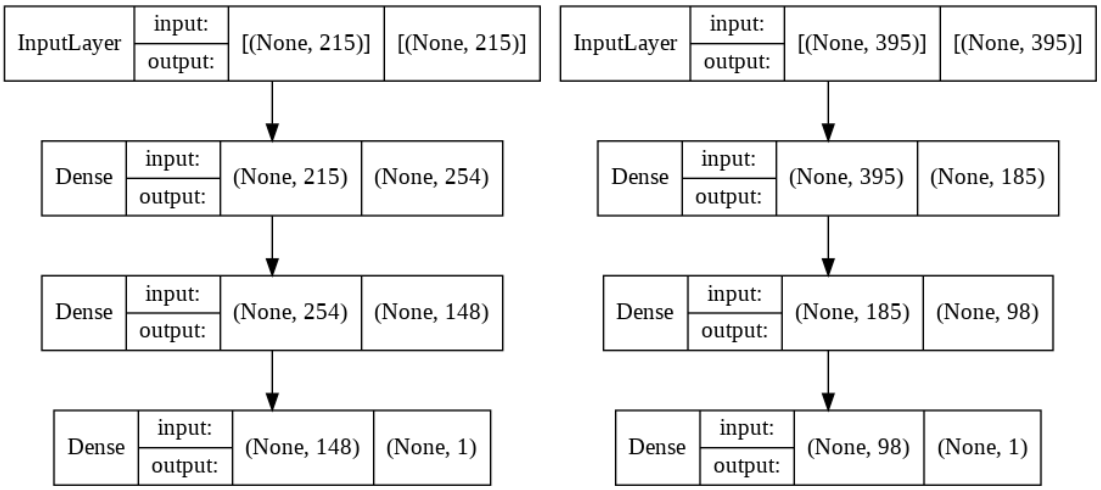
5.5.2 Multivariate Data

The optimal hyperparameter of four stocks obtained from SHERPA algorithm with two delays for the multivariate time series are represented in Table 5.8.

Table 5.8: Optimal hyperparameter of three deep learning models on the multivariate data.

Model	Optimal hyperparameter							
	Half-day trading delay ($m = 43$)				One-day trading delay ($m = 79$)			
	AAPL	ADBE	DVN	MRNA	AAPL	ADBE	DVN	MRNA
MLP								
•Hidden unit1	42	254	187	225	126	185	234	103
•Hidden unit2	58	148	204	128	161	98	209	247
•Activation	tanh	tanh	tanh	tanh	tanh	tanh	tanh	tanh
•Learning rate	0.004617	0.006607	0.005925	0.003161	0.001894	0.001717	0.001742	0.003559
CNN								
•Conv1D	64	10	32	128	10	32	10	32
•Kernel size	5	3	5	5	1	5	5	2
•Hidden unit1	73	63	33	35	75	102	42	114
•Activation	tanh	relu	relu	relu	tanh	relu	relu	relu
•Learning rate	0.006220	0.005527	0.006556	0.017122	0.013092	0.005403	0.008067	0.021873
LSTM								
•LSTM(layer1)	47	190	181	206	246	181	224	84
•LSTM(layer2)	87	214	69	83	167	245	116	250
•Activation	relu	tanh	relu	relu	relu	relu	relu	tanh
•Learning rate	0.002249	0.001660	0.003344	0.006018	0.004311	0.003225	0.009433	0.004824

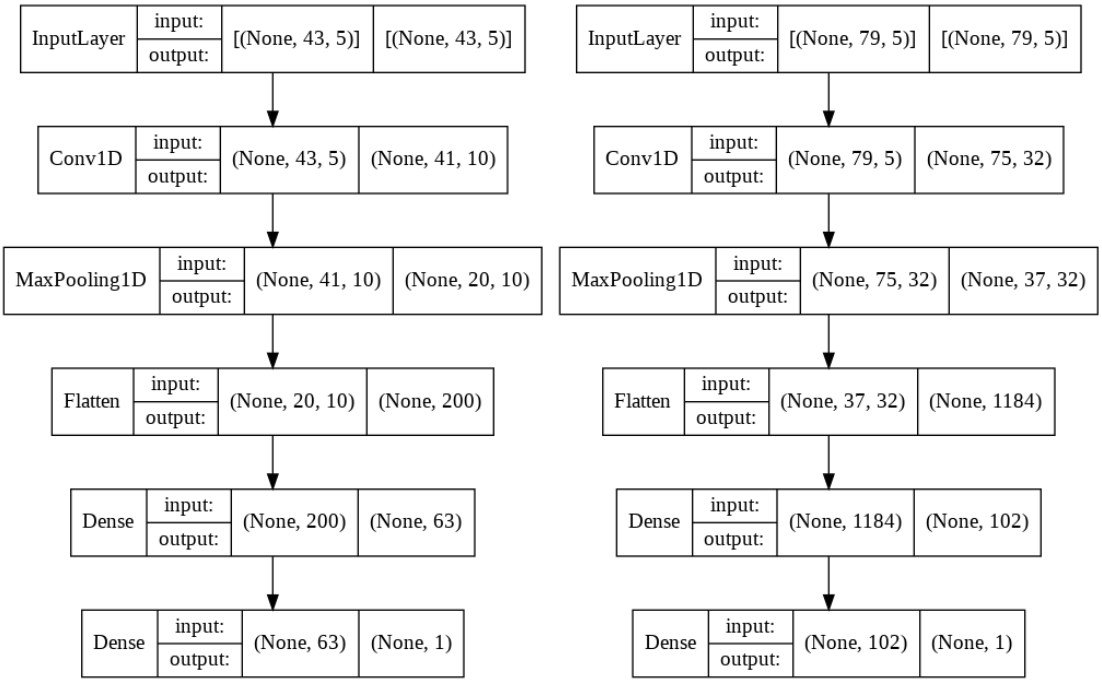
We then applied the optimal hyperparameter into the training set of each stock. The sample network architecture of the ADBE stock used in the training sets with three DL models are shown in Figures 5.18 - 5.20. The rest of the multivariate DL architectures are presented in Appendix. A.19 - A.27.



(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.18: The architecture of the multivariate MLP model on ADBE stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)

The details of the MLP model for the multivariate data are as follows. For Figure 5.18a, the shape of the input data to the network's input layer is (None,215), indicating that the previous 215 observations (i.e., half-day trading delay of OHLCV values; $43 \times 5 = 215$) of the time series are used as the input. This MLP model architecture consists of two hidden layers. The first and the second layers have 254 and 148 neurons, respectively. The 148 nodes at the output produce the predicted closing prices for the next timestamp. In Figure 5.18b, the shape of the input layer is (None,395) refers to the previous one-day closing values of OHLCV values ($79 \times 5 = 395$) are used as the input and yields the forecast for the next timestamp. The details of the MLP model, as shown in Figure 5.18b are similar with Figure 5.18a. The only change is the number of neurons in each hidden layer which are 185 and 98 from the first and the second layers, respectively.

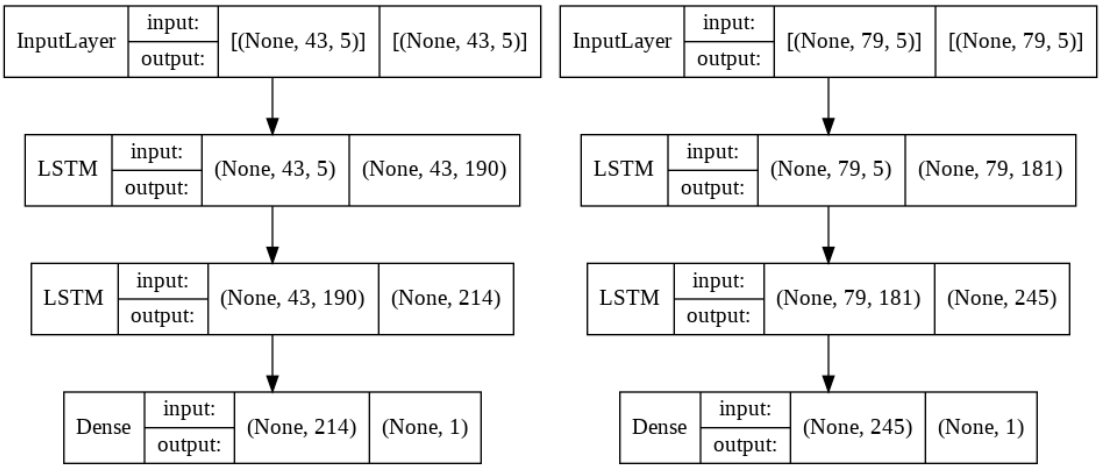


(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.19: The architecture of the multivariate CNN model on ADBE stock

The details of the CNN architecture of ADBE stock (Figure 5.19) are as follows. For half-day trading delay (Figure 5.19a), the first layer is the input layer which contains the shape (43,5). It indicates that we use the multivariate input, including the opening, highest, lowest, volume, and closing prices of the previous 43's stock price records. The CNN model consists of only one convolution layer that extracts 10 feature maps from the input data with a kernel of size 3. The output shape of the third layer is (20,10) because the max-pooling layer reduces the dimension of the data by a factor of $1/2$ [186]. Next, a flattening operation is used for transformation of the max-pooling layer into one-dimension array ($20 \times 10 = 200$). After that, the one-dimensional vector is passed through a dense layer and sent to the final output layer. In a one-day trading delay as shown in Figure 5.19b, the CNN architecture process is similar to a half-day trading delay, except the previous one-day OHLCV values (79 observations) are added to the model. The CNN model consists of only one convolution

layer that extracts 32 feature maps from the input data with a kernel of size 5. After the max-pooling step, the size of the feature maps to 37 [187]. The output flatten later is (None, 1184) shows that we flatten the output of the convolutional layers to create a one-dimension vector ($37 \times 32 = 1184$). Next, the number of neurons in the dense layer is 102. Finally, the closing prices for the next timestamp are computed using the 102 nodes at the output of the dense layer.



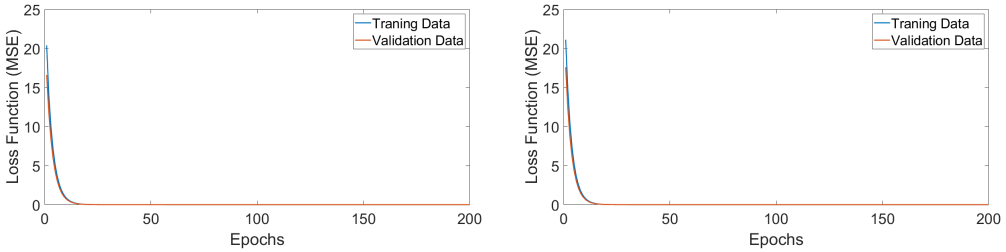
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.20: The architecture of the multivariate LSTM model on ADBE stock

From Figure 5.20a, the shape (43, 5) of the input prices to the network refers to the five attributes of the previous half-day trading ($m = 43$). The first hidden layer of LSTM, having 190 nodes, receives that data from the input layer [188]. The second hidden layer of LSTM has 214 nodes. The 214 modes of the LSTM block are passed on to a dense layer. Finally, the output layer containing only one node receives the output of the LSTM layer. For one-day delay, the details of LSTM model architecture from Figure 5.20b are similar to half-delay delay, except the number of nodes obtained from two LSTM layers are 181 and 245, respectively.

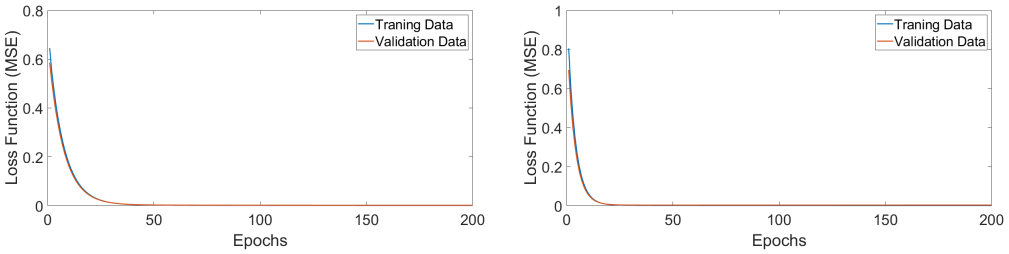
The sample convergence plot of the loss function obtained from the training and validation sets of the ADBE stock on the MLP, CNN and LSTM models are

presented in Figures 5.21 - 5.23. The rest of the convergence plots are revealed in Appendix A.28 - A.36.



(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.21: The convergence plot between the loss function of the training and validation sets while training the MLP for the multivariate ADBE stock



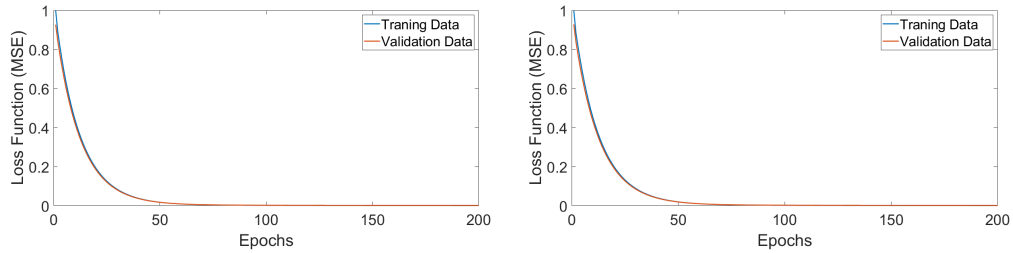
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.22: The convergence plot between the loss function of the training and validation sets while training the CNN for the multivariate ADBE stock

From Figures 5.21 - 5.23, it can be observed that the training and validation loss decreased and stabilized around the very near points to each other. That means our model architecture can be used for future closing price prediction. We then compute and compare the evaluation metrics between the training set and the validation set for others stocks.

The MAE, MAPE and RMSPE obtained from the MLP, CNN and LSTM models over the training and validation set are shown in Table 5.9.

From Table 5.9, the MAE, MAPE, and RMSPE obtained from the training and validation sets are almost similar, resulting in further confirmation of no overfitting. This indicates that the performance of all individual DL models behaved well on both the training and validation sets and can be applied to



(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure 5.23: The convergence plot between the loss function of the training and validation sets while training the LSTM for the multivariate ADBE stock

Table 5.9: MAE, MAPE, and RMSPE of three deep learning models obtained over the training set and the validation set

Model	Delay	Stock	Training set			Validation set		
			MAE	MAPE	RMSPE	MAE	MAPE	RMSPE
MLP	43	AAPL	1.6292	1.2407	1.4723	1.1946	0.7109	0.8944
		ADBE	6.3935	1.2551	1.5151	6.1042	0.9885	1.3823
		DVN	0.4260	1.7766	2.1626	0.3346	0.7939	1.1568
		MRNA	5.8028	3.0710	3.8759	5.5952	2.0704	3.3281
	79	AAPL	0.9398	0.7117	0.9412	1.5616	0.9239	1.1548
		ADBE	7.4241	1.4504	1.7446	7.0317	1.1403	1.6815
		DVN	0.5800	2.4086	2.9690	0.5370	1.2753	1.7420
		MRNA	8.5199	3.8759	4.8572	8.0658	2.9838	4.5557
CNN	43	AAPL	0.7880	0.5870	0.7797	1.7982	1.0672	1.2924
		ADBE	6.5078	1.2634	1.4940	5.8432	0.9488	1.41888
		DVN	0.5083	2.1211	2.6344	0.5001	1.1825	1.6013
		MRNA	7.4046	3.2285	3.8506	5.9755	2.1690	3.6129
	79	AAPL	1.7621	1.3512	1.7176	3.0668	1.8026	2.1406
		ADBE	13.4431	2.5963	2.9198	10.7105	1.7673	2.4494
		DVN	1.7910	7.5129	8.4227	1.1522	6.6814	8.0634
		MRNA	11.3233	5.4668	6.4662	10.3681	3.8775	5.4577
LSTM	43	AAPL	2.4918	1.9251	2.3980	4.0215	2.3424	2.8521
		ADBE	7.1833	1.4308	1.7594	5.1997	0.8450	1.2648
		DVN	1.1940	5.0314	5.7046	0.6037	4.4101	4.7465
		MRNA	5.5966	2.8183	3.4791	5.1013	1.8916	3.0501
	79	AAPL	5.2895	3.3160	4.0755	5.0073	2.9092	3.5733
		ADBE	7.6241	1.5088	1.8197	5.7147	0.9309	1.3635
		DVN	0.8806	3.7211	4.3243	0.5029	3.1789	3.5260
		MRNA	7.7340	4.2320	5.3137	6.5452	2.4476	3.6660

predict out-of-sample data sets.

After receiving the individual forecast obtained from two different delays, the optimal weights are chosen using the DE algorithm (Algorithm 7). We then compute the predicted closing prices obtained from our proposed model and compare them with the individual forecast for each stock. The prediction models and the evaluation metrics of the test set of the MLP, CNN and LSTM models are shown in Tables 5.10 - 5.11, respectively.

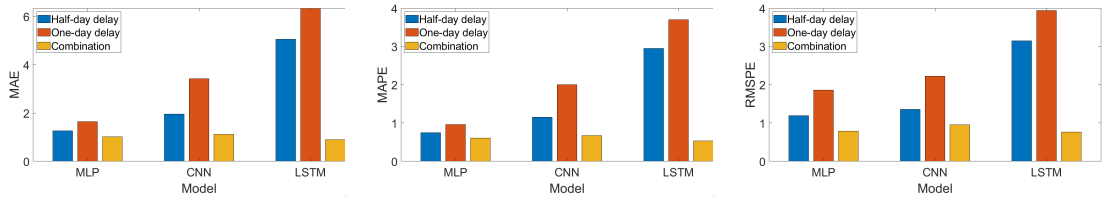
Table 5.10: Predictive model of the two-delay combination model on multivariate stock price

Model	Stock	Two-delay combination model
MLP	AAPL	$\hat{y}_c = 2.3465\hat{y}_{(m=43)} - 1.3465\hat{y}_{(m=79)}$
	ADBE	$\hat{y}_c = 2.6474\hat{y}_{(m=43)} - 1.6474\hat{y}_{(m=79)}$
	DVN	$\hat{y}_c = 1.8037\hat{y}_{(m=43)} - 0.8037\hat{y}_{(m=79)}$
	MRNA	$\hat{y}_c = 2.9039\hat{y}_{(m=43)} - 1.9039\hat{y}_{(m=79)}$
CNN	AAPL	$\hat{y}_c = 1.9521\hat{y}_{(m=43)} - 0.9521\hat{y}_{m=79)}$
	ADBE	$\hat{y}_c = 1.8338\hat{y}_{(m=43)} - 0.8338\hat{y}_{(m=79)}$
	DVN	$\hat{y}_c = 1.2242\hat{y}_{(m=43)} - 0.2242\hat{y}_{(m=79)}$
	MRNA	$\hat{y}_c = 1.8835\hat{y}_{(m=43)} - 0.8835\hat{y}_{(m=79)}$
LSTM	AAPL	$\hat{y}_c = 4.7755\hat{y}_{(m=43)} - 3.7755\hat{y}_{(m=79)}$
	ADBE	$\hat{y}_c = 4.9444\hat{y}_{(m=43)} - 3.9444\hat{y}_{(m=79)}$
	DVN	$\hat{y}_c = -2.5519\hat{y}_{(m=43)} + 3.5519\hat{y}_{(m=79)}$
	MRNA	$\hat{y}_c = 2.5827\hat{y}_{(m=43)} - 1.5827\hat{y}_{(m=79)}$

Table 5.11: The comparison of the evaluation metrics together with the percentage improvement over the best single model of the test set

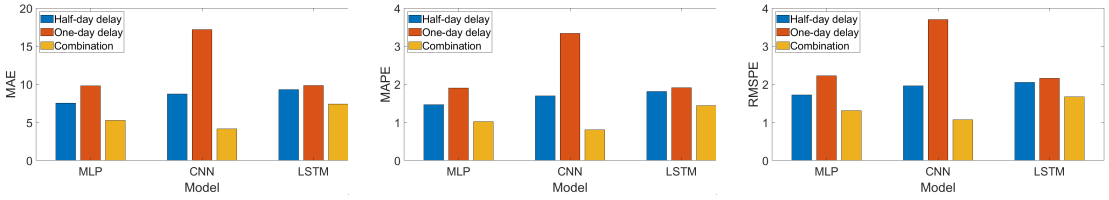
Model	Stock	MAE				MAPE				RMSPE			
		$m = 43$	$m = 79$	Hybrid	PI (%)	$m = 43$	$m = 79$	Hybrid	PI (%)	$m = 43$	$m = 79$	Hybrid	PI (%)
MLP	AAPL	1.2684	1.6373	***1.0156***	19.93	0.7481	0.9645	***0.6045***	19.20	1.1953	1.8632	***0.7887***	34.02
	ADBE	7.5393	9.7927	***5.2741***	30.04	1.4641	1.9023	***1.0219***	30.20	1.7173	2.2267	***1.3089***	23.78
	DVN	0.5055	0.7669	***0.4166***	17.59	1.0283	1.5620	***0.8525***	17.09	1.3940	2.0659	***1.1678***	16.22
	MRNA	6.7714	9.1518	***3.4976***	48.35	3.7159	5.0079	***1.9269***	48.14	4.2414	5.6939	***2.5957***	38.80
CNN	AAPL	1.9610	3.4172	***1.1243***	42.67	1.1532	1.9987	***0.9548***	17.20	1.3544	2.2223	***0.6717***	50.41
	ADBE	8.7128	17.1952	***4.1346***	52.55	1.6913	3.3344	***1.0678***	36.87	1.9562	3.7015	***0.8022***	58.99
	DVN	1.2684	2.7751	***0.8453***	33.36	0.7481	1.6195	***0.5052***	32.47	0.9398	1.8632	***0.7281***	22.53
	MRNA	6.2193	12.2030	***3.0463***	51.02	3.4024	6.6567	***1.6657***	51.04	4.0348	7.3771	***2.2992***	43.01
LSTM	AAPL	5.0608	6.3435	***0.8972***	82.27	2.9497	3.6952	***0.5348***	81.87	3.1466	3.9348	***0.7657***	75.66
	ADBE	9.3189	9.8247	***7.4119***	20.46	1.8091	1.9069	***1.4397***	20.42	2.0468	2.1542	***1.6739***	18.22
	DVN	2.0491	1.5069	***0.4838***	67.89	4.1179	3.0311	***0.9884***	67.39	4.3010	3.2306	***1.4101***	56.35
	MRNA	5.9994	8.5993	***2.4369***	59.38	3.2743	4.9374	***1.3058***	60.12	3.7386	5.4732	***1.7367***	53.55

The comparison performances of the evaluation metrics obtained from different models on multivariate case study are illustrated in Figures 5.24 - 5.27.



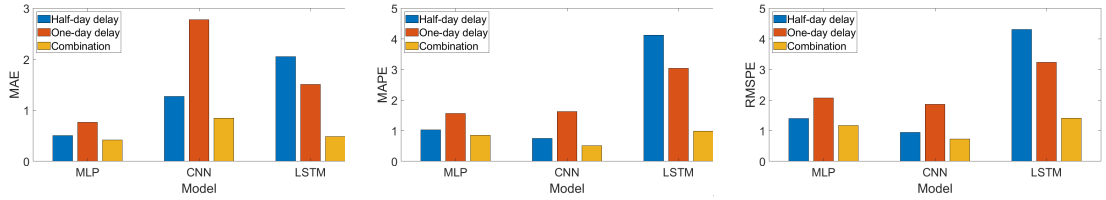
(a) (b) (c)

Figure 5.24: The evaluation metrics of forecasting results obtained from different models on multivariate AAPL stock: (a) MAE; (b) MAPE; (c) RMSPE



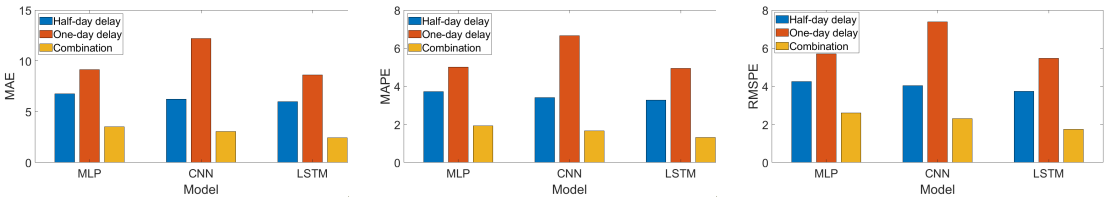
(a) (b) (c)

Figure 5.25: The evaluation metrics of forecasting results obtained from different models on multivariate ADBE stock: (a) MAE; (b) MAPE; (c) RMSPE



(a) (b) (c)

Figure 5.26: The evaluation metrics of forecasting results obtained from different models on multivariate DVN stock: (a) MAE; (b) MAPE; (c) RMSPE

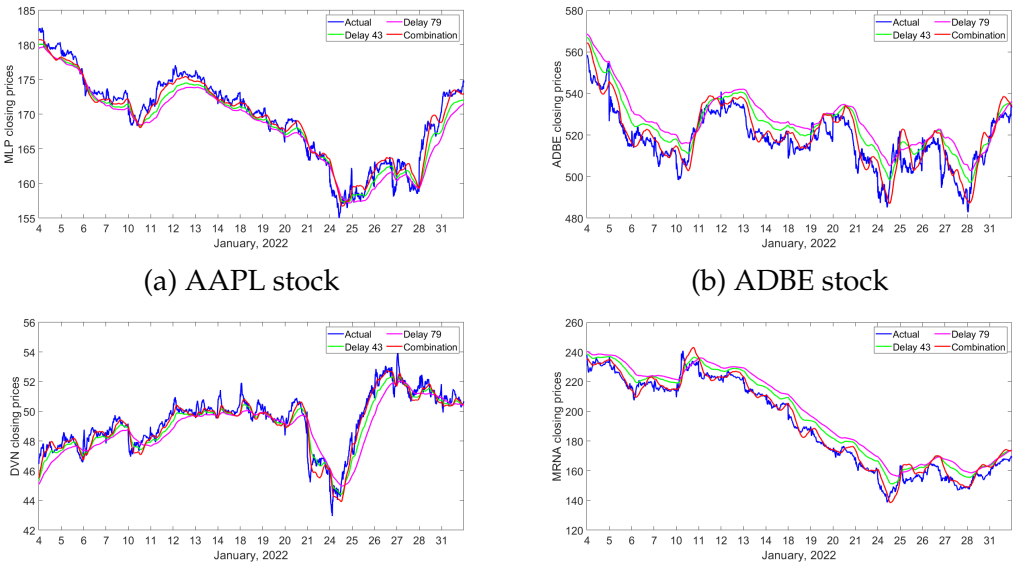


(a) (b) (c)

Figure 5.27: The evaluation metrics of forecasting results obtained from different models on multivariate MRNA stock: (a) MAE; (b) MAPE; (c) RMSPE

Table 5.11 and Figures 5.24 - 5.27 reveal the MAE, MAPE, and RMSPE of two individual delays (half-day delay and one-day delay) and two-delay combination forecasting models of the MLP, CNN and LSTM models. It found that the combination model between half-day delay and one-day delay provided the lowest all evaluation metrics for all deep learning models and stocks.

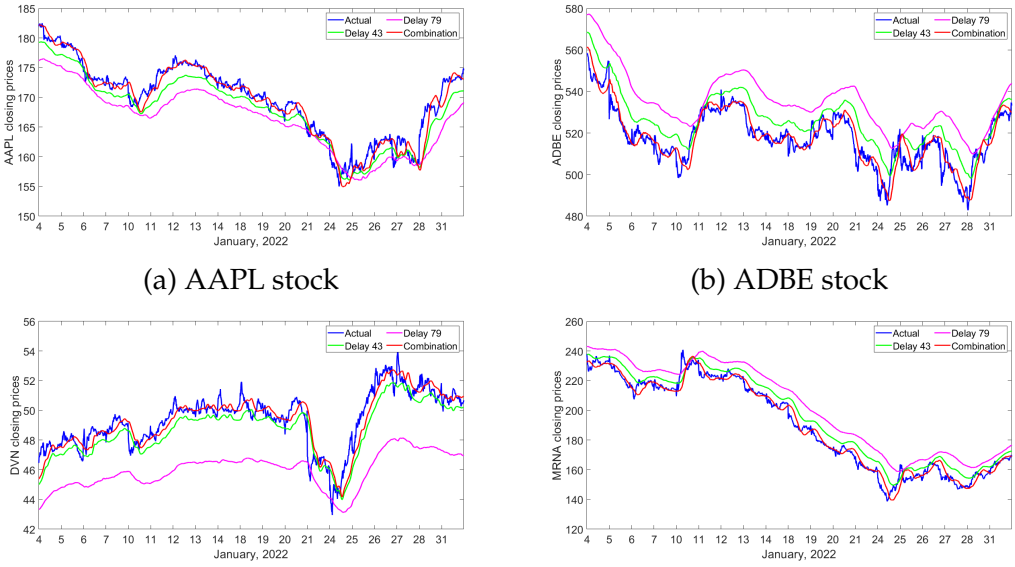
The comparison of the prediction curves between the individual deep learning model and the combination model and its actual price curve of four stocks obtained from the MLP, CNN and LSTM model are illustrated in Figures 5.28 - 5.30. respectively.



(a) AAPL stock (b) ADBE stock

(c) DVN stock (d) MRNA stock

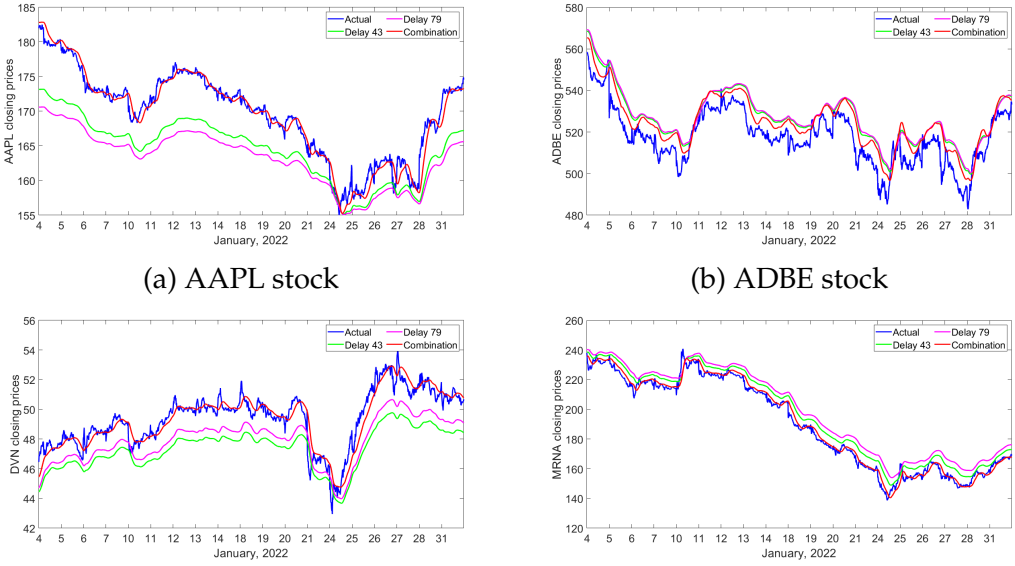
Figure 5.28: The forecasting curve of actual data versus the individual and combination methods on four stocks obtained from MLP model.



(a) AAPL stock (b) ADBE stock

(c) DVN stock (d) MRNA stock

Figure 5.29: The forecasting curve of actual data versus the individual and combination methods on four stocks obtained from CNN model.



(a) AAPL stock (b) ADBE stock

(c) DVN stock (d) MRNA stock

Figure 5.30: The forecasting curve of actual data versus the individual and combination methods on four stocks obtained from LSTM model.

Figures 5.28 - 5.30 present the observed closing prices compared to the predicted prices obtained from the individual deep learning and combination models in January 2022. For all Figures, the blue lines display the actual stock prices, and the green and pink lines represent the predicted closing prices obtained from half-day and one-day delays deep learning models, respectively. The red lines show the predicted closing prices obtained from the two-delay combination models. The predicted stock prices obtained from all models show similar stock price patterns to the actual stock prices. Our proposed combination model performed better than the individual forecast because our proposed model provides the lowest evaluation metrics for all stocks and deep learning models.

Based on the experimental results with the forecast error criteria on testing data, we notice that our proposed model attained higher performance and better convergence for closing price forecasting. This demonstrates that the two-delay combination approach has obtained an excellent predictive performance for all stocks and deep learning models.

5.5.3 Forecasting Evaluation Based on Diebold-Mariano (DM) Test

We apply the DM test to test whether the two forecasts have significantly different accuracy. In all cases the null hypothesis (H_0) is forecasts from Method i and j are equally accurate. Here $i = 1, 2, \ldots, 36$, $j = 1, 2, \ldots, 36$, and $i \neq j$. The results obtained from the DM test on the univarite and multivariate data are displayed in Tables 5.12 - 5.13.

As shown from Tables 5.12 - 5.13, the p-value for all cases less than 5% level of significance. The null hypothesis is rejected at the 5% level of significance. That is to say, the forecasts obtained from method i, j where $i \neq j$ are not equally accurate. The positive test statistic refers to the model i forecast producing a larger average loss than the model j forecast. Hence, model j is

Table 5.12: The DM test for univariate data

Stock	Method	H_0	Test Statistics	p-value	Result
AAPL	MLP	$m_{43} = m_{79}$	29.604	0.000	Sign of mean loss is + : m_{79} outperform m_{43}
		$m_{43} = c$	22.216	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	2.013	0.044	Sign of mean loss is + : c outperform m_{79}
	CNN	$m_{43} = m_{79}$	-25.112	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	31.734	0.000	Sign of mean loss is + :c outperform m_{43}
		$m_{79} = c$	32.093	0.000	Sign of mean loss is + : c outperform m_{79}
	LSTM	$m_{43} = m_{79}$	-60.058	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	57.345	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	59.310	0.000	Sign of mean loss is + : c outperform m_{79}
ADBE	MLP	$m_{43} = m_{79}$	-40.421	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	14.202	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	27.918	0.000	Sign of mean loss is + : c outperform m_{79}
	CNN	$m_{43} = m_{79}$	-45.379	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	30.234	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	39.229	0.000	Sign of mean loss is + : c outperform m_{79}
	LSTM	$m_{43} = m_{79}$	-44.280	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	37.878	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	43.172	0.000	Sign of mean loss is + : c outperform m_{79}
DVN	MLP	$m_{43} = m_{79}$	-52.527	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	-606.223	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	-606.226	0.000	Sign of mean loss is + : c outperform m_{79}
	CNN	$m_{43} = m_{79}$	-11.215	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	42.169	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	54.142	0.000	Sign of mean loss is + : c outperform m_{79}
	LSTM	$m_{43} = m_{79}$	119.726	0.000	Sign of mean loss is + : m_{79} outperform m_{43}
		$m_{43} = c$	63.487	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	30.645	0.000	Sign of mean loss is + : c outperform m_{79}
MRNA	MLP	$m_{43} = m_{79}$	-65.264	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	50.846	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	60.466	0.000	Sign of mean loss is + : c outperform m_{79}
	CNN	$m_{43} = m_{79}$	-70.327	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	43.483	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	62.498	0.000	Sign of mean loss is + : c outperform m_{79}
	LSTM	$m_{43} = m_{79}$	-66.440	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	49.163	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	55.626	0.000	Sign of mean loss is + : c outperform m_{79}

Table 5.13: The DM test for multivariate data

Stock	Method	H_0	Test Statistics	p-value	Result
AAPL	MLP	$m_{43} = m_{79}$	-22.803	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	12.522	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	18.755	0.000	Sign of mean loss is + : c outperform m_{79}
	CNN	$m_{43} = m_{79}$	-41.486	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	23.672	0.000	Sign of mean loss is + :c outperform m_{43}
		$m_{79} = c$	37.357	0.000	Sign of mean loss is + : c outperform m_{79}
	LSTM	$m_{43} = m_{79}$	-57.759	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	57.643	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	58.423	0.000	Sign of mean loss is + : c outperform m_{79}
ADBE	MLP	$m_{43} = m_{79}$	-32.921	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	17.324	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	25.564	0.000	Sign of mean loss is + : c outperform m_{79}
	CNN	$m_{43} = m_{79}$	-51.181	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	28.871	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	45.713	0.000	Sign of mean loss is + : c outperform m_{79}
	LSTM	$m_{43} = m_{79}$	-35.150	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	32.060	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	32.855	0.000	Sign of mean loss is + : c outperform m_{79}
DVN	MLP	$m_{43} = m_{79}$	-20.767	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	11.182	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	18.367	0.000	Sign of mean loss is + : c outperform m_{79}
	CNN	$m_{43} = m_{79}$	-77.882	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	17.812	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	71.399	0.000	Sign of mean loss is + : c outperform m_{79}
	LSTM	$m_{43} = m_{79}$	70.510	0.000	Sign of mean loss is + : m_{79} outperform m_{43}
		$m_{43} = c$	54.830	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	43.022	0.000	Sign of mean loss is + : c outperform m_{79}
MRNA	MLP	$m_{43} = m_{79}$	-40.947	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	27.975	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	35.560	0.000	Sign of mean loss is + : c outperform m_{79}
	CNN	$m_{43} = m_{79}$	-55.323	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	27.203	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	47.237	0.000	Sign of mean loss is + : c outperform m_{79}
	LSTM	$m_{43} = m_{79}$	-57.936	0.000	Sign of mean loss is - : m_{43} outperform m_{79}
		$m_{43} = c$	38.807	0.000	Sign of mean loss is + : c outperform m_{43}
		$m_{79} = c$	49.231	0.000	Sign of mean loss is + : c outperform m_{79}

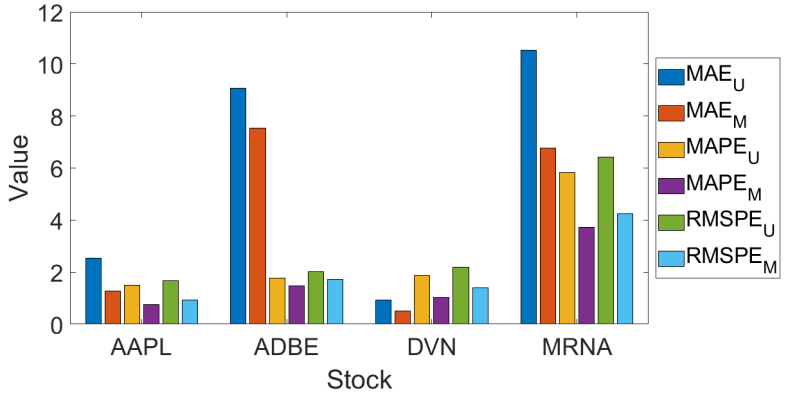
better than that model i. On the other hand, the negative test statistic indicates that the model j forecast produces a larger average loss than the model i forecast. Hence, model i is better than that model j. It can be seen that the two-delay combination method outperforms the individual forecast for all stocks and deep learning techniques.

5.5.4 Analysis of Prediction Results

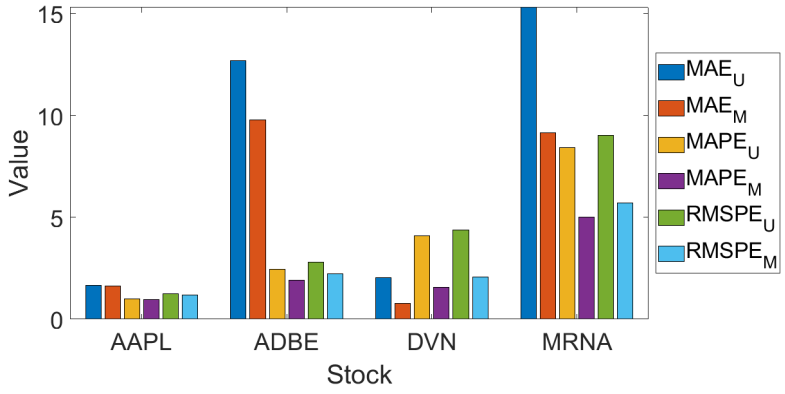
In this section, we compare the accuracy results of the different models on univariate and multivariate features. In addition, we also compare the evaluation metrics (MAE, MAPE, RMSPE) between the univariate and the multivariate time series obtained from the MLP, CNN and LSTM models. The comparison results are depicted in Figures 5.31 - 5.33. respectively.

Figures 5.31 - 5.33 show that the forecast errors obtained from the univariate time series provide higher results than the forecast errors obtained from the multivariate data for all stocks and DL models in terms of MAE, MAPE and RMSPE values.

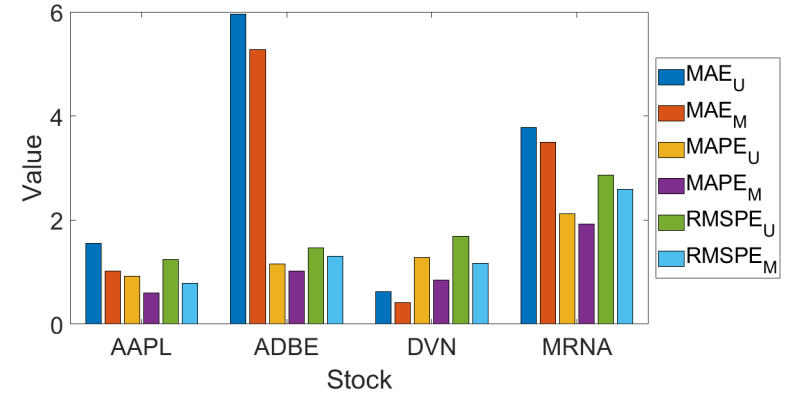
The previous section shows that the two-delay combination procedure outperforms the individual method. We then compare the forecast prices obtained from the two-delay combination method on the univariate and multivariate data. Finally, the time series plot of the actual prices and their forecast observations through the proposed combination scheme for all four stocks and three DL methods are displayed in Figures 5.34 - 5.36.



(a) Half-day delay

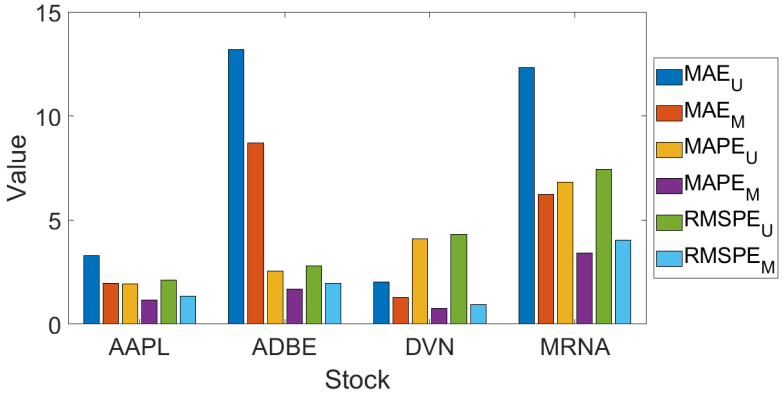


(b) One-day delay

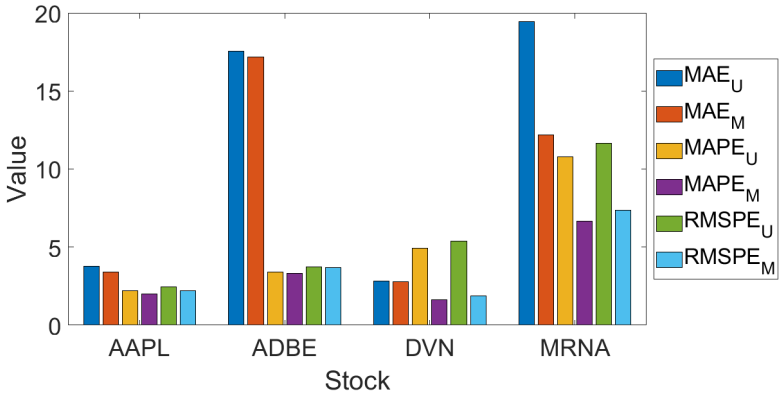


(c) Combination

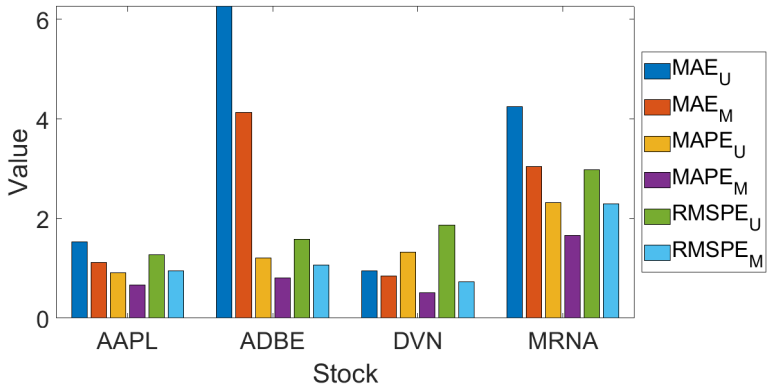
Figure 5.31: Bar diagram showing the performance of all forecast errors obtained from the MLP model on the univariate and multivariate stock price time series.



(a) Half-day delay

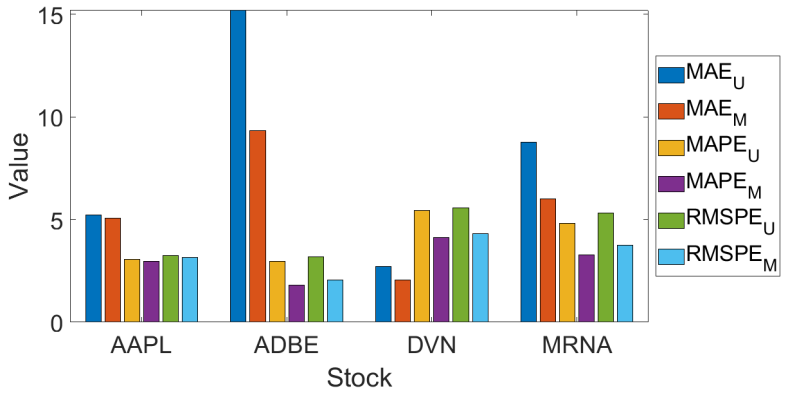


(b) One-day delay

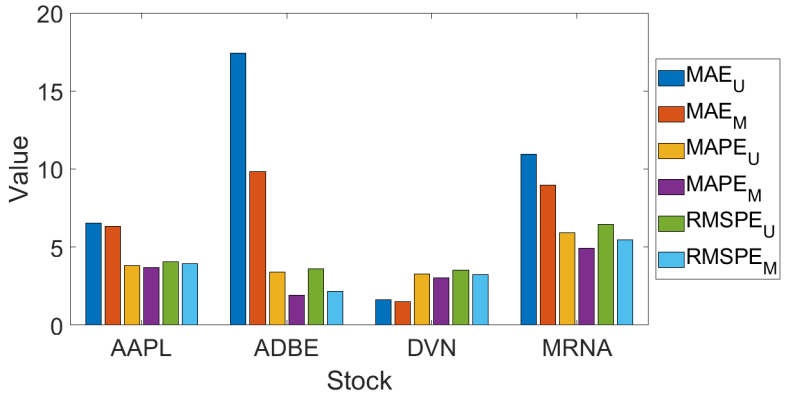


(c) Combination

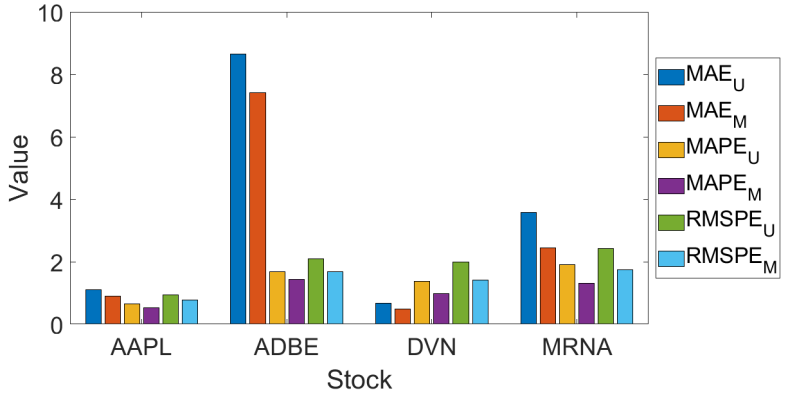
Figure 5.32: Bar diagram showing the performance of all forecast errors obtained from the CNN model on the univariate and multivariate stock price time series.



(a) Half-day delay

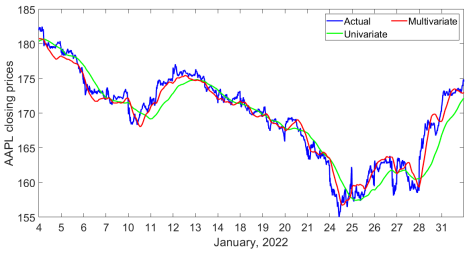


(b) One-day delay

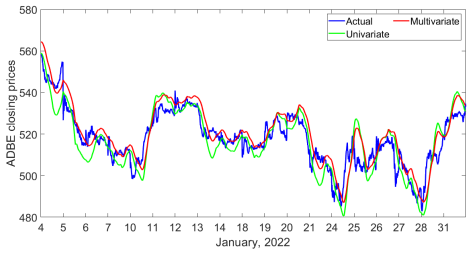


(c) Combination

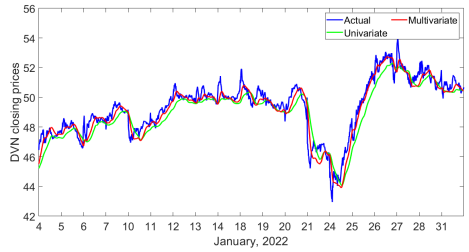
Figure 5.33: Bar diagram showing the performance of all forecast errors obtained from the LSTM model on the univariate and multivariate stock price time series.



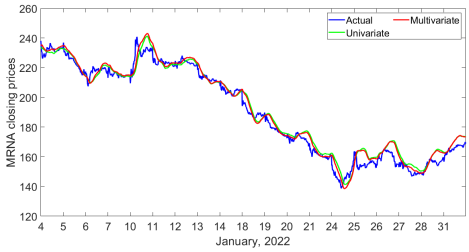
(a) AAPL stock



(b) ADBE stock

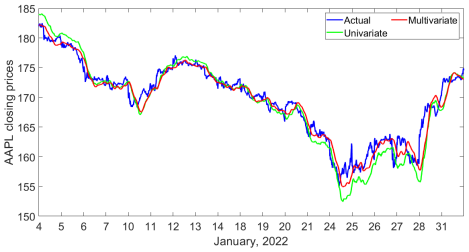


(c) DVN stock

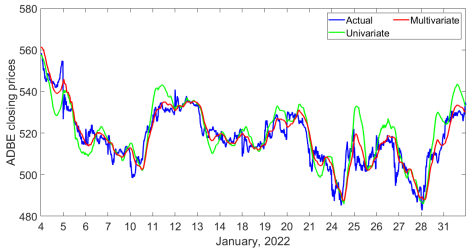


(d) MRNA stock

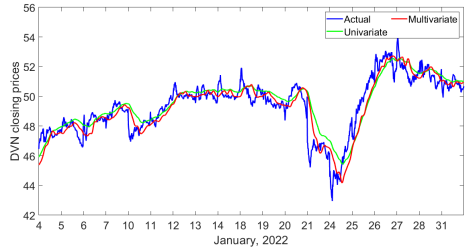
Figure 5.34: Diagrams of actual and combination forecast prices of the MLP method for the time series:(a) AAPL stock, (b) ADBE stock, (c) DVN stock, and (d) MRNA stock



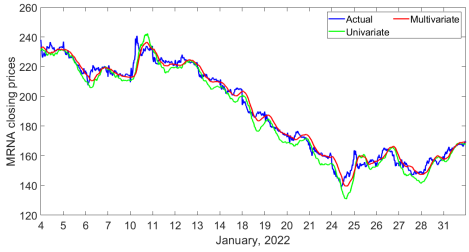
(a) AAPL stock



(b) ADBE stock

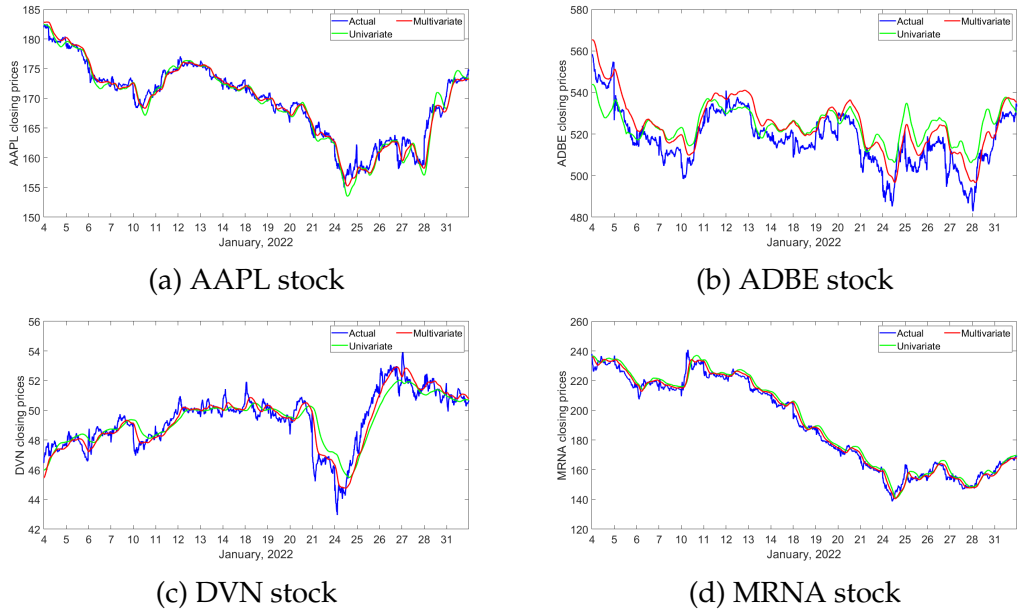


(c) DVN stock



(d) MRNA stock

Figure 5.35: Diagrams of actual and combination forecast prices of the CNN method for the time series:(a) AAPL stock, (b) ADBE stock, (c) DVN stock, and (d) MRNA stock



(a) AAPL stock

(b) ADBE stock

(c) DVN stock

(d) MRNA stock

Figure 5.36: Diagrams of actual and combination forecast prices of the LSTM method for the time series:(a) AAPL stock, (b) ADBE stock, (c) DVN stock, and (d) MRNA stock

As shown from Figures 5.34 - 5.36, we have learnt that multivariate data have a superior predictive capacity compared to univariate data. The forecasting performance under the multivariate time series input condition is improved compared with the forecasting performance under the univariate input condition.

5.6 Conclusion

This chapter proposed a novel two-delay combination procedure to improve the predictive accuracy for the univariate and multivariate time series. The data used in this chapter are collected from the Thomson Reuters database every 5 minutes between January 2021 and January 2022. In our framework, before applying the intraday financial data to the traditional deep learning model, we divided data into three sets, namely training, validation, and test sets. We then scale data between 0 and 1 using the Min-Max normalization

technique. This procedure improves the performance and training stability of the model. The traditional deep learning (DL) methods used in this chapter are the MLP, CNN, and LSTM procedures. Each DL technique is applied to four stocks, namely Apple Inc. (AAPL), Adobe Inc. (ADBE), Devon Energy Corporation (DVN), and Moderna Inc. (MRNA) stocks, using a half-day delay and one-day delay of each DL model. Next, the SHERPA algorithm is utilized to seek the hyperparameter for the DL techniques. The essential step to confirm that our model architecture can predict the following prices is to examine the over/underfitting problem by comparing the loss function between the training and validation sets. After that, the individual delay is integrated using the linear combination technique.

Next, the differential evolution algorithm (DE) is presented to estimate the optimal weight of each combination technique. Then the MAE, MAPE, RMSPE, and percentage improvement are computed to compare the performance between the individual and the combination procedures. The experimental results indicate that the two-day combination technique using the DE algorithm is more effective than the single technique in terms of the evaluation metrics. Our proposed approach via DE weight identification improves predictive accuracy for the univariate and multivariate time series by up to 78.97% and 82.27%, respectively. The Diebold-Mariano tests show that the forecasts in all pairwise comparisons are not equally accurate. This confirms that the two-delay combination method is outstanding in the other methods. In addition, the proposed technique gives a beneficial advantage in improving the accuracy of each DL technique by using different delays to forecast stock prices. That means applying several DL methods in the combination technique is unnecessary. Based on the general idea of the combination forecast model, it requires at least two different approaches to build the forecast combination. Under our proposed technique, using only one method with two different delays can reduce the evaluation metrics. Consequently, the two-delay combination model

is a potential method with satisfactory prediction performance. Finally, the multivariate analysis enables building more accurate forecasting models than univariate analysis for all stocks and DL models.

Chapter 6

Conclusions and Future Work

This dissertation considered various topics related to delay in the time series analysis, namely the delay in the classical autoregressive model, the delay in the stochastic differential equation and the two-delay combination method using the deep learning technique. The proposed methods for solving these problems are described in Chapters 3-5. In the following content, the results and the contribution of the dissertation are summarized.

6.1 Conclusions

Chapter 3, we proposed a novel autoregressive model called the m-delay autoregressive (MAR) model. This model is an extraordinary case of the traditional autoregressive model. The predicted observations depend on only the previous data at time $t-1$ and $t-m$, respectively. The least squares approach is developed to estimate two model parameters of the m-delay autoregressive model. Our parameter identification is named the m-delay autoregressive coefficients. For Monte Carlo simulations, the effectiveness of the m-delay formula is examined. It indicates that for the small and medium delay, the sample size over 300 provides good results of approximating the average of the two model parameters. For the larger delay, the average of two model parameters

approaches the actual ones when the sample size is about 500.

Furthermore, the brute-force algorithm is applied to seek the optimal delay. For the case study, the average minimum temperature in Perth, Western Australia, from January 1994 to June 2019, a totally of 306 observations are utilized. The experimental results obtained from the MAR model and the classical AR model demonstrated no significant differences between the two models. In addition, the MAR model reduces computing time in the prediction step because our proposed model requires only two parameters, while m parameters are mandatory for the general AR model. Accordingly, the MAR model is effective for predicting time series data.

The matching volatility obtained from the stochastic delay differential equation (SDDE) and the real-world stock is presented in Chapter 4. The parameter of the drift term (λ) and the volatility (σ) are two parameters in this model. The model identification is divided into two cases: one unknown parameter (λ) and two unknown parameters (λ, σ). For one parameter estimation, the drift term (λ) parameter is unknown, while the volatility (σ) is given. The m-delay autoregressive coefficient (ARC) algorithm is proposed to estimate the parameter (λ). Next, we compute the estimated volatility from the Monte Carlo simulations and compare it with the actual volatility. The data used in this chapter are the closing prices recorded every 5 minutes and 15 minutes from the Thomson Reuters database in 2008 total of 19,750 and 6,750, respectively. We selected some of the top stocks in the New York Stock Exchange (NYSE), including the International Business Machines Corporation (IBM), the Microsoft Corporation (MSFT), the Standard and Poor's 500 (S&P 500) and the Standard and Poor's 100 (S&P 100). The numerical results indicated that the estimated volatility using the ARC algorithm follows a similar pattern to the actual volatility for all stocks. Using a larger delay can improve the performance of the matching volatility because the estimated volatility is close to the actual volatility. For two model identification, the differential evolution (DE)

algorithm is utilized to estimate two model parameters, namely the parameter of the drift term (λ) and the volatility (σ) of the SDDE. Empirical findings show that the estimated volatility for all stocks with two different sampling frequencies fits the actual volatility when the delay is large.

The deep learning (DL) techniques for stock price prediction are presented in Chapter 5. The two-delay combination technique using DE weighted optimization algorithm is proposed to reduce the forecast error obtained from the individual DL method. The data used in this chapter are the univariate and multivariate stock price time series. For univariate data, only the closing price is an input variable. In contrast, five inputs for the multivariate data set are the opening price, the highest price, the lowest price, the closing price, and the trading volume (OHLCV). We selected four big companies' stocks on the New York Stock Exchange (NYSE), which operates on a weekday between 9.30 am. and 4.00 pm., including Apple Inc. (AAPL), Adobe Inc. (ADBE), Devon Energy Corporation (DVN), and Moderna, Inc. (MRNA) stocks. The historical financial data series were divided into three sets, namely training, validation and test sets. For each stock, 19,829 data points from January to December 2021 were used for training and validating the deep learning models. 83 % (January - October 2021) were used for training (learning parameters), and the remaining 17 % (November - December 2021) to validate the performance of the network and avoid overfitting. The remaining one months (January 2022) were used to evaluate the model performance.

The results of four real-world stock price time series and three traditional DL methods indicate that the two-delay combination model using DE weight optimization is more effective than the single DL technique for the univariate and multivariate data. One benefit of our proposed technique is that applying several DL methods in a data set is unnecessary because our approach combines different delays on each data set. In contrast, the classical combination method combines various individual techniques. Moreover, our proposed

technique provides improving the accuracy of each DL. Finally, using multivariate data enable us to build more accurate forecasting models than univariate data for all stocks and DL models.

6.2 Future Work

The future works can be consider the following three extensions based on different delay project.

Firstly, the historical data have been applied to each chapter to evaluate our proposed models. The missing data is one of the possible problems for the real-world data set. This thesis uses the spline interpolation technique to fill any missing value. It would be interesting to replace missing data with other promising methods such as mean imputation, autoregressive-model-based, genetic algorithm, maximum likelihood (EM) based method, and maximum likelihood method. Changing the method for handling the missing data might affect the experimental results.

Secondly, adding more delay of the SDDE in Chapter 4 might be a concern to increase the accuracy of the matching process

Thirdly, the data used in Chapter 4 and Chapter 5 are 5-minute time interval. A much more interesting is the investigation of the time window. Higher frequency (e.g. 1-min, 30-sec) would probably boost the model accuracy.

Finally, in Chapter 5, we applied only a supervised deep learning model to predict closing prices. However, there are several deep learning methods to predict future observation, including semi-supervised and unsupervised deep learning approaches. For the combination technique, we can combine multiple delays of each DL model. Furthermore, we focus on only the linear combination forecast technique. The non-linear combination method is another interesting technique to increase the accuracy of the DL model.

Appendices

Appendix A

List of Abbreviations & Figures

The following list is neither exhaustive nor exclusive, but may be helpful.

AAPL Apple Inc.

ADBE Adobe Inc.

AI Artificial Intelligence

ANN Artificial Neural Network

AR Autoregressive

ARC Autoregressive Coefficients Algorithm

ARCH Autoregressive Conditional Heteroskedastic

ARIMA Autoregressive Integrated Moving Average

ARMA Autoregressive Moving Average

CNN Convolutional Neural Network

CNN-LSTM Convolutional Neural Network and Long Short-Term Memory

DE Differential Evolution

DL Deep Learning

DVN Devon Energy Corporation

GARCH Generalized Autoregressive Conditional Heteroskedastic

IBM International Business Machines Corporation

JPM JP Morgan Chase & Co

LSTM Long Short-Term Memory

MAR m-Delay AR Model

MAPE Mean Absolute Percentage Error

MdAE Median Absolute Error

MdAPE Median Absolute Percentage Error

ML Machine Learning

MLP Multilayer Perceptrons

MRNA Moderna Inc

MSE Mean Square Error

MSFT Microsoft Corporation

NN Neural Network

NYSE New York Stock Exchange

RMSE Root Mean Square Error

RMSPE Root Mean Square Percentage Error

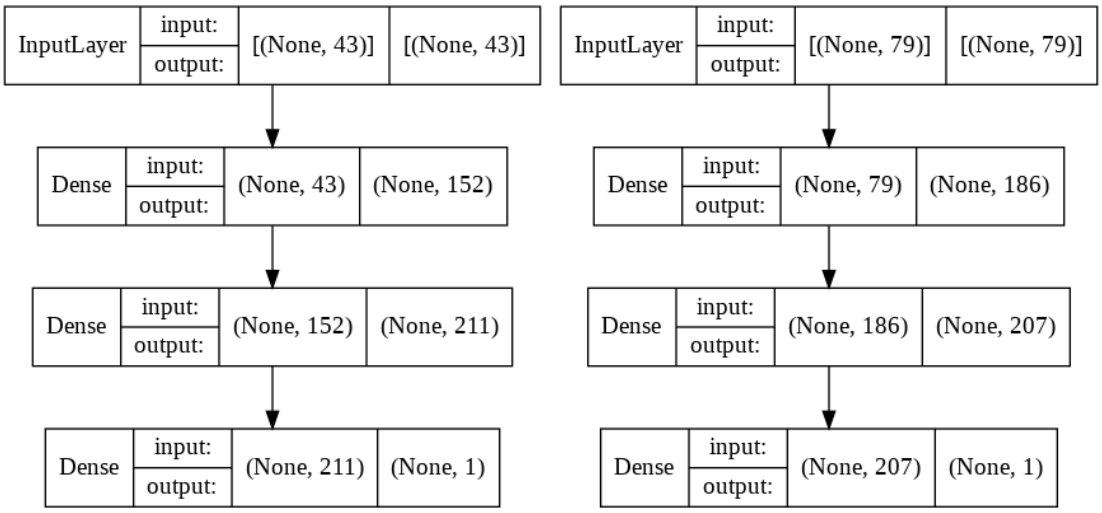
RNN Recurrent Neural Network

SDE Stochastic Differential Equation

SDDE Stochastic Delay Differential Equation

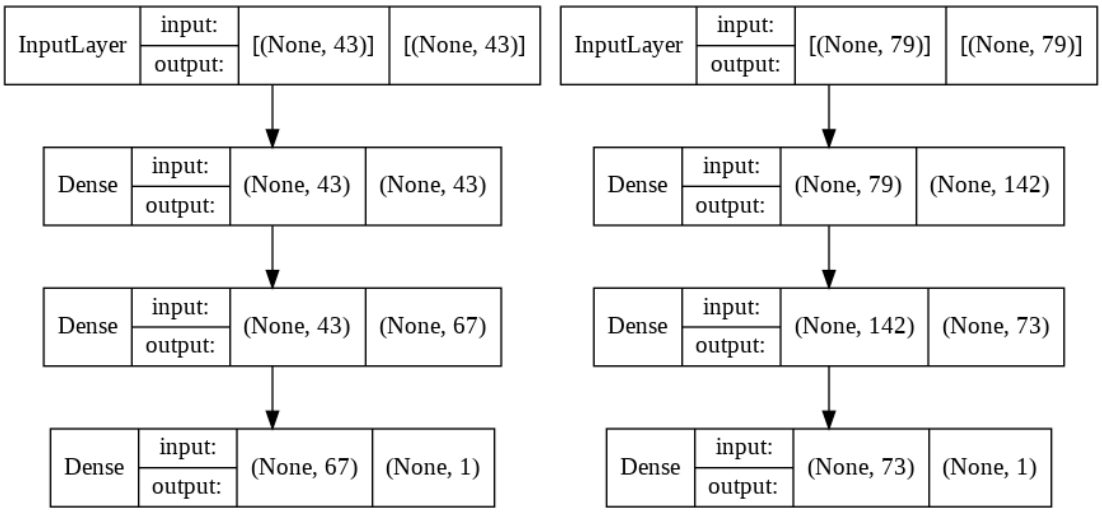
S&P 500 Standard and Poor's 500

S&P 100 Standard and Poor's 100



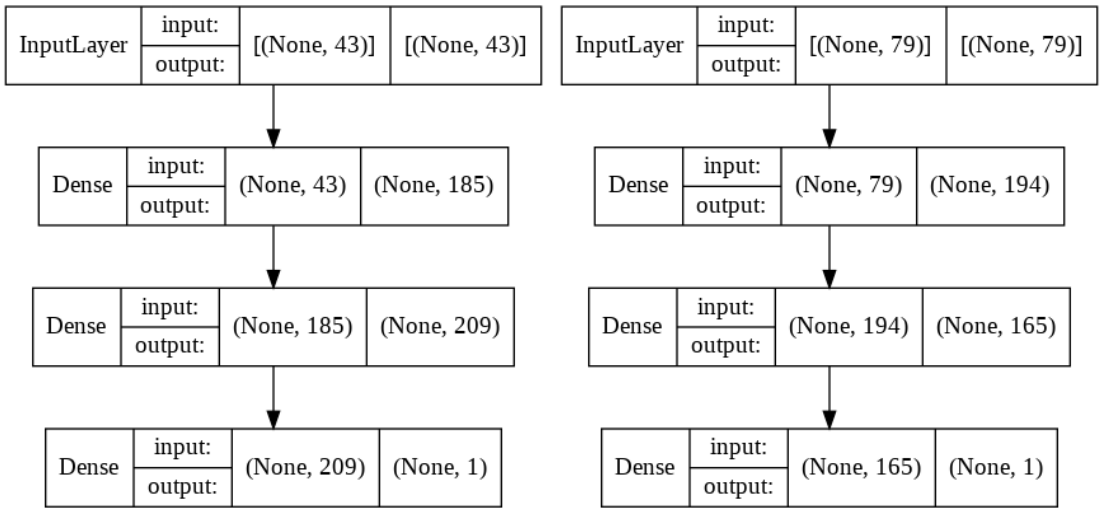
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.1: The architecture of univariate MLP model on AAPL stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



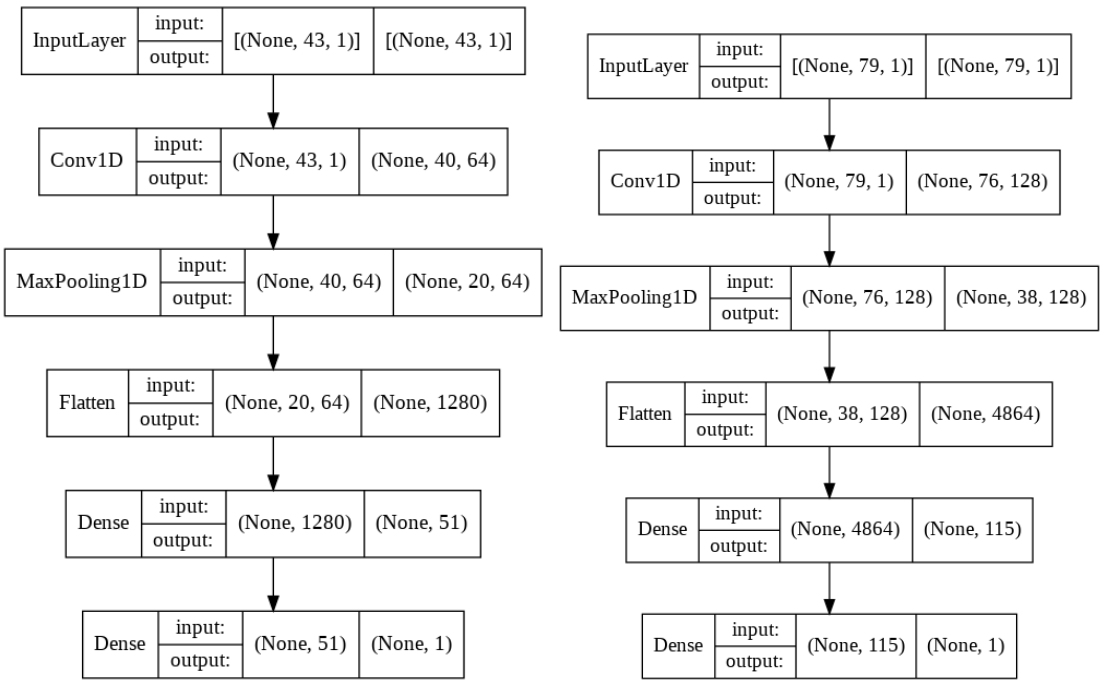
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.2: The architecture of univariate MLP model on DVN stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



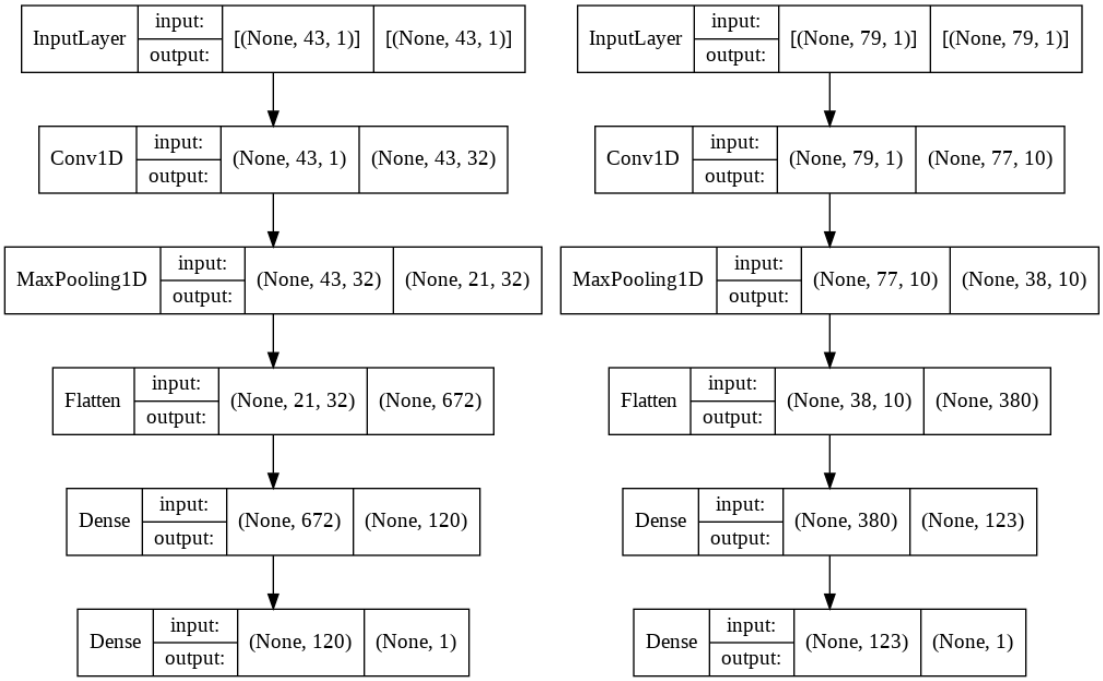
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.3: The architecture of univariate MLP model on MRNA stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



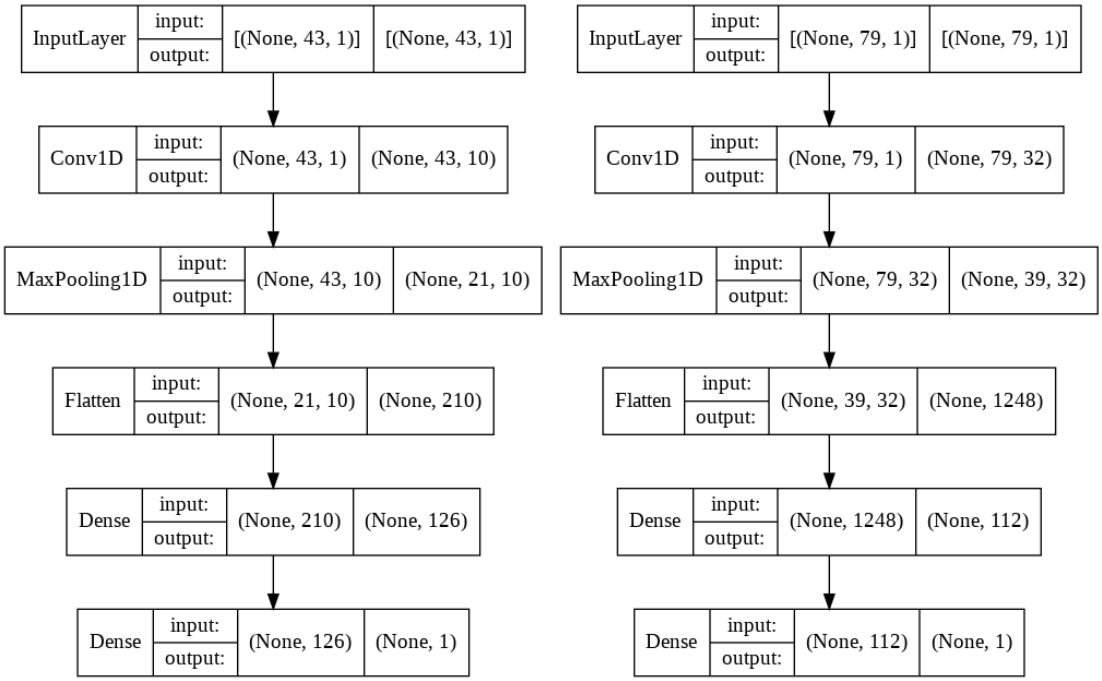
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.4: The architecture of univariate CNN model on AAPL stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



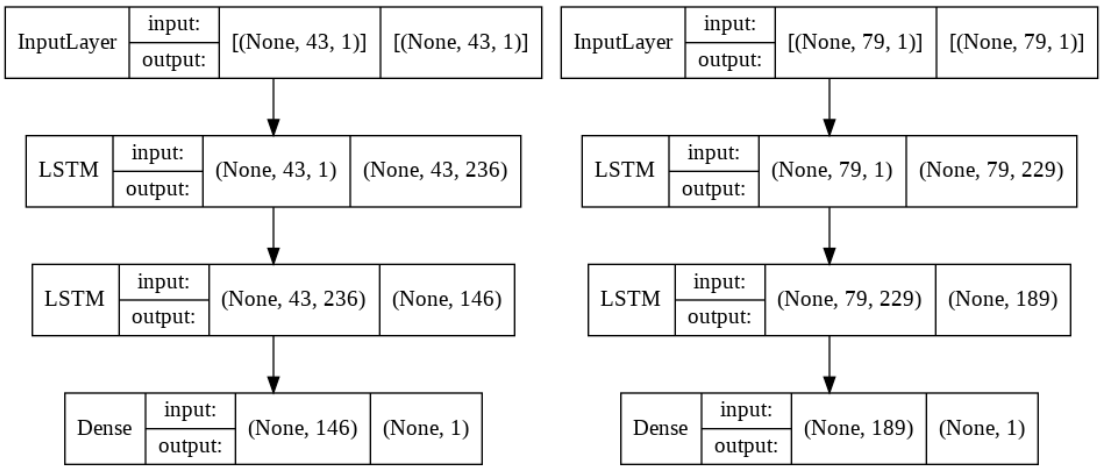
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.5: The architecture of univariate CNN model on DVN stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



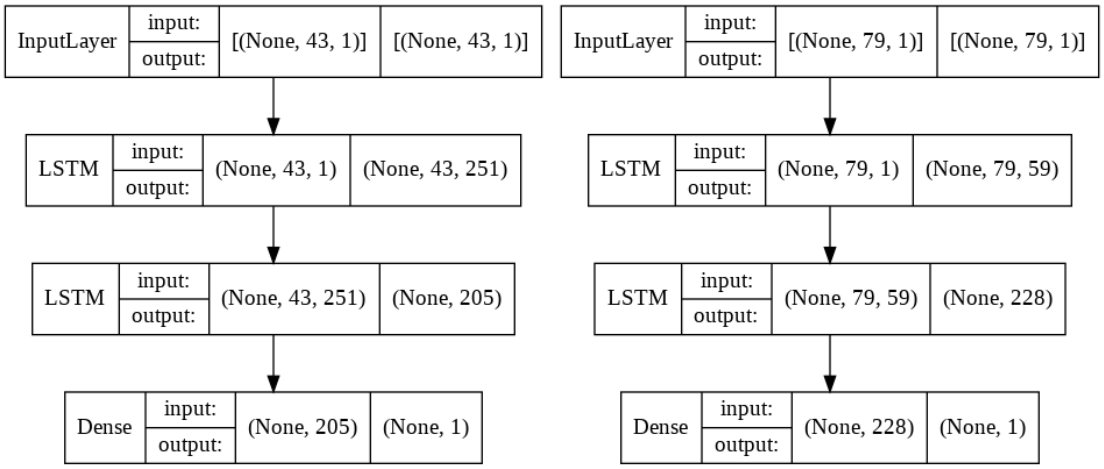
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.6: The architecture of univariate CNN model on MRNA stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



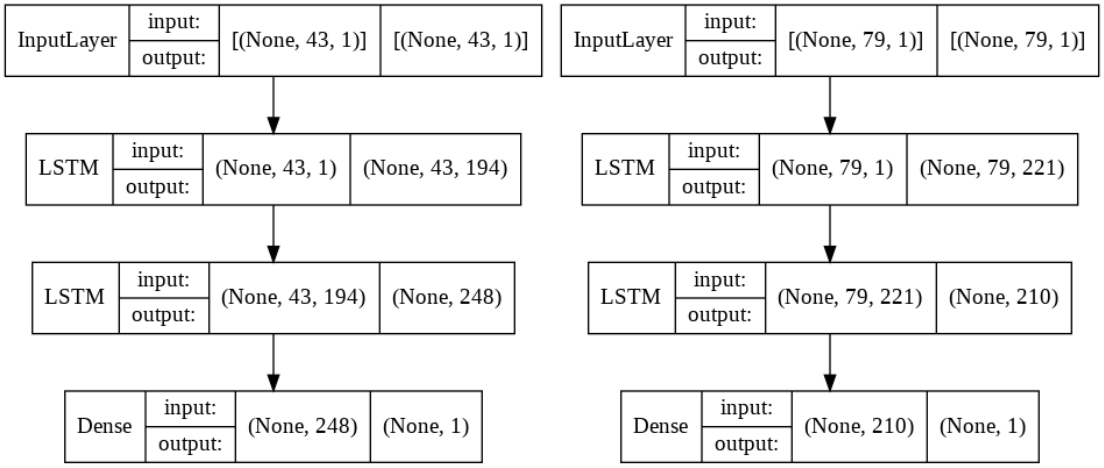
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.7: The architecture of univariate LSTM model on AAPL stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



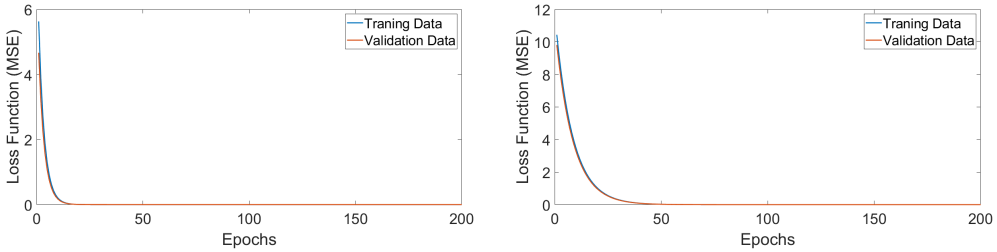
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.8: The architecture of univariate LSTM model on DVN stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



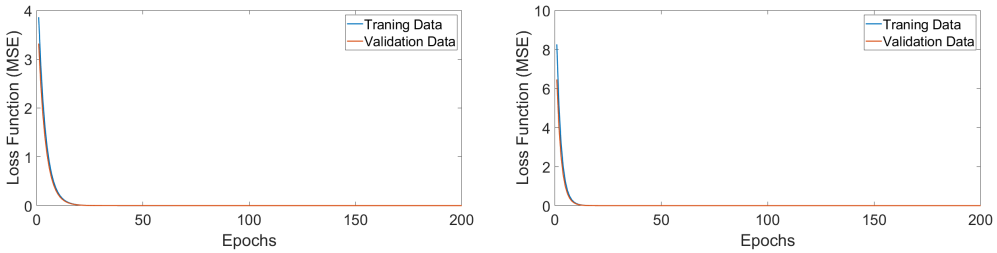
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.9: The architecture of univariate LSTM model on MRNA stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



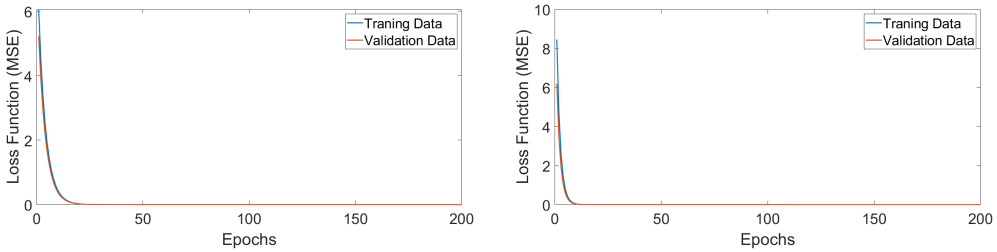
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.10: The convergence plot between the loss function of the training and validation sets while training the MLP for the univariate AAPL stock



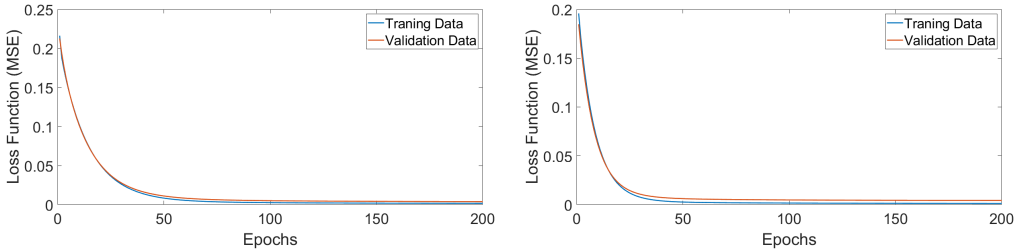
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.11: The convergence plot between the loss function of the training and validation sets while training the MLP for the univariate DVN stock



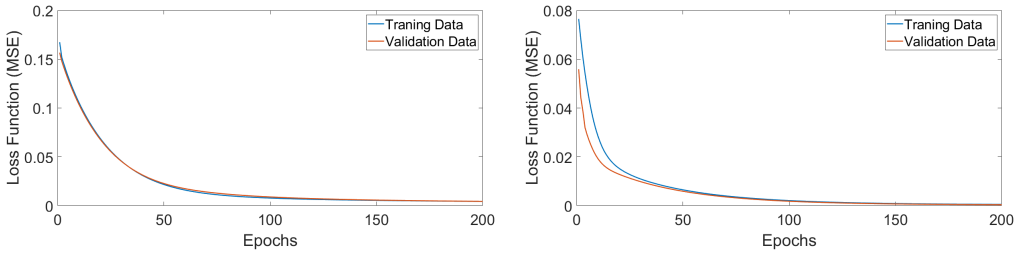
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.12: The convergence plot between the loss function of the training and validation sets while training the MLP for the univariate MRNA stock



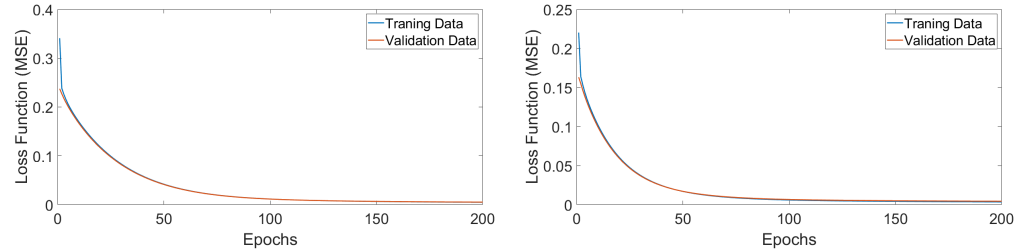
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.13: The convergence plot between the loss function of the training and validation sets while training the CNN for the univariate AAPL stock



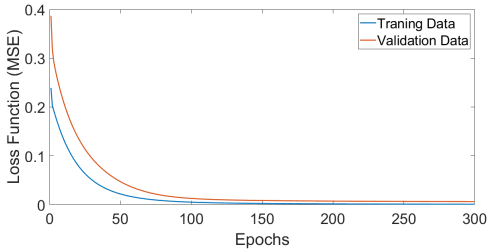
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.14: The convergence plot between the loss function of the training and validation sets while training the CNN for the univariate DVN stock

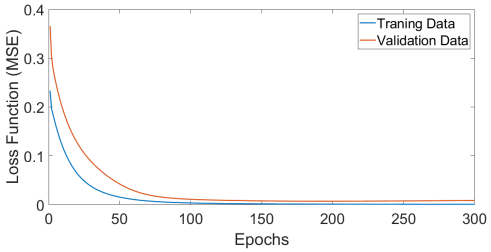


(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.15: The convergence plot between the loss function of the training and validation sets while training the CNN for the univariate MRNA stock

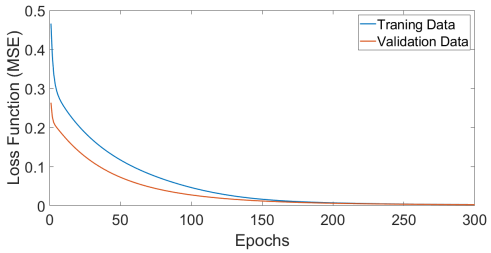


(a) Half-day trading delay ($m = 43$)

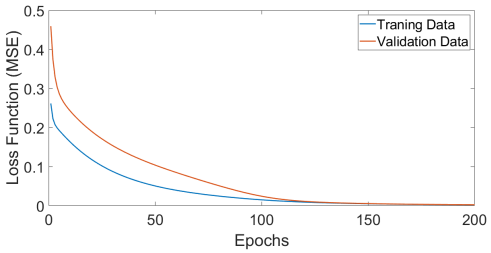


(b) One-day trading delay ($m = 79$)

Figure A.16: The convergence plot between the loss function of the training and validation sets while training the LSTM for the univariate AAPL stock

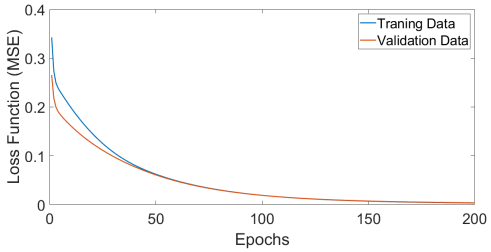


(a) Half-day trading delay ($m = 43$)

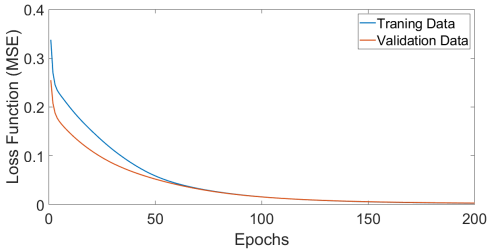


(b) One-day trading delay ($m = 79$)

Figure A.17: The convergence plot between the loss function of the training and validation sets while training the LSTM for the univariate DVN stock

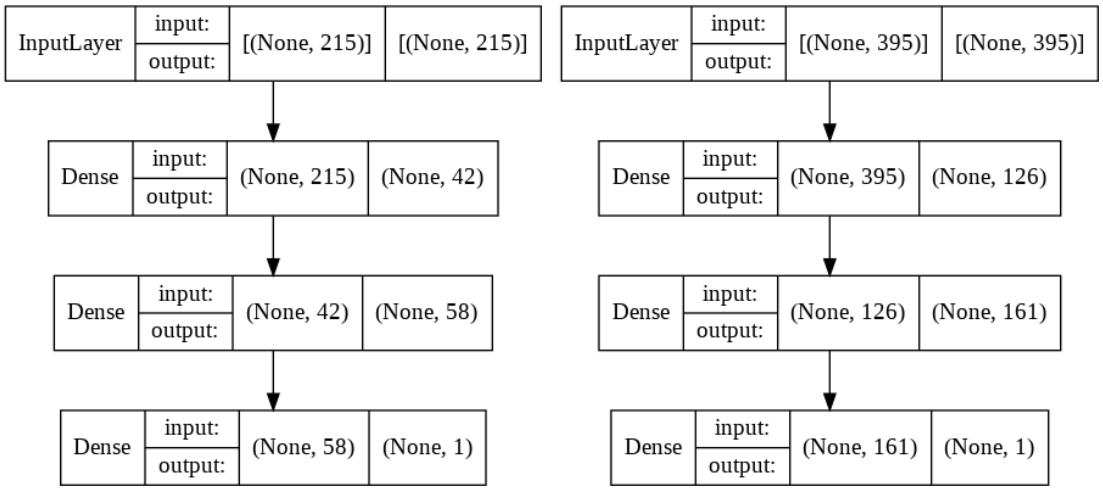


(a) Half-day trading delay ($m = 43$)



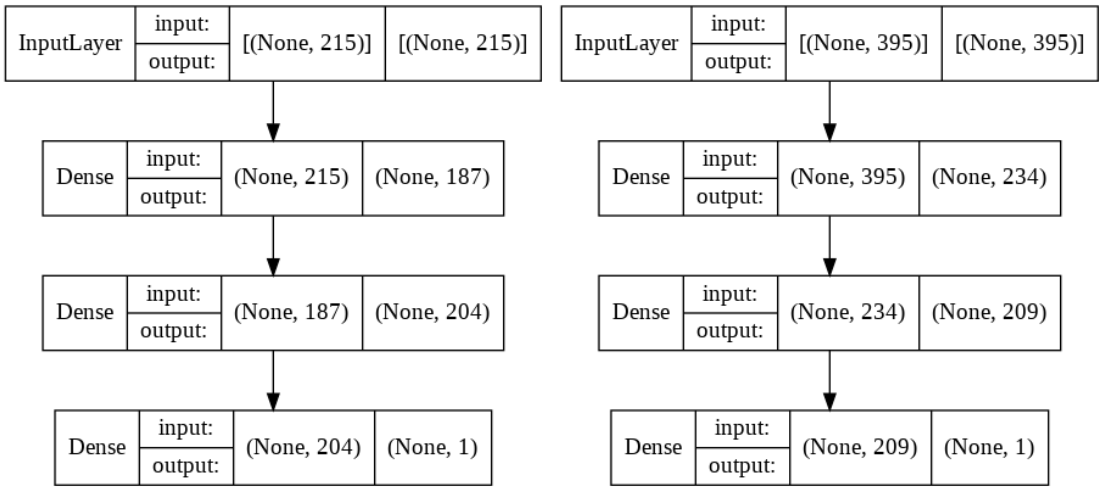
(b) One-day trading delay ($m = 79$)

Figure A.18: The convergence plot between the loss function of the training and validation sets while training the LSTM for the univariate MRNA stock



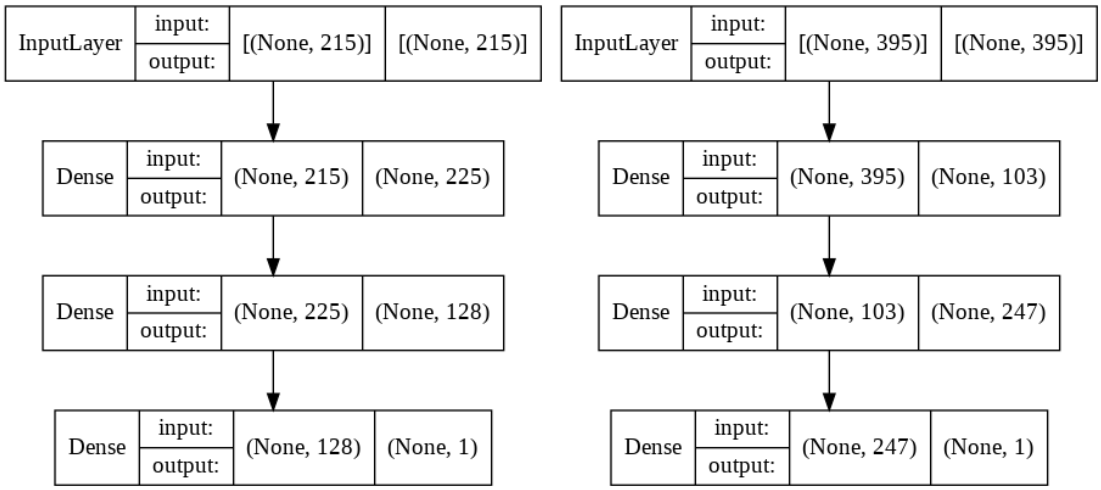
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.19: The architecture of multivariate MLP model on AAPL stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



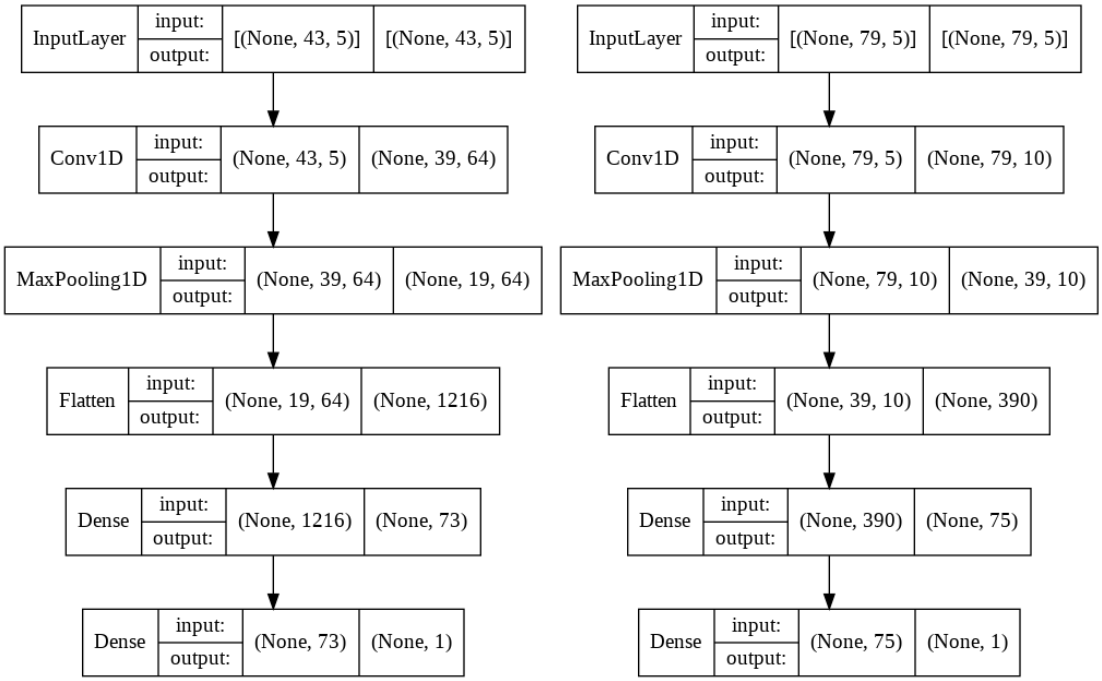
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.20: The architecture of multivariate MLP model on DVN stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



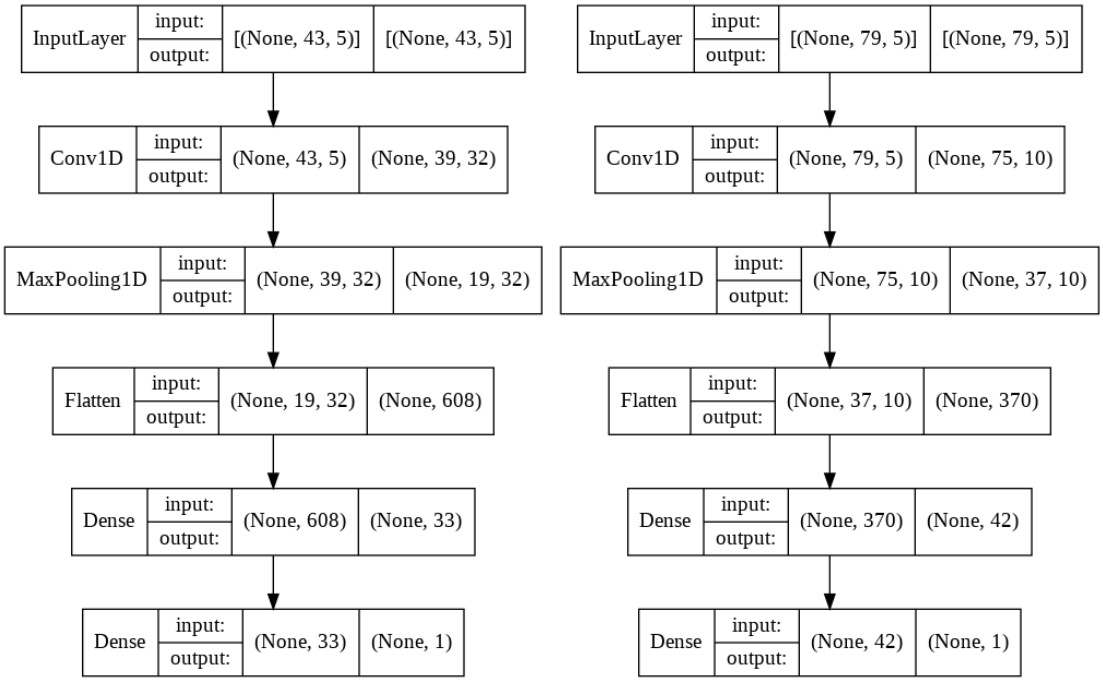
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.21: The architecture of multivariate MLP model on MRNA stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



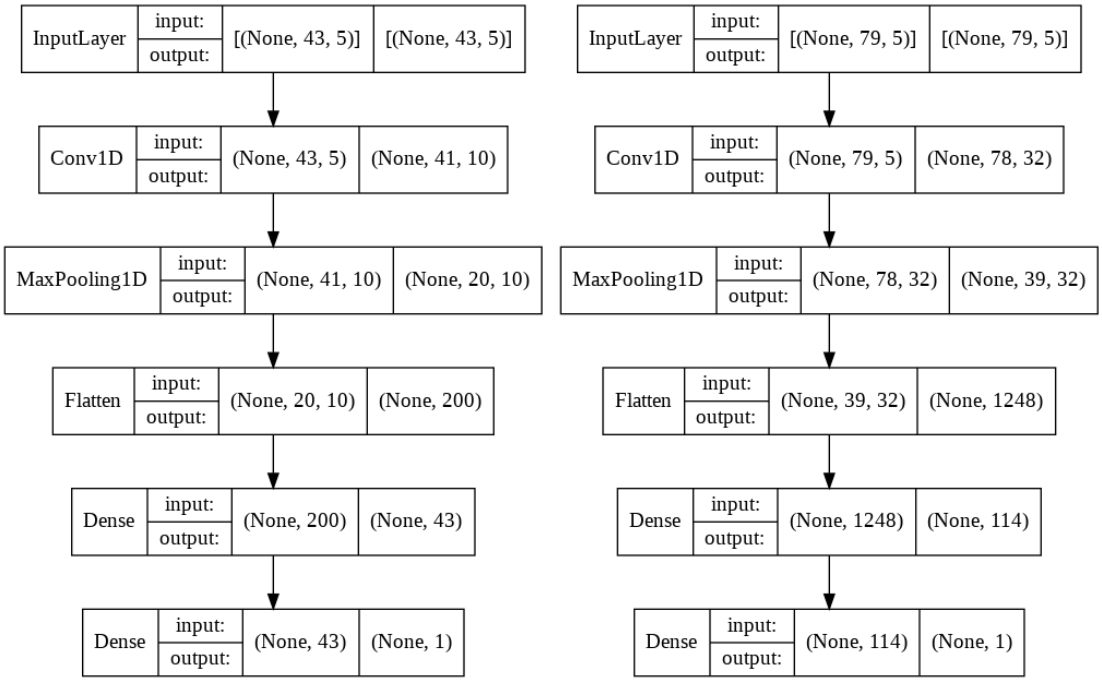
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.22: The architecture of multivariate CNN model on AAPL stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



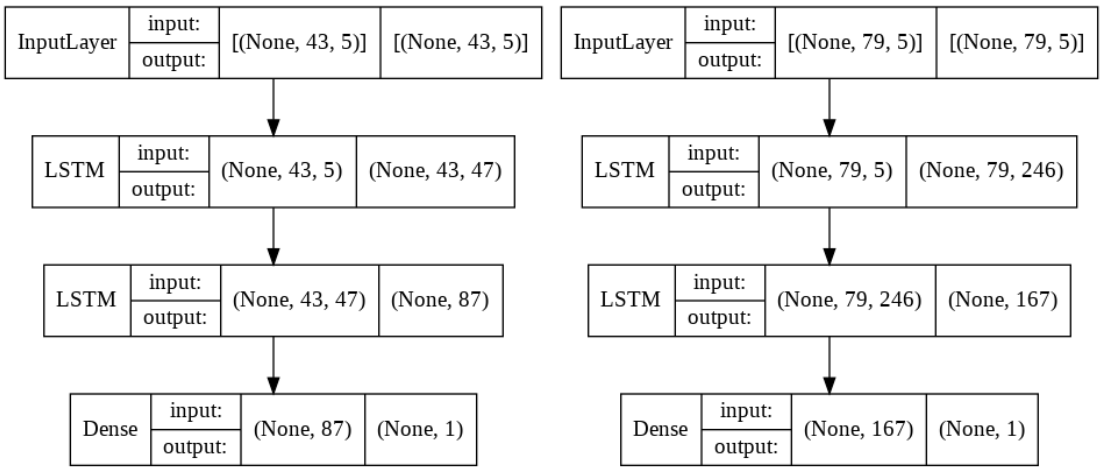
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.23: The architecture of multivariate CNN model on DVN stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



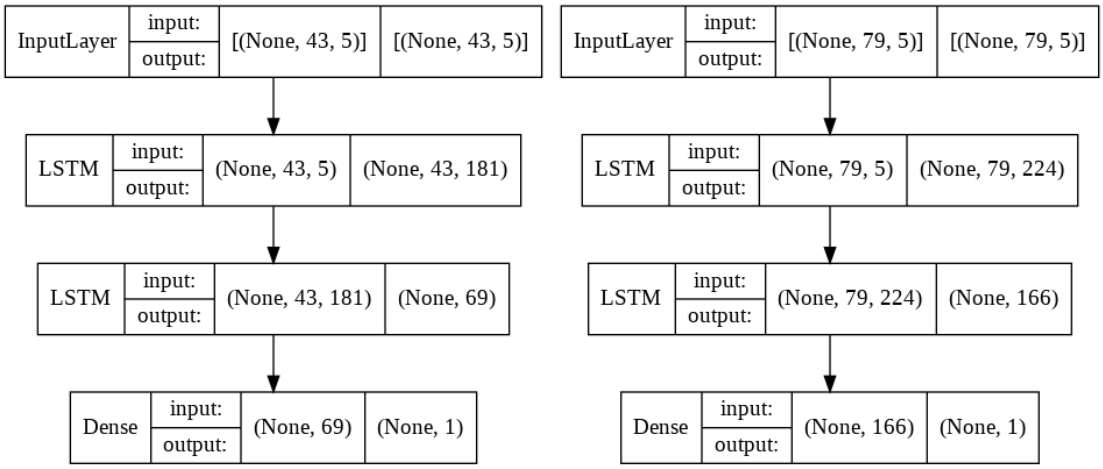
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.24: The architecture of multivariate CNN model on MRNA stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



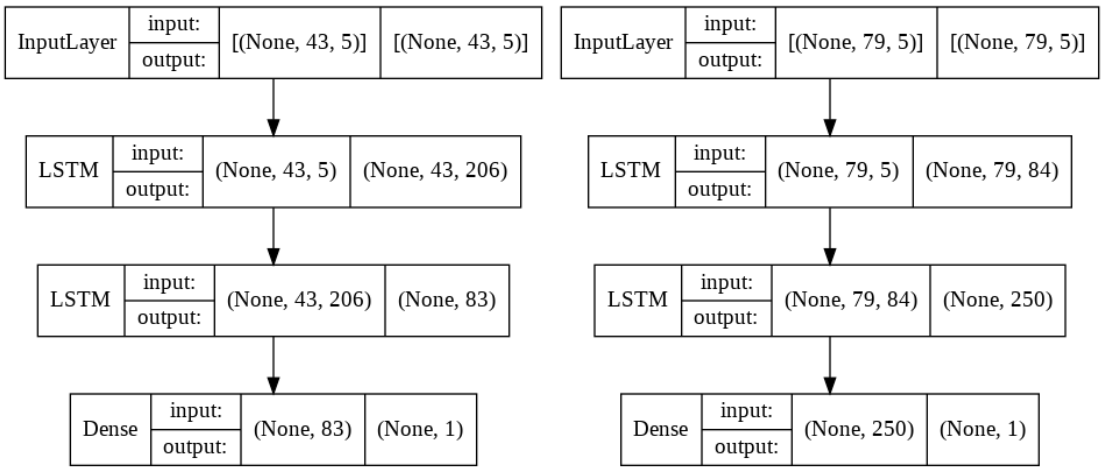
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.25: The architecture of multivariate LSTM model on AAPL stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



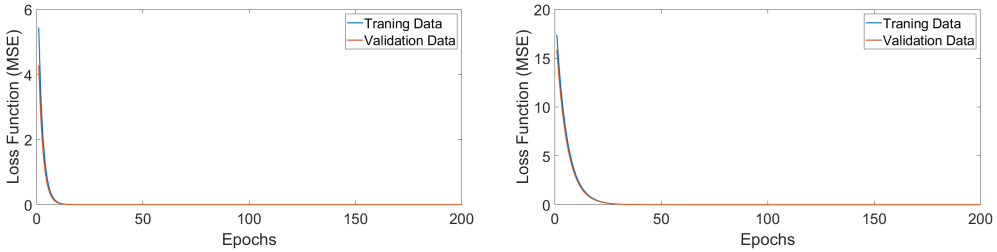
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.26: The architecture of multivariate LSTM model on DVN stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



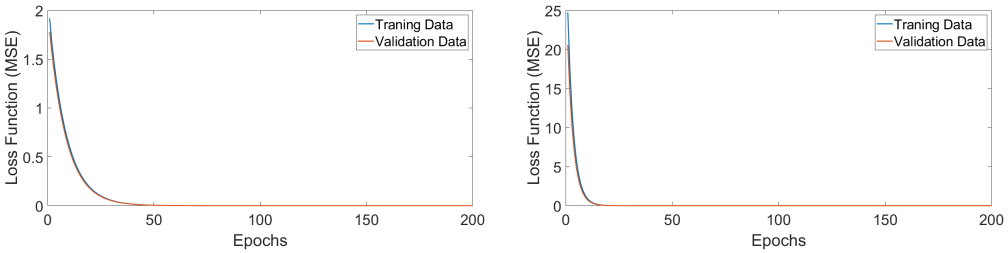
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.27: The architecture of multivariate LSTM model on MRNA stock with: (a) half-day trading delay ($m = 43$) and (b) one-day trading delay ($m = 79$)



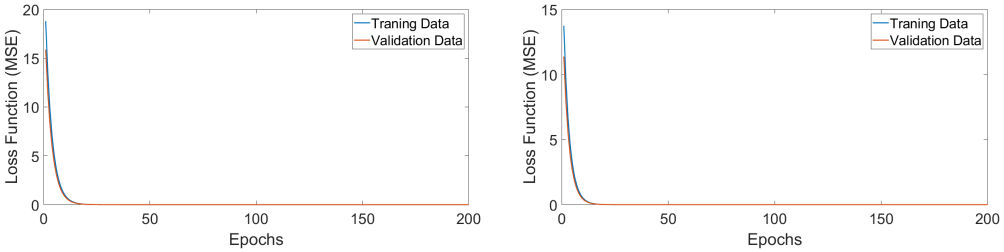
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.28: The convergence plot between the loss function of the training and validation sets while training the MLP for the multivariate AAPL stock



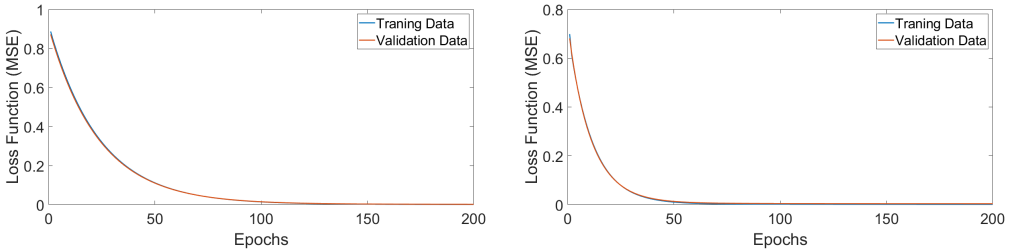
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.29: The convergence plot between the loss function of the training and validation sets while training the MLP for the multivariate DVN stock



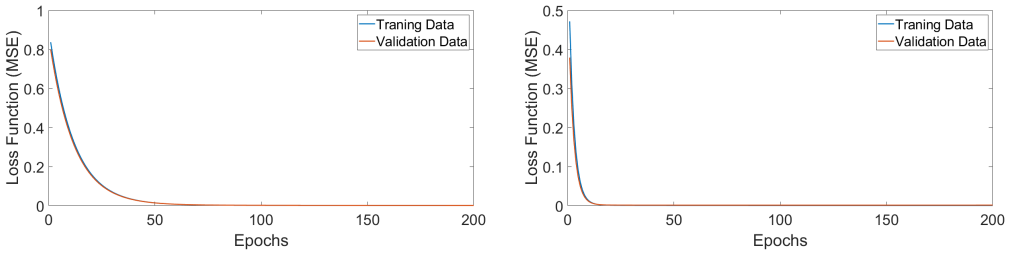
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.30: The convergence plot between the loss function of the training and validation sets while training the MLP for the multivariate MRNA stock



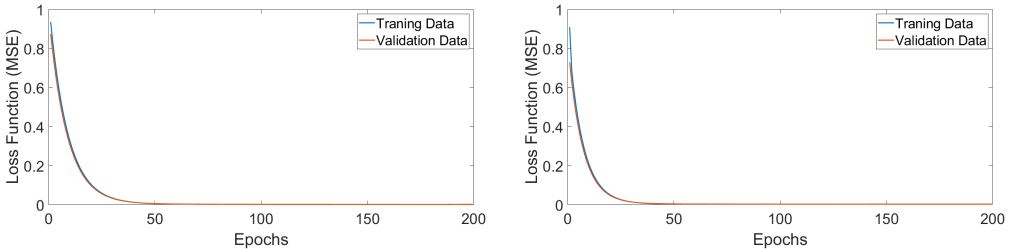
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.31: The convergence plot between the loss function of the training and validation sets while training the CNN for the multivariate AAPL stock



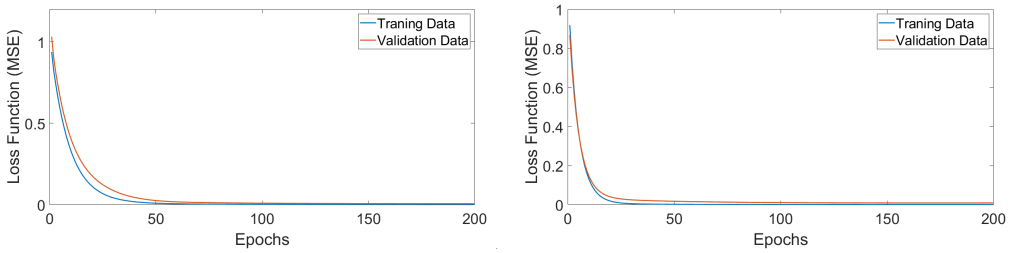
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.32: The convergence plot between the loss function of the training and validation sets while training the CNN for the multivariate DVN stock



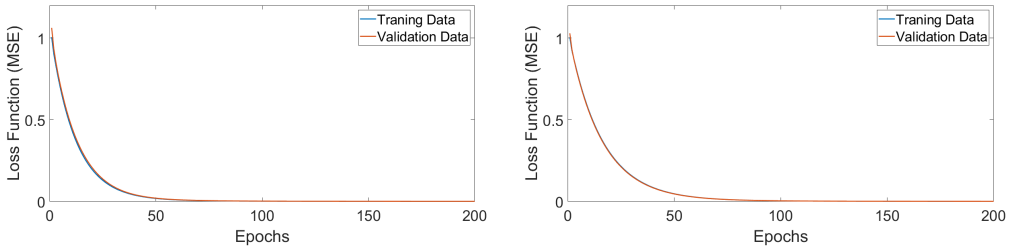
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.33: The convergence plot between the loss function of the training and validation sets while training the CNN for the multivariate MRNA stock



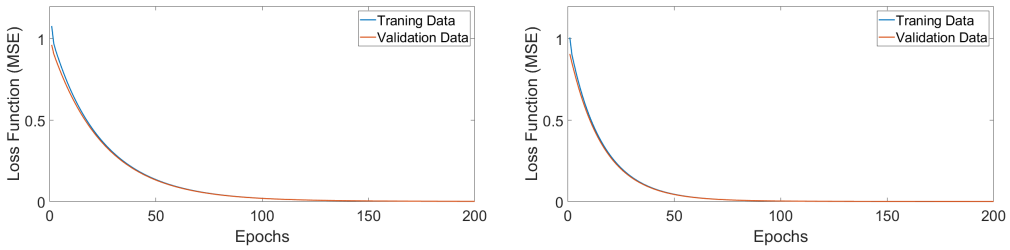
(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.34: The convergence plot between the loss function of the training and validation sets while training the LSTM for the multivariate AAPL stock



(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.35: The convergence plot between the loss function of the training and validation sets while training the LSTM for the multivariate DVN stock



(a) Half-day trading delay ($m = 43$) (b) One-day trading delay ($m = 79$)

Figure A.36: The convergence plot between the loss function of the training and validation sets while training the LSTM for the multivariate MRNA stock

Appendix B

Copyright Information

In this appendix, the copyright agreements for published papers allow authors to reproduce an extract of all works in a doctorals thesis. The hierarchical order of permission to re-produce published material were obtained from IEEE conferences and journals in which the author has published.

Re: Request for permission to 100% re-produce my publication into my thesis

cmes@techscience.com <cmes@techscience.com>
Wed 20/10/2021 9:20 AM
To: Manlika Ratchagit <manlika.ratchagit@postgrad.curtin.edu.au>

Dear Manlika Ratchagit,

You can do this with proper acknowledgment and citation.

If you have any questions, please feel free to let us know.

Best regards,

Amy Dong

CMES Editorial Office
871 Coronado Center Drive, Suite 200,
Henderson, Nevada, 89052, USA
https://www.techscience.com/

From: Manlika Ratchagit
Date: 2021-10-17 10:09
To: cmes@techscience.com
Subject: Request for permission to 100% re-produce my publication into my thesis

Dear Editor,

My name is Ms Manlika Ratchagit. I am studying PhD program in the area of Mathematics and Statistics, Curtin University. My manuscript has already been published in CMES-Computer Modeling in Engineering & Sciences. Vol.122, No.2, 2020, pp.487-504. The digital object identifier of my manuscript is 10.32604/cmes.2020.08865

I am the first author of this paper. I kindly ask you, will I be able to 100% re-produce (copy the material of this paper and paste it into my thesis) my published paper into my thesis?

Thank you for considering my request. I am looking forward to your reply.

Sincerely yours,

Manlika Ratchagit
PhD student (Mathematics & Statistics)

School of Electrical Engineering, Computing and Mathematical Sciences
Curtin University
Bentley 6102, Perth, Western Australia

Email: manlika.ratchagit@postgrad.curtin.edu.au

IEEE COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

On Parameter Estimation of Stochastic Delay Difference Equation using the Two m-delay Autoregressive Coefficients
Ms. Manlika Ratchagit, Prof. Benchawan Wiwatanapataphee and Dr. Darfiana Nur
2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)

COPYRIGHT TRANSFER

The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the "IEEE") all rights under copyright that may exist in and to: (a) the Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

GENERAL TERMS

1. The undersigned represents that he/she has the power and authority to make and execute this form.
2. The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.
3. The undersigned agrees that publication with IEEE is subject to the policies and procedures of the IEEE PSPB Operations Manual.
4. In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall be null and void. In this case, IEEE will retain a copy of the manuscript for internal administrative/record-keeping purposes.
5. For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.
6. The author hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the author has obtained any necessary permissions. Where necessary, the author has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE

You have indicated that you DO wish to have video/audio recordings made of your conference presentation under terms and conditions set forth in "Consent and Release."

CONSENT AND RELEASE

1. In the event the author makes a presentation based upon the Work at a conference hosted or sponsored in whole or in part by the IEEE, the author, in consideration for his/her participation in the conference, hereby grants the IEEE the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive, in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the "Presentation"). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IEEE and live or recorded broadcast of the Presentation during or after the conference.
2. In connection with the permission granted in Section 1, the author hereby grants IEEE the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice and biographical information as part of the advertisement, distribution and sale of products incorporating the Work or Presentation, and releases IEEE from any claim based on right of privacy or publicity.

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Manlika Ratchagit — **Signature**

19-10-2020 — **Date (dd-mm-yyyy)**

Information for Authors

AUTHOR RESPONSIBILITIES

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE PSPB Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE's publishing policies may be found at http://www.ieee.org/publications_standards/publications/rights/authorrightsresponsibilities.html Authors are advised especially of IEEE PSPB Operations Manual section 8.2.1.B12: "It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it." Authors are also advised of IEEE PSPB Operations Manual section 8.1.1B: "Statements and opinions given in work published by the IEEE are the expression of the authors."

RETAINED RIGHTS/TERMS AND CONDITIONS

- Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
- Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author's personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale.
- Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use.The IEEE Intellectual Property Rights office must handle all such third-party requests.
- Authors whose work was performed under a grant from a government funding agency are free to fulfill any deposit mandates from that funding agency.

AUTHOR ONLINE USE

- **Personal Servers.** Authors and/or their employers shall have the right to post the accepted version of IEEE-copyrighted articles on their own personal servers or the servers of their institutions or employers without permission from IEEE, provided that the posted version includes a prominently displayed IEEE copyright notice and, when published, a full citation to the original IEEE publication, including a link to the article abstract in IEEE Xplore. Authors shall not post the final, published versions of their papers.
- **Classroom or Internal Training Use.** An author is expressly permitted to post any portion of the accepted version of his/her own IEEE-copyrighted articles on the author's personal web site or the servers of the author's institution or company in connection with the author's teaching, training, or work responsibilities, provided that the appropriate copyright, credit, and reuse notices appear prominently with the posted material. Examples of permitted uses are lecture materials, course packs, e-reserves, conference presentations, or in-house training courses.
- **Electronic Preprints.** Before submitting an article to an IEEE publication, authors frequently post their manuscripts to their own web site, their employer's site, or to another server that invites constructive comment from colleagues. Upon submission of an article to IEEE, an author is required to transfer copyright in the article to IEEE, and the author must update any previously posted version of the article with a prominently displayed IEEE copyright notice. Upon publication of an article by the IEEE, the author must replace any previously posted electronic versions of the article with either (1) the full citation to the

IEEE COPYRIGHT AND CONSENT FORM

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any wording of the form be changed. This form is intended for original material submitted to the IEEE and must accompany any such material in order to be published by the IEEE. Please read the form carefully and keep a copy for your files.

Parameter Identification of Stochastic Delay Differential Equations Using Differential Evolution
Ms. Manlika Ratchagit and Dr. Honglei Xu
2022 37th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)

COPYRIGHT TRANSFER

The undersigned hereby assigns to The Institute of Electrical and Electronics Engineers, Incorporated (the "IEEE") all rights under copyright that may exist in and to: (a) the Work, including any revised or expanded derivative works submitted to the IEEE by the undersigned based on the Work; and (b) any associated written or multimedia components or other enhancements accompanying the Work.

GENERAL TERMS

1. The undersigned represents that he/she has the power and authority to make and execute this form.
2. The undersigned agrees to indemnify and hold harmless the IEEE from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.
3. The undersigned agrees that publication with IEEE is subject to the policies and procedures of the IEEE PSPB Operations Manual.
4. In the event the above work is not accepted and published by the IEEE or is withdrawn by the author(s) before acceptance by the IEEE, the foregoing copyright transfer shall be null and void. In this case, IEEE will retain a copy of the manuscript for internal administrative/record-keeping purposes.
5. For jointly authored Works, all joint authors should sign, or one of the authors should sign as authorized agent for the others.
6. The author hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data or other material from the works of others, the author has obtained any necessary permissions. Where necessary, the author has obtained all third party permissions and consents to grant the license above and has provided copies of such permissions and consents to IEEE

You have indicated that you DO wish to have video/audio recordings made of your conference presentation under terms and conditions set forth in "Consent and Release."

CONSENT AND RELEASE

1. In the event the author makes a presentation based upon the Work at a conference hosted or sponsored in whole or in part by the IEEE, the author, in consideration for his/her participation in the conference, hereby grants the IEEE the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive, in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the "Presentation"). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IEEE and live or recorded broadcast of the Presentation during or after the conference.
2. In connection with the permission granted in Section 1, the author hereby grants IEEE the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice and biographical information as part of the advertisement, distribution and sale of products incorporating the Work or Presentation, and releases IEEE from any claim based on right of privacy or publicity.

BY TYPING IN YOUR FULL NAME BELOW AND CLICKING THE SUBMIT BUTTON, YOU CERTIFY THAT SUCH ACTION CONSTITUTES YOUR ELECTRONIC SIGNATURE TO THIS FORM IN ACCORDANCE WITH UNITED STATES LAW, WHICH AUTHORIZES ELECTRONIC SIGNATURE BY AUTHENTICATED REQUEST FROM A USER OVER THE INTERNET AS A VALID SUBSTITUTE FOR A WRITTEN SIGNATURE.

Manlika Ratchagit — **Signature**

27-05-2022 — **Date (dd-mm-yyyy)**

Information for Authors

AUTHOR RESPONSIBILITIES

The IEEE distributes its technical publications throughout the world and wants to ensure that the material submitted to its publications is properly available to the readership of those publications. Authors must ensure that their Work meets the requirements as stated in section 8.2.1 of the IEEE PSPB Operations Manual, including provisions covering originality, authorship, author responsibilities and author misconduct. More information on IEEE's publishing policies may be found at http://www.ieee.org/publications_standards/publications/rights/authorrightsresponsibilities.html Authors are advised especially of IEEE PSPB Operations Manual section 8.2.1.B12: "It is the responsibility of the authors, not the IEEE, to determine whether disclosure of their material requires the prior consent of other parties and, if so, to obtain it." Authors are also advised of IEEE PSPB Operations Manual section 8.1.1B: "Statements and opinions given in work published by the IEEE are the expression of the authors."

RETAINED RIGHTS/TERMS AND CONDITIONS

- Authors/employers retain all proprietary rights in any process, procedure, or article of manufacture described in the Work.
- Authors/employers may reproduce or authorize others to reproduce the Work, material extracted verbatim from the Work, or derivative works for the author's personal use or for company use, provided that the source and the IEEE copyright notice are indicated, the copies are not used in any way that implies IEEE endorsement of a product or service of any employer, and the copies themselves are not offered for sale.
- Although authors are permitted to re-use all or portions of the Work in other works, this does not include granting third-party requests for reprinting, republishing, or other types of re-use.The IEEE Intellectual Property Rights office must handle all such third-party requests.
- Authors whose work was performed under a grant from a government funding agency are free to fulfill any deposit mandates from that funding agency.

AUTHOR ONLINE USE

- **Personal Servers.** Authors and/or their employers shall have the right to post the accepted version of IEEE-copyrighted articles on their own personal servers or the servers of their institutions or employers without permission from IEEE, provided that the posted version includes a prominently displayed IEEE copyright notice and, when published, a full citation to the original IEEE publication, including a link to the article abstract in IEEE Xplore. Authors shall not post the final, published versions of their papers.
- **Classroom or Internal Training Use.** An author is expressly permitted to post any portion of the accepted version of his/her own IEEE-copyrighted articles on the author's personal web site or the servers of the author's institution or company in connection with the author's teaching, training, or work responsibilities, provided that the appropriate copyright, credit, and reuse notices appear prominently with the posted material. Examples of permitted uses are lecture materials, course packs, e-reserves, conference presentations, or in-house training courses.
- **Electronic Preprints.** Before submitting an article to an IEEE publication, authors frequently post their manuscripts to their own web site, their employer's site, or to another server that invites constructive comment from colleagues. Upon submission of an article to IEEE, an author is required to transfer copyright in the article to IEEE, and the author must update any previously posted version of the article with a prominently displayed IEEE copyright notice. Upon publication of an article by the IEEE, the author must replace any previously posted electronic versions of the article with either (1) the full citation to the

Re: Request for permission to reproduce my publication into my thesis

Mathematics Editorial Office <mathematics@mdpi.com>
Fri 23/09/2022 6:56 PM
To: Manlika Ratchagit <manlika.ratchagit@postgrad.curtin.edu.au>

Dear Dr. Manlika Ratchagit,

/Mathematics/ is an open access journal. The copyright of the published articles in the /Mathematics/ belongs to the author. You own the copyright of the article and do not need get the without permission of the editorial department. Wish you all the best.

Kind regards,

Ms. Patty Hu
Managing Editor
Quick Contact in Skype:live:f5eda186f7ff1c2d
Mathematics (http://www.mdpi.com/journal/mathematics)
--
Mathematics is Recruiting Topical Advisory Panel (TAP) Members, Please apply here:
https://www.mdpi.com/journal/mathematics/topical_advisory_panel_application

On 2022/9/22 20:23, Manlika Ratchagit wrote:
> CAUTION - EXTERNAL: This email originated from outside of MDPI
> organisation. BE CAUTIOUS especially to click links or open
> attachments.
>
> Dear *Editor,*
>
> My name is Ms Manlika Ratchagit. My manuscript has already been
> published in Mathematics. Vol.10, No.19, 2022. The digital object
> identifier of my manuscript is 10.3390/math10193447.
>
> I am the first author of this paper. I would like to reproduce an
> extract of this work in a doctoral's thesis which I am currently
> undertaking at Curtin University in Perth, Western Australia. I am
> carrying out this research in my own right and have no association
> with any commercial organisation or sponsor.
>
> Once completed, the thesis will be made available in online form via
> Curtin University's Institutional Repository espace
> (http://espace.curtin.edu.au <http://espace.curtin.edu.au>). The
> material will be provided strictly for educational purposes and on a
> non-commercial basis.
>
> I look forward to hearing from you and thank you in advance for your
> consideration of my request.
>
> Sincerely yours,
>
> Manlika Ratchagit PhD student (Mathematics & Statistics)
>

> School of Electrical Engineering, Computing and Mathematical
> Sciences Curtin University
>
> Bentley 6102, Perth, Western Australia
>
> Email: manlika.ratchagit@postgrad.curtin.edu.au
> <mailto:manlika.ratchagit@postgrad.curtin.edu.au>
>
>

Appendix C

Statement of Attribution

Statement of Attribution

Chapters 3 to 5 are based upon several works published in journals and presented at conferences throughout the author's PhD. These chapters are reproductions of published manuscripts, except for formatting consistent with the thesis.

The research presented in Chapter 3 was published within the peer-reviewed journal, '*CMES-Computer Modeling in Engineering & Sciences*' on February 1st, 2020:

> **Ratchagit, M.,** Wiwatanapataphee, B. and Dokuchaev, N., 2020. The m-delay Autoregressive Model with Application. *CMES-Computer Modeling in Engineering and Sciences*, *122*(2), pp. 487-504, doi:10.32604/cmes.2020.08865.

> All authors provided the concept and designed the methodology; I collected and analysed the data; I wrote the manuscript; Nikolai Dokuchaev supervised the research; all authors contributed to the revisions of the manuscript and approved the final version.

The research presented in Chapter 4 was published in IEEE conference proceedings, '3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)', Indonesia, Dec. 2020:

> **Ratchagit, M.,** Wiwatanapataphee, B. and Nur, D., 2020. On Parameter Estimation of Stochastic Delay Difference Equation using the Two m-delay Autoregressive Coefficients. In *2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, pp. 310-314, doi:10.1109/ISRITI51436.2020.9315414.

> All authors conceived the ideas and designed the methodology; I conducted the experiments and analysed the data; I wrote the manuscript; Benchawan Wiwatanapataphee supervised the research; all

authors contributed to the revisions of the manuscript and approved the final version.

Another research in Chapter 4 is inpress in IEEE conference proceedings, ’37 th International Technical Conference on Circuits/Systems, Computers, and Communications (ITC-CSCC) ’, Thailand, Jul. 2022:

Ratchagit, M., and Xu, H., 2022, Parameter Identification of Stochastic Delay Differential Equations Using Differential Evolution. *In 2022 37th International Technical Conference on Circuits/Systems, Computers, and Communications (ITC-CSCC).*

All authors contributed to manuscript conceptualization, editing, review for publication; I conducted the experiments, analysed the data, and wrote the manuscript; Honglei Xu supervised the research; all authors contributed to the revisions of the manuscript and approved the final manuscript.

The research presented in Chapter 5 was published within the peer-reviewed journal, ‘Mathematics’ on September 22nd, 2022:

Ratchagit, M., and Xu, H., 2022. A Two-Delay Combination Model for Stock Price Prediction. *Mathematics*, 10(19), pp. 3447, doi:10.3390/math10193447.

All authors contributed to manuscript conceptualization, editing, review for publication; I conducted the experiments, analysed the data, and drafted the manuscript; Honglei Xu supervised the research; all authors contributed to the revisions of the manuscript and approved the final manuscript.

Student name: Manlika Ratchagit

Signature: **Date:**

Co-authors:

Name: Prof Nikolai Dokuchaev

Signature: **Date:**

Name: Dr Honglei Xu

Signature: **Date:**

Name: Assoc/Prof Benchawan Wiwatanapataphee

Signature: **Date:**

Name: Dr Darfiana Nur

Signature: **Date:**

Bibliography

[1] Miglani A, Kumar N. Deep learning models for traffic flow prediction in autonomous vehicles: A review, solutions, and challenges. *Vehicular Communications*. 2019;20:100184. doi:10.1016/j.vehcom.2019.100184.

[2] Phung VH, Rhee EJ. A deep learning approach for classification of cloud image patches on small datasets. *Journal of information and communication convergence engineering*. 2018;16(3):173-8. doi:10.6109/jicce.2018.16.3.173.

[3] Chen Y, Fang R, Liang T, et al. Stock Price Forecast Based on CNN-BiLSTM-ECA Model. *Scientific Programming*. 2021;2021:2446543. doi:10.1155/2021/2446543.

[4] Hajirahimi Z, Khashei M. Hybrid structures in time series modeling and forecasting: A review. *Engineering Applications of Artificial Intelligence*. 2019;86:83-106. doi: 10.1016/j.engappai.2019.08.018.

[5] Nielsen A. *Practical time series analysis: Prediction with statistics and machine learning*. O'Reilly Media, Incorporated; 2019. Accessed on October 15, 2021. https://books.google.co.th/books?id=uq0avgEACAAJ.

[6] Attoue N, Shahrour I, Younes R. Smart building: Use of the artificial neural network approach for indoor temperature forecasting. *Energies*. 2018;11(2):395. doi:10.3390/en11020395.

[7] Aghelpour P, Mohammadi B, Biazar S. Long-term monthly average temperature forecasting in some climate types of Iran, using the mod-

els SARIMA, SVR, and SVR-FA. *Theoretical and Applied Climatology*. 2019;138(3):1471-80. doi:10.1007/s00704-01902905w.

[8] Cifuentes J, Marulanda G, Bello A, Reneses J. Air temperature forecasting using machine learning techniques: a review. *Energies*. 2020;13(16):4215. doi:10.3390/en13164215.

[9] Zhao Q, Liu Y, Ma X, Yao W, Yao Y, X L. An improved rainfall forecasting model based on GNSS observations. *IEEE Transactions on Geoscience and Remote Sensing*. 2020;58(7):4891-900. doi:10.1109/TGRS.2020.2968124.

[10] Ridwan WM, Sapitang M, Aziz A, Kushiar KF, Ahmed AN, El-Shafie A. Rainfall forecasting model using machine learning methods: Case study Terengganu, Malaysia. *Ain Shams Engineering Journal*. 2021;12(2):1651-63. doi:10.1016/j.asej.2020.09.011.

[11] Jiao X, Li G, Chen JL. Forecasting international tourism demand: a local spatiotemporal model. *Annals of Tourism Research*. 2020;83:102937. doi:10.1016/j.annals.2020.102937.

[12] Xie G, Qian Y, Wang S. A decomposition-ensemble approach for tourism forecasting. *Annals of Tourism Research*. 2020;81:102891. doi:10.1016/j.annals.2020.102891.

[13] Mondoloni S, Rozen N. Aircraft trajectory prediction and synchronization for air traffic management applications. *Progress in aerospace sciences*. 2020;119:100640. doi:10.1016/j.paerosci.2020.100640.

[14] Solvoll G, Mathisen TA, Welde M. Forecasting air traffic demand for major infrastructure changes. *Research in transportation economics*. 2020;82:100873. doi:10.1016/j.retrec.2020.100873.

[15] Finnis J, Shewmake J, Neis B, Telford D. Marine forecasting and fishing safety: improving the fit between forecasts and harvester needs. *Journal of agromedicine*. 2019;24(4):324-32. doi:10.1080/1059924X.2019.1639576.

[16] Wen J, Yang J, Jiang B, Song H, H W. Big data driven marine environment information forecasting: a time series prediction network. *IEEE Transactions on Fuzzy Systems*. 2020;29(1):4-18. doi:10.1109/TFUZZ.2020.3012393.

[17] Hanifi S, Liu X, Lin Z, Lotfian S. A critical review of wind power forecasting methodspast, present and future. *Energies*. 2020;13(15):3764. doi:10.3390/en13153764.

[18] Ahmadi M, Khashei M. Current status of hybrid structures in wind forecasting. *Engineering applications of artificial intelligence*. 2021;99:104133. doi:10.1016/j.engappai.2020.104133.

[19] Jeon S, Hong B, V C. Pattern graph tracking-based stock price prediction using big data. *Future Generation Computer Systems*. 2018;80:171-87. doi:10.1016/j.future.2017.02.010.

[20] Lu W, Li J, Wang J, Qin L. A CNN-BiLSTM-AM method for stock price prediction. *Neural Computing and Applications*. 2020;33(3):47414753 . doi:10.1007/s00521-02005532z.

[21] Yu P, Yan X. Stock price prediction based on deep neural networks. *Neural Computing and Applications*. 2020;32(6):1609-28. doi:10.1007/s00521-01904212x.

[22] Lei Y, Li N, Guo L, Li N, Yan T, Lin J. Machinery health prognostics: A systematic review from data acquisition to RUL prediction. *Mechanical systems and signal processing*. 2018;104:799-834. doi:10.1016/j.ymssp.2017.11.016.

[23] Liu Y, Xv J, Yuan H, Lv J, Ma Z. Health assessment and prediction of overhead line based on health index. *IEEE Transactions on Industrial Electronics*. 2018;66(7):5546-57. doi:10.1109/TIE.2018.2868028.

[24] Rong G, Mendez A, Assi E, Zhao B, Sawan M. Artificial intelligence in healthcare: review and prediction case studies. *Engineering*. 2020;6(3):291-301. doi:10.1016/j.eng.2019.08.015.

[25] Yang P, Yang G, Qi J, Sheng B, Yang Y, Zhang S, et al. The effect of multiple interventions to balance healthcare demand for controlling COVID-19 outbreaks: a modelling study. *Scientific reports*. 2021;11(1):1-3. doi:10.1038/s41598-02182170y.

[26] Uddin S, Khan A, Hossain M, Moni M. Comparing different supervised machine learning algorithms for disease prediction. *BMC medical informatics and decision making*. 2019;19(1):1-6. doi:10.1186/s12911-01910048.

[27] Mohan S, Thirumalai C, Srivastava G. Effective heart disease prediction using hybrid machine learning techniques. *IEEE access*. 2019;7:81542-54. doi:10.1109/ACCESS.2019.2923707.

[28] Huang Y, Xu C, Ji M, Xiang W, He D. Medical service demand forecasting using a hybrid model based on ARIMA and self-adaptive filtering method. *BMC Medical Informatics and Decision Making*. 2020;20(1):1-4. doi:10.1186/s12911-020012561.

[29] Newbold P, Granger C. Experience with forecasting univariate time series and the combination of forecasts. *Journal of the Royal Statistical Society: Series A (General)*. 1974;137(2):131-46. doi:10.2307/2344546.

[30] Lu W, Li J, Wang J, Qin L. A CNN-BiLSTM-AM method for stock price prediction. *Neural Computing and Applications*. 2021;33(10):4741-53. doi:10.1007/s00521-02005532z.

[31] Moosa I. *Univariate Time Series Techniques. In: Exchange Rate Forecasting: Techniques and Applications. Finance and Capital Markets Series*. Palgrave Macmillan; 2000. Accessed on November 01, 2021. doi: 10.1057/97802303790083.

[32] Zhou J, Lee S, Wang X, Li Y, Wu WKK, Liu T, et al. Development of a multivariable prediction model for severe COVID-19 disease: a population-based study from Hong Kong. NPJ digital medicine. *NPJ digital medicine*. 2021;4(1):1-9. doi:10.1038/s41746-02200586w.

[33] Nguyen T, Kashani A, Ngo T, Bordas S. Deep neural network with highorder neuron for the prediction of foamed concrete strength. *ComputerAided Civil and Infrastructure Engineering*. 2019;34(4):316-32. doi:10.1111/mice.12422.

[34] Bouktif S, Fiaz A, Awad M. Augmented textual features-based stock market prediction. *IEEE Access*. 2020;8:40269-82. doi:10.1109/ACCESS.2020.2976725.

[35] Gui G, Liu F, Sun J, Yang J, Zhou Z, Zhao D. Flight delay prediction based on aviation big data and machine learning. *IEEE Transactions on Vehicular Technology*. 2019;69(1):140-50. doi:10.1109/TVT.2019.2954094.

[36] Yu B, Guo Z, Asian S, Wang H, Chen G. Flight delay prediction for commercial air transport: A deep learning approach. *Transportation Research Part E: Logistics and Transportation Review*. 2019;125(1):203-21. doi:10.1016/j.tre.2019.03.013.

[37] Qu J, Zhao T, Ye M, Li J, Liu C. Flight delay prediction using deep convolutional neural network based on fusion of meteorological data. *Neural Processing Letters*. 2020;152(2):1464-84. doi:10.1007/s11063-020103184.

[38] Bello S, Afolabi RF, Ajayi DT, Sharma T, Owoeye DO, Oduyoye O, et al. Empirical evidence of delays in diagnosis and treatment of pulmonary

tuberculosis: systematic review and meta-regression analysis. *BMC public health*. 2019;19(1):1-11. doi:10.1186/s12889-01970264.

[39] Iqbal S, Saidullah S, Ahmed RI, Khan MAA, Ahmed NISAR, KHAN MF. Factors Contributing to Delayed Diagnosis of Congenital Heart Disease in Pediatric Population. *Pakistan Journal of Medical & Health Sciences*. 2021;15(5):251-68. doi:10.53350/pjmhs211551488.

[40] Hoyer N, Prior TS, Bendstrup E, Wilcke T, Shaker SB. Risk factors for diagnostic delay in idiopathic pulmonary fibrosis. *Respiratory research*. 2019;20(1):1-9. doi:10.1186/s12931-01910760.

[41] Moeletsi M, Mellaart E, Mpandeli N, Hamandawana H. The use of rainfall forecasts as a decision guide for small-scale farming in Limpopo Province, South Africa. *The Journal of Agricultural Education and Extension*. 2013;19(2):133-45. doi:10.1080/1389224X.2012.734253.

[42] Ustaoglu B, Cigizoglu H, Karaca M. Forecast of daily mean, maximum and minimum temperature time series by three artificial neural network methods. *Meteorological Applications: A journal of forecasting, practical applications, training techniques and modelling*. 2008;15(4):431-45. doi:10.1002/met.83.

[43] Box G, Jenkins G, Reinsel G. Time Series Analysis [internet] 2nd edition. 4th ed. New York, USA: John Wiley & Sons; 2008 [cited 2021 Jun 9]. doi:10.1002/9781118619193.

[44] Ip E, Zhang Q, Sowinski T, Simpson S. Analysis of feedback mechanisms with unknown delay using sparse multivariate autoregressive method. *PloS one*. 2015;10(8):e0131371. doi:10.1371/journal.pone.0131371.

[45] Amo-Salas M, López-Fidalgo J, Pedregal D. Experimental designs for autoregressive models applied to industrial maintenance. *Reliability engineering & system safety*. 2015;133:87-94. doi:10.1016/j.ress.2014.09.003.

[46] Acedański J. Forecasting industrial production in Poland a comparison of different methods. *Ekonometria*. 2013;39:40-51.

[47] Sharafi M, Ghaem H, Tabatabaee HR, Faramarzi H. Forecasting the number of zoonotic cutaneous leishmaniasis cases in south of Fars province, Iran using seasonal ARIMA time series method. *Asian Pacific journal of tropical medicine*. 2017;10(1):79-86. doi:10.1016/j.apjtm.2016.12.007.

[48] Tsitsika EV, Maravelias CD, Haralabous J. Modeling and forecasting pelagic fish production using univariate and multivariate ARIMA models. *Fisheries science*. 2007;73(5):979-88. doi:10.1111/j.1444-2906.2007.01426.x.

[49] Tealab A. Time series forecasting using artificial neural networks methodologies: A systematic review. *Future Computing and Informatics Journal*. 2018;3(2):334-40. doi:10.1016/j.fcij.2018.10.003.

[50] Taskaya-Temizel T, Casey M. A comparative study of autoregressive neural network hybrids. *Neural Networks*. 2005;18(5-6):781-9. doi:10.1016/j.neunet.2005.06.003.

[51] Pan F, Zhang H, Xia M. A hybrid time-series forecasting model using extreme learning machines. 2009 Second International Conference on Intelligent Computation Technology and Automation. 2009[cited 2021 Jun 9];1:933-6. doi:10.1109/ICICTA.2009.232.

[52] Qi M, Zhang G. An investigation of model selection criteria for neural network time series forecasting. *European Journal of Operational Research*. 2001;132(3):666-80. doi:10.1016/S0377-2217(00)00171-5.

[53] Zhang G. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*. 2003;50:159-75. doi:10.1016/S0925-2312(01)00702-0.

[54] Saxena P, Sharma K, Easo S. Forecasting enrollments based on fuzzy time series with higher forecast accuracy rate. *Int J Computer Technology & Applications*. 2012;3(3):957-61. doi:10.1.1.643.4986.

[55] Cryer J, Chan K. Time series analysis: with applications in R. 4th ed. New York, USA: Springer Science & Business Media; 2008 [cited 2021 Jun 9].

[56] Broersen P. Finite-Sample Bias Propagation in the Yule-Walker Method of Autoregressive Estimation. In: 17th World Congress the International Federation of Automatic Control ; July 6-11, 2008; Seoul, Korea. Accessed May 04, 2022. p. 2744-9. doi:10.3182/20080706-5KR1001.00462.

[57] Hyndman R, Koehler A. Another look at measures of forecast accuracy. *International Journal of Forecasting*. 2006;22(4):679-88. doi: 10.1016/j.ijforecast.2006.03.001.

[58] Maddala G, Wu S. A comparative study of unit root tests with panel data and a new simple test. *Oxford Bulletin of Economics and statistics*. 1999;61(S1):631-52. doi:10.1111/1468-0084.0610s1631.

[59] Hassler U. (Mis) specification of long memory in seasonal time series. *Journal of Time Series Analysis*. 1994;15(1):19-30. doi:10.1111/j.1467-9892.1994.tb00174.x.

[60] Pourahmadi M. Foundations of time series analysis and prediction theory. New York, USA: John Wiley & Sons; 2001.

[61] Pena-Sanchez Y, Merigaud A, Ringwood J. Short-term forecasting of sea surface elevation for wave energy applications: The autoregressive model revisited. *IEEE Journal of Oceanic Engineering*. 2018;45(2):462-71. doi:10.1109/JOE.2018.2875575.

[62] Kandula S, Shaman J. Near-term forecasts of influenza-like illness: An evaluation of autoregressive time series approaches. *Epidemics*. 2019;27:41-51. doi:10.1016/j.epidem.2019.01.002.

[63] Maleki M, Mahmoudi M, Wraith D, Pho K. Time series modelling to forecast the confirmed and recovered cases of COVID-19. *Travel medicine and infectious disease*. 2020;37:101742. doi:10.1016/j.tmaid.2020.101742.

[64] Tsay RS. Analysis of financial time series [internet]. 2nd ed. New York, USA: Hoboken, N.J. : Wiley; 2005.

[65] Mikosch T. Elementary Stochastic Calculus with Finance in View [internet]. Singapore: World Scientific Publ: Singapore; 1998.

[66] Lima LS, Miranda LLB. Price dynamics of the financial markets using the stochastic differential equation for a potential double well. *Physica A: Statistical Mechanics and its Applications*. 2018;490:828-33. doi:10.1016/j.physa.2017.08.106.

[67] Bayer C, Qiu J, Yao Y. Pricing options under rough volatility with backward SPDEs. *SIAM Journal on Financial Mathematics*. 2022;13(1):179-212. doi:10.1137/20M1357639.

[68] Black F, Scholes M. The pricing of options and corporate liabilities. *Journal of political economy*. 1973;81(3):637-54. doi:10.1086/260062.

[69] Zheng Y. *Asset pricing based on stochastic delay differential equation* [dissertation]. Iowa State University; 2015. Accessed March 21, 2021. https://www.proquest.com/docview/1708627243?accountid=10382&pq-origsite=primo.

[70] Marathe R, Ryan S. On the validity of the geometric Brownian motion assumption. *The Engineering Economist*. 2005;50(2):159-92. doi:10.1080/00137910590949904.

[71] Anagnostou I, Kandhai D. Risk factor evolution for counterparty credit risk under a hidden markov model. *Risks*. 2019;7(2):66. doi:10.3390/risks7020066.

[72] Tambue A, Brown E, Mohammed S. A stochastic delay model for pricing debt and equity: Numerical techniques and applications. *Communications in Nonlinear Science and Numerical Simulation*. 2015;20(1):281-97. doi:10.1016/j.cnsns.2014.05.010.

[73] Eissa M, B T. A stochastic corporate claim value model with variable delay. *Journal of Physics: Conference Series*. 2018;1053(1):012018. doi:10.1088/1742-6596/1053/1/012018.

[74] Lee K, Lee K, Park S. A financial market of a stochastic delay equation. *Bulletin of the Korean Mathematical Society*. 2019;56(5):1129-41. doi:10.4134/BKMS.b180848.

[75] Ernst P, Soleymani F. A Legendre-based computational method for solving a class of Itô stochastic delay differential equations. *Numerical Algorithms*. 2019;80(4):1267-82. doi:10.1007/s11075-0180526y.

[76] Lahmiri S. Modeling and predicting historical volatility in exchange rate markets. *Physica a-Statistical Mechanics and Its Applications*. 2017;471:387-95. doi: 10.1016/j.physa.2016.12.061.

[77] Masset P. Volatility stylized facts. *Available at SSRN 1804070*. 2011:1-91. doi: 10.2139/ssrn.1804070.

[78] Andersen T, Bollerslev T, Lange S. Forecasting financial market volatility: Sample frequency vis-a-vis forecast horizon. *Journal of empirical finance*. 1999;6(5):457-77. doi: 10.1016/S0927-5398(99)00013-4.

[79] Zhang L, Mykland P, Aït-Sahalia Y. A tale of two time scales: Determining integrated volatility with noisy high-frequency data. *Jour-*

nal of the American Statistical Association. 2005;100(472):1394-411. doi; 10.1198/016214505000000169.

[80] Luong C, Dokuchaev N. Modeling Dependency Of Volatility On Sampling Frequency Via Delay Equations. *Annals of Financial Economics*. 2016;11(2):1650007. doi: 10.1142/S201049521650007X.

[81] Andersen T, Bollerslev T, Christoffersen P, Diebold F. Volatility forecast using hybrid Neural Network models. *Handbook of economic forecasting*. 2006;1:777-878. doi: 10.1016/S1574-0706(05)01015-3.

[82] Shickel B, Tighe P, Bihorac A, Rashidi P. Deep EHR: a survey of recent advances in deep learning techniques for electronic health record (EHR) analysis. *IEEE journal of biomedical and health informatics*. 2017;22(5):1589-604. doi:10.1109/JBHI.2017.2767063.

[83] Salim HK, Stewart RA, Sahin O, Dudley M. Drivers, barriers and enablers to end-of-life management of solar photovoltaic and battery energy storage systems: A systematic literature review. *Journal of cleaner production*. 2019;211:537-54. doi:10.1016/j.jclepro.2018.11.229.

[84] Jiang T, Gradus J, Rosellini A. Supervised machine learning: a brief primer. *Behavior Therapy*. 2020;51(5):675-87. doi:10.1016/j.beth.2020.05.002.

[85] Jin Y, Xu W, Wang P, Yan J. SAE Network: A Deep Learning Method for Traffic Flow Prediction. In: 2018 International Conference on Information, Cybernetics, and Computational Social Systems, ICCSS 2018; 2018. p. 241-6. doi:10.1109/ICCSS.2018.8572451.

[86] Zhao L, Zhou YH, Lu HP, Fujita H. Parallel computing method of deep belief networks and its application to traffic flow prediction. *Knowledge-Based Systems*. 2019;163:972-87. doi:10.1016/j.knosys.2018.10.025.

[87] Alzubaidi L, Zhang J, Humaidi A, Al-Dujaili A, Duan Y, Al-Shamma O, et al. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of big Data*. 2021;8(1):1-74. doi:10.1186/s40537-021004448.

[88] Mansoor M, Grimaccia F, Leva S, Mussetta M. Comparison of echo state network and feed-forward neural networks in electrical load forecasting for demand response programs. *Mathematics and Computers in Simulation*. 2021;184:105-17. doi:10.1016/j.matcom.2020.07.011.

[89] Zhang D, Zhang N, Ye N, Fang J, Han X. Hybrid learning algorithm of radial basis function networks for reliability analysis. *IEEE Transactions on reliability*. 2020[cited 2021 Jun 3];70(3):887-900. doi:10.1109/TR.2020.3001232.

[90] Liang Y, Niu D, Hong WC. Short term load forecasting based on feature extraction and improved general regression neural network model. *Energy*. 2019;166:653-63. doi:10.1016/j.energy.2018.10.119.

[91] Yin C, Rosendahl L, Luo Z. Methods to improve prediction performance of ANN models. *Simulation Modelling Practice and Theory*. 2003;11(3-4):211-22. doi:10.1016/S0360-8352(02)00036-0.

[92] Nguyen TT, Yoon S. A novel approach to short-term stock price movement prediction using transfer learning. *Applied Sciences*. 2019;9(22):4745. doi:10.3390/app9224745.

[93] Rajabi S, Roozkhosh P, Farimani N. MLP-based Learnable Window Size for Bitcoin price prediction. *Applied Soft Computing*. 2022;30:109584. doi:10.1016/j.asoc.2022.109584.

[94] Peng Z, Khan F, Khan F, Shaikh P, Yonghong D, Ullah I, et al. An Application of Hybrid Models for Weekly Stock Market Index Prediction:

Empirical Evidence from SAARC Countries. *Complexity*. 2021;2021:1-10. doi:10.1155/2021/5663302.

[95] Zhang R, Zhang Z, Wang D, Du M. Financial Distress Prediction with a Novel Diversity-Considered GA-MLP Ensemble Algorithm. *Neural Processing Letters*. 2022;54(2):1175-94. doi:10.1007/s11063-021106749.

[96] EL-SAID M, ZAKI RM, EID MM. Advanced meta-heuristics, convolutional neural networks, and feature selectors for efficient COVID-19 X-ray chest image classification. *IEEE Access*. 2021;9:36019-37. doi: 10.1109/ACCESS.2021.3061058.

[97] Riedmiller M. Advanced supervised learning in multi-layer perceptronsfrom backpropagation to adaptive learning algorithms. *Computer Standards & Interfaces*. 1994;3(16):265-78. doi:10.1016/0920-5489(94)90017-5.

[98] Apicella A, Donnarumma F, Isgró F, Prevete R. A survey on modern trainable activation functions. *Neural Networks*. 2021;138:14-32. doi:10.1016/j.neunet.2021.01.026.

[99] Hoseinzade E, Haratizadeh S. CNNpred: CNN-based stock market prediction using a diverse set of variables. *Expert Systems with Applications*. 2019;129:273-85. doi:10.1016/j.eswa.2019.03.029.

[100] Ben Fredj H, Bouguezzi S, Souani C. Face recognition in unconstrained environment with CNN. *The Visual Computer*. 2021;37(2):217-26. doi:10.1007/s00371-020017949.

[101] Nguyen N, Nguyen D. Parameter estimation of Pendubot model using modified differential evolution algorithm. *International Journal of Modelling and Simulation*. 2019;39(3):157-65. doi: 10.1080/02286203.2018.1525938.

[102] Dang C, Moreno-García M, Prieta F. An Approach to Integrating Sentiment Analysis into Recommender Systems. *Sensors*. 2019;21(16):5666. doi:10.3390/s21165666.

[103] Anwar S, Majid M, Qayyum A, Awais M, Alnowami M, MK K. Medical image analysis using convolutional neural networks: a review. *Journal of Medical Systems*. 2018;42(11):1-3. doi:10.1007/s10916-01810881.

[104] Lv M, Hong Z, Chen L, Chen T, Zhu T, Ji S. Temporal multi-graph convolutional network for traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*. 2020;22(6):3337 3348. doi:10.1109/TITS.2020.2983763.

[105] Liang Y, Lin Y, Lu Q. Forecasting gold price using a novel hybrid model with ICEEMDAN and LSTM-CNN-CBAM. *Expert Systems with Applications*. 2022;206:117847. doi:10.1016/j.eswa.2022.117847.

[106] Kanwal A, Lau M, Ng S, Sim K, Chandrasekaran S. BiCuDNNLSTM-1dCNNA hybrid deep learning-based predictive model for stock price prediction. *Expert Systems with Applications*. 2022;202:117123. doi:10.1016/j.eswa.2022.117123.

[107] Cao J, Wang J. Stock price forecasting model based on modified convolution neural network and financial time series analysis. *International Journal of Communication System*. 2019;32(12):e3987. doi:10.1002/dac.3987.

[108] Storm H, Baylis K, Heckelei T. Machine learning in agricultural and applied economics. *European Review of Agricultural Economics*. 2020;47(3):849-92. doi:10.1093/erae/jbz033.

[109] Liu Y, Gong C, Yang L, Chen Y. DSTP-RNN: A dual-stage two-phase attention-based recurrent neural network for long-term and multivariate time series prediction. *Expert Systems with Applications*. 2020;143:113082. doi:10.1016/j.eswa.2019.113082.

[110] Liu X, Xu Q, N W. A survey on deep neural network-based image captioning. *The Visual Computer*. 2019;35(3):445-70. doi:10.1007/s00371-0181566y.

[111] Wang X, Chen C, Z X. Domain-specific machine translation with recurrent neural network for software localization. *Empirical Software Engineering*. 2017;24(6):3514-45. doi:10.1007/s10664-01909702z.

[112] Shin D, Choi K, Kim C. Deep learning model for prediction rate improvement of stock price using RNN and LSTM. *The Journal of Korean Institute of Information Technology*. 2017;15(10):9-16. doi:10.14801/jkiit.2017.15.10.9.

[113] Hochreiter S, Schmidhuber J. Long short-term memory. *Neural computation*. 1997;9(8):1735-80. doi:10.1162/neco.1997.9.8.1735.

[114] Chen Y, Lin W, Wang J. A dual-attention-based stock price trend prediction model with dual features. *IEEE Access*. 2019;7:148047-58. doi: 10.1109/ACCESS.2019.2946223.

[115] Qian F, Chen X. Stock prediction based on LSTM under different stability. In: 2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA); April 12-15, 2019; Chengdu, China. Accessed May 04, 2022. p. 483-6. doi:10.1109/ICCCBDA.2019.8725709.

[116] Sharaf M, Hemdan E, El-Sayed A, El-Bahnasawy N. StockPred: a framework for stock Price prediction. *Multimedia Tools and Applications*. 2021;80:17923-54. doi:10.1007/s11042-021105798.

[117] Greff K, Srivastava RK, Koutnk J, Steunebrink BR, Schmidhuber J. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*. 2016;28(10):2222 2232. doi: 10.1109/TNNLS.2016.2582924.

[118] Xiao Y, Yin Y. Hybrid LSTM neural network for short-term traffic flow prediction. *Information*. 2019;10(3):105. doi:10.3390/info10030105.

[119] Shahid F, Zameer A, Muneeb M. A novel genetic LSTM model for wind power forecast. *Energy*. 2021;223:120069. doi:10.1016/j.energy.2021.120069.

[120] Fernando T, Denman S, Sridharan S, Fookes C. Soft+ hard-wired attention: An lstm framework for human trajectory prediction and abnormal event detection. *Neural networks*. 2018;108:466-78. doi:10.1016/j.neunet.2018.09.002.

[121] Bathla G, Rani R, Aggarwal H. Stocks of year 2020: prediction of high variations in stock prices using LSTM. *Multimedia Tools and Applications*. 2022;2022:1-7. doi:10.1007/s11042-022123905.

[122] Bhandari H, Rimal B, Pokhrel N, Rimal R, Dahal K, Khatri R. Predicting stock market index using LSTM. *Machine Learning with Applications*. 2022;2022:100320. doi:10.1016/j.mlwa.2022.100320.

[123] Ahmad F, Singh P. Stock Price Prediction using ML and LSTM based Deep Learning models. *Journal of Informatics Electrical and Electronics Engineering (JIEEE)*. 2022;3(1):1-3. doi:0.54060/JIEEE/003.01.008.

[124] Liu Y, Ma Q, Liu H, Guo Z. Public Attitudes and Influencing Factors towards COVID-19 Vaccination for Adolescents/Children: A Scoping Review. *Public Health*. 2022;205:169-81. doi:10.1016/j.puhe.2022.02.002.

[125] Lin Y, Liao Q, Lin Z, Tan B, Yu Y. A novel hybrid model integrating modified ensemble empirical mode decomposition and LSTM neural network for multi-step precious metal prices prediction. *Resources Policy*. 2022;78:102884. doi:10.1016/j.resourpol.2022.102884.

[126] Liu H, Qi L, Sun M. Short-Term Stock Price Prediction Based on CAE-LSTM Method. *Wireless Communications and Mobile Computing)*. 2022;2022:1-7. doi:10.1155/2022/4809632.

[127] Gers FA, Eck D, Schmidhuber J. Applying LSTM to time series predictable through time-window approaches. In: The 12th Italian Workshop on Neural Nets; May 17-19, 2001; Vietri sul Mare, Salerno, Italy. Accessed Dec 20, 2022. p. 193-200. doi:10.1007/978 − 1447102199_20.

[128] Ma X, Tao Z, Wang Y, Yu H, Wang Y. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C: Emerging Technologies*. 2015;54:187-97. doi:10.1016/j.trc.2015.03.014.

[129] Saud A, Shakya S. Analysis of look back period for stock price prediction with RNN variants: A case study on banking sector of NEPSE. *Procedia Computer Science*. 2020;167:788-98. doi:10.1016/j.procs.2020.03.419.

[130] Kang H, Yang S, Huang J, Oh J. Time series prediction of wastewater flow rate by bidirectional LSTM deep learning. *International Journal of Control, Automation and Systems*. 2020;12(18):3023-30. doi:10.1007/s12555-01909846.

[131] Wu N, Green B, Ben X, O Banion S. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv preprint*. 2020;2020:1-10. arXiv:2001.08317.

[132] Fan C, Wang J, Gang W, Li S. Assessment of deep recurrent neural network-based strategies for short-term building energy predictions. *Applied energy*. 2019;236:700-10. doi:10.1016/j.apenergy.2018.12.004.

[133] Marino DL, Amarasinghe K, Manic M. Building energy load forecasting using deep neural networks. In: The 42nd Annual Conference of the

IEEE Industrial Electronics Society; October 23-26, 2016; Florence, Italy. Accessed Dec 22, 2022. p. 7046-51. doi:10.1109/IECON.2016.7793413.

[134] Vinayakumar R, Soman KP, Poornachandran P. Applying deep learning approaches for network traffic prediction. In: International Conference on Advances in Computing, Communications and Informatics (ICACCI); September 13-16, 2017; Udupi, India. Accessed Dec 23, 2022. p. 2353-8. doi:10.1109/ICACCI.2017.8126198.

[135] Ratchagit M, Wiwatanapataphee B, Dokuchaev N. The m-delay Autoregressive Model with Application. *CMES-Computer Modeling in Engineering and Science*. 2020;122(2):487-504. doi:10.32604/cmes.2020.08865.

[136] Marquardt DW. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*. 1963;11(2):431 441. doi:10.1137/0111030.

[137] Razali N, Wah Y. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of statistical modeling and analytics*. 2011;2(1):21-33.

[138] Son K, Lee WS, Lee KB. Prediction of the learning curves of 2 dental CAD software programs. *The Journal of Prosthetic Dentistry*. 2019;121(1):95-100. doi: 10.1016/j.prosdent.2018.01.004.

[139] Dickie J, Nandi A. A comparative study of AR order selection methods. *Signal Processing*. 1994;40(2-3):239-55. doi:10.1016/0165-1684(94)90071-X.

[140] Akaike H. Statistical predictor identification. *Annals of the institute of Statistical Mathematics*. 1970;22(1):203-17. doi:10.1007/BF02506337.

[141] Burshtein D, Weinstein E. Some relations between the various criteria for autoregressive model order determination. *IEEE trans-*

actions on acoustics, speech, and signal processing. 1985;33(4):1017-9. doi:10.1109/TASSP.1985.1164656.

[142] Ratchagit M, Wiwatanapataphee B, Nur D. On Parameter Estimation of Stochastic Delay Difference Equation using the Two *m*-delay Autoregressive Coefficients. In: 2020 3rd International Seminar on Research of Information Technology and Intelligent Systems (ISRITI); December 10-11, 2020; Yogyakarta, Indonesia. Accessed May 04, 2022. p. 310-4. doi:10.1109/ISRITI51436.2020.9315414.

[143] Ratchagit M, Xu H. Parameter identification of stochastic delay differential equations using differential evolution. In: The 37th International Technical Conference on Circuits/Systems, Computers, and Communications (ITC-CSCC); July 5-8, 2022; Phuket, Thailand. Accessed October 10, 2022. p. 804-6. doi: 10.1109/ITC-CSCC55581.2022.9894864.

[144] Eugene W, Moshe Z. On the relation between ordinary and stochastic differential equations. *International Journal of Engineering Science*. 1965;3(2):213-29. doi:10.1016/0020-7225(65)90045-5.

[145] Mikosch T. *Elementary stochastic calculus with finance in view*. 2nd ed. World scientific; 1998. Accessed March 15, 2021. https://books.google.com.au/books?hl=en&lr=&id=bHKBWbxTmwcC.

[146] Melnikov A, Wan H. On modifications of the Bachelier model. *Annals of Finance*. 2021;17:187-214. doi:10.1007/s10436-020003811.

[147] Bachelier L. Théorie de la spéculation. *Annales Scientifiques de lÉcole Normale Supérieure*. 1990;3:21-86.

[148] Uhlenbeck GE, Ornstein LS. On the relation between ordinary and stochastic differential equations. *Physical Review*. 1930;36(5):823-41. doi:10.1103/PhysRev.36.823.

[149] Boukhetala K, Guidoum A. Sim.DiffProc: A Package for Simulation of Diffusion Processes in R. Preprint Submitted to Journal of Statistical Software. 2011:hal:006298410. https://hal.archives-ouvertes.fr/hal-00629841/file/R-SimDiffProc.pdf.

[150] Dokuchaev N. A pathwise inference method for the parameters of diffusion terms. *Journal of Nonparametric Statistics*. 2017;29(4):731-43. doi:/10.1080/10485252.2017.1367789.

[151] Dokuchaev N. Modelling possibility of short-term forecasting of market parameters for portfolio selection. *Annals of Economics and Finance*. 2015;16(1):143-61.

[152] Reuters T. Thomson Reuters Tick History Intraday Summaries. Accessed June 11, 2020. https://hosted.datascope.reuters.com/DataScope; 2020.

[153] E S. VIX futures and options: A case study of portfolio diversification during the 2008 financial crisis. *The Journal of Alternative Investments*. 2009;12(2):68-85. doi:10.3905/JAI.2009.12.2.068.

[154] Ghimire S, Yaseen ZM, Farooque AA, Deo RC, Zhang J, Tao X. Streamflow prediction using an integrated methodology based on convolutional neural network and long short-term memory networks. *Scientific Reports*. 2021;11(1):1-26. doi: 10.1038/s41598-021967514.

[155] Storn R, Price K. Differential evolutiona simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*. 1997;11(4):341-59. doi: 10.1023/A:1008202821328.

[156] Nadimi-Shahraki MH, Taghian S, Mirjalili S, Faris H. MTDE: An effective multi-trial vector-based differential evolution algorithm and its applications for engineering design problems. *Applied Soft Computing*. 2020;97:106761. doi: 10.1016/j.asoc.2020.106761.

[157] Mohamed AW, Mohamed AK, Elfeky EZ, Saleh M. Enhanced directed differential evolution algorithm for solving constrained engineering optimization problems. *International Journal of Applied Metaheuristic Computing*. 2019;10(1):1-28. doi: 10.4018/IJAMC.2019010101.

[158] Constantine DM, Tymerski R, Greenwood G. Differential Evolution Optimization of the Broken Wing Butterfly Option Strategy. *Technology and Investment*. 2020;11(3):93-105. doi: 10.4236/ti.2020.113003.

[159] Jana RK, Ghosh I, Das D. A differential evolution-based regression framework for forecasting Bitcoin price. *Annals of Operations Research*. 2021;306(1):295-320. doi: 10.1007/s10479-021040008.

[160] Rout UK, Sahu RK, Panda S. Design and analysis of differential evolution algorithm based automatic generation control for interconnected power system. *Ain Shams Engineering Journal*. 2013;4(3):409 421. doi:10.1016/j.asej.2012.10.010.

[161] Li X, Xu L, Wang H, Song J, Yang SX. A differential evolution-based routing algorithm for environmental monitoring wireless sensor networks. *Sensors*. 2010;10(6):5425 5442. doi:10.3390/s100605425.

[162] Abou El Ela AA, Abido MA, Spea SR. Differential evolution algorithm for optimal reactive power dispatch. *Electric Power Systems Research*. 2011;81(2):458 464. doi:10.1016/j.epsr.2010.10.005.

[163] Ratchagit M, Xu H. A Two-Delay Combination Model for Stock Price Prediction. *Mathematics*. 2022;10(19):3447. doi:10.3390/math10193447.

[164] Bates JM, Granger CW. The combination of forecasts. *Journal of the operational research society*. 1969;20(4):451-68. doi: 10.1057/jors.1969.103.

[165] Petropoulos F, Svetunkov I. A simple combination of univariate models. *International journal of forecasting*. 2020;36(1):110-5. doi:10.1016/j.ijforecast.2019.01.006.

[166] Kawauchi K, Hirata K, Katoh C, Ichikawa S, Manabe O, Kobayashi K, et al. A convolutional neural network-based system to prevent patient misidentification in FDG-PET examinations. *Scientific reports*. 2019;9(1):1-9. doi:10.1038/s41598-01943656y.

[167] Saitoh K. *Deep Learning from the Basics: Python and Deep Learning: Theory and Implementation*. Packt Publishing Ltd; 2021. Accessed February 24, 2022. http://ebookcentral.proquest.com/lib/curtin/detail.action?docID=6524508.

[168] Haidar A, Verma B. Monthly rainfall forecasting using one-dimensional deep convolutional neural network. *IEEE Access*. 2018;6:69053-63. doi:10.1109/ACCESS.2018.2880044.

[169] Garbin C, Zhu X, Marques O. Dropout vs. batch normalization: an empirical study of their impact to deep learning. *Multimedia Tools and Applications*. 2020;79(19):12777-815. doi:10.1007/s11042-019084539.

[170] Chassagnon G, Vakalopolou M, Paragios N, Revel M. Deep learning: definition and perspectives for thoracic imaging. *European radiology*. 2020;30(4):2021-30. doi:10.1007/s00330-019065643.

[171] Schatteburg J, A L. Protocol for the diagnosis of keratoconus using convolutional neural networks. *Plos one*. 2022;17(2):e0264219. doi:10.1371/journal.pone.0264219.

[172] Kaur S, Aggarwal H, Rani R. Diagnosis of Parkinsons disease using deep CNN with transfer learning and data augmentation. *Multimedia Tools and Applications*. 2021;80(7):10113-39. doi:10.1007/s11042-020101141.

[173] Brownlee J. *Better deep learning: train faster, reduce overfitting, and make better predictions*. Machine Learning Mastery; 2018. Accessed February 24, 2022. https://books.google.com.au/books/about/$Better_Deep_Learning$.html.

[174] Borhani T, Saniedanesh M, Bagheri M, Lim J. QSPR prediction of the hydroxyl radical rate constant of water contaminants. *Water research*. 2016;1:344-53. doi:10.1016/j.watres.2016.04.038.

[175] Dami S, Esterabi M. Predicting stock returns of Tehran exchange using LSTM neural network and feature engineering technique. *Multimedia Tools and Applications*. 2021;80(13):19947-70. doi:10.1007/s11042-021107783.

[176] Ketkar N, Santana E. *Deep learning with Python*. Berkeley: Apress; 2017. Accessed February 24, 2022. doi: 10.1007/978-1-4842-2766-4.

[177] Du J, Liu Q, Chen K, Wang J. Forecasting stock prices in two ways based on LSTM neural network. In: IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC); March 15-17, 2019; Chengdu, China. Accessed February 15, 2022. p. 1083-6. doi:10.1109/ITNEC.2019.8729026.

[178] Chen Y, Lin W, Wang J. Which artificial intelligence algorithm better predicts the Chinese stock market? *IEEE Access*. 2018;6:48625-33. doi: 10.1109/ACCESS.2018.2859809.

[179] Wei B, Yuan D, Li H, Xu Z. Combination forecast model for concrete dam displacement considering residual correction. *Structural Health Monitoring*. 2019;18(1):232-44. doi:10.1177/1475921717748608.

[180] Zhemkov M. Nowcasting Russian GDP using forecast combination approach. *International Economics*. 2021;168:10-24. doi:10.1016/j.inteco.2021.07.006.

[181] Kourentzes N, Barrow D, Petropoulos F. Another look at forecast selection and combination: Evidence from forecast pooling. *International Journal of Production Economics*. 2019;209:226-35. doi:10.1016/j.ijpe.2018.05.019.

[182] Hertel L, Collado J, Sadowski P, Ott J, Baldi P. Sherpa: Robust Hyperparameter Optimization for Machine Learning. *SoftwareX*. 2000;12:100591. doi:10.1016/j.softx.2020.100591.

[183] Beucler T, Pritchard M, Rasp S, Ott J, Baldi P, Gentine P. Enforcing analytic constraints in neural networks emulating physical systems. *Physical Review Letters*. 2021;126(9):098302. doi:10.1103/PhysRevLett.126.098302.

[184] Ott J, Pritchard M, Best N, Linstead E, Curcic M, Baldi P. A Fortran-Keras deep learning bridge for scientific computing. *Scientific Programming*. 2020;2020:1-13. doi:10.1155/2020/8888811.

[185] Lu Y, Collado J, Whiteson D, Baldi P. Sparse autoregressive models for scalable generation of sparse images in particle physics. *Physical Review D*. 2021;103(3):036012. doi:10.1103/PhysRevD.103.036012.

[186] Markova M. Convolutional neural networks for forex time series forecasting. In: 2021 International Conference on Intelligent Technologies (CONIT); September 6-10, 2021; St. Constantin and Helena, Bulgaria. Accessed May 04, 2022. p. 030024. doi:10.1063/5.0083533.

[187] Sen J, Mehtab S, Nath G. stock price prediction using deep learning models. *Lattice-The Machine Learning Journal*. 2020;1(3):34-40.

[188] Sen J, Mehtab S. Accurate stock price forecasting using robust and optimized deep learning models. In: 2021 International Conference on Intelligent Technologies (CONIT); June 25-27, 2021; Hubli, India. Accessed May 04, 2022. p. 1-9. doi:10.1109/CONIT51480.2021.9498565.

Every reasonable effort has been made to acknowledge the owners of copyright material. I would be pleased to hear from any copyright owner who has been omitted or incorrectly acknowledged.