

การระบุตัวแบบอัตโนมัติสำหรับการวิเคราะห์อนุกรมเวลาโดยใช้การเรียนรู้ลึก



วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรดุษฎีบัณฑิต

สาขาวิชาคณิตศาสตร์ประยุกต์และวิทยาการคณนา

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

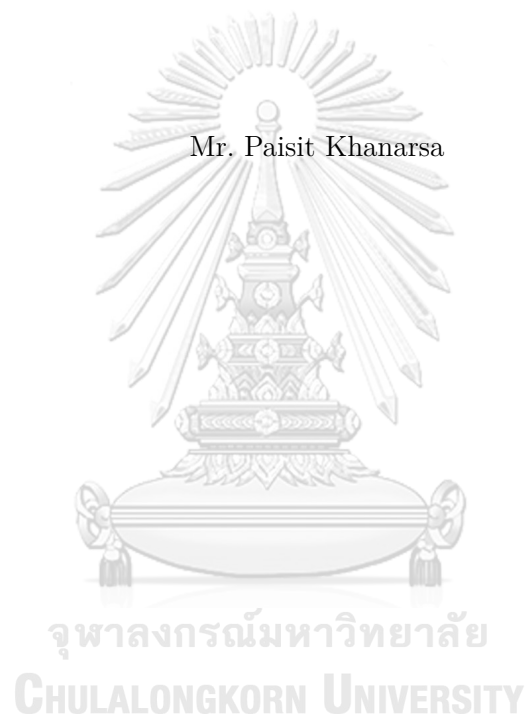
คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2562

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

AUTOMATIC MODEL IDENTIFICATION FOR TIME SERIES ANALYSIS
USING DEEP LEARNING

Mr. Paisit Khanarsa



A Dissertation Submitted in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy Program in Applied Mathematics and

Computational Science

Department of Mathematics and Computer Science

Faculty of Science

Chulalongkorn University

Academic Year 2019

Copyright of Chulalongkorn University

Dissertation Title AUTOMATIC MODEL IDENTIFICATION FOR TIME SE-
RIES ANALYSIS USING DEEP LEARNING
By Mr. Paisit Khanarsa
Field of Study Applied Mathematics and Computational Science
Dissertation Advisor Assistant Professor Krung Sinapiromsaran, Ph.D.

Accepted by the Faculty of Science, Chulalongkorn University in Partial Fulfillment
of the Requirements for the Doctoral Degree

..... Dean of the Faculty of Science
(Professor Polkit Sangvanich, Ph.D.)

DISSERTATION COMMITTEE

..... Chairman
(Assistant Professor Jiraphan Suntornchost, Ph.D.)

..... Dissertation Advisor
(Assistant Professor Krung Sinapiromsaran, Ph.D.)

..... Examiner
(Assistant Professor Kitiporn Plaimas, Ph.D.)

..... Examiner
(Associate Professor Petarpa Boonserm, Ph.D.)

..... External Examiner
(Assistant Professor Thidaporn Supapakorn, Ph.D.)

CHULALONGKORN UNIVERSITY

ไพสิฐุ ชั้นอาสา : การระบุตัวแบบอัตโนมัติสำหรับการวิเคราะห์อนุกรมเวลาโดยใช้การเรียนรู้ลึก. (AUTOMATIC MODEL IDENTIFICATION FOR TIME SERIES ANALYSIS USING DEEP LEARNING) อ.ที่ปรึกษาวิทยานิพนธ์หลัก : ผศ.ดร.กรง สินอภิรณสรณาย, 151 หน้า.

ข้อมูลอนุกรมเวลาส่วนใหญ่สามารถอธิบายผ่านกระบวนการเชิงเส้นด้วยตัวแบบออโตรีเกรซีฟอินทิเกรตเต้ดมุวี่งเอเวอเรนจ์ ซึ่งต้องการเวกเตอร์สามส่วนประกอบได้แก่ อันดับออโตรีเกรซีฟ อันดับดิฟเฟอเรนซ์ และ อันดับมุวี่งเอเวอเรนจ์ก่อนการหาสัมประสิทธิ์ที่เหมาะสม การระบุตัวแบบซึ่งตัดสินอันดับเหล่านั้นถูกวิเคราะห์ผ่านฟังก์ชันสหสัมพันธ์ในตัวเองบางส่วนเพื่อระบุอันดับออโตรีเกรซีฟ และฟังก์ชันสหสัมพันธ์ในตัวเองเพื่อระบุอันดับมุวี่งเอเวอเรนจ์ และฟังก์ชันสหสัมพันธ์ในตัวเองขยายเพื่อระบุอันดับทั้งสองซึ่งเป็นปัญหาที่ทำนายสำหรับนักสถิติ ดังนั้นวิธีอริมาอัตโนมัติถูกนำเสนอเพื่อแปรผันอันดับเหล่านั้นแบบอัตโนมัติและประมาณค่าสัมประสิทธิ์ที่สอดคล้อง วิทยานิพนธ์นี้นำเสนอสถาปัตยกรรมของโครงข่ายประสาทคอนโวลูชันสามแบบ ทั้งสามวิธีถูกขยายเพื่อไปสร้างตัวแบบออโตรีเกรซีฟอินทิเกรตเต้ดมุวี่งเอเวอเรนจ์ตามฤดูกาล และตัวแบบออโตรีเกรซีฟคอนดิชันนัล เฮเทอโรสเกดาสทิสติ จากการทดลองตัวแบบการเรียนรู้ลึกที่นำเสนอ ให้ผลดีกว่าวิธีอริมาอัตโนมัติในกรณีระบุอันดับอริมาและลำดับซาริมาผ่านพริซีชัน รีคอลและ คะแนนเอฟวัน

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

ภาควิชา	คณิตศาสตร์และ	ลายมือชื่อนิสิต
	วิทยาการคอมพิวเตอร์	ลายมือชื่อ อ.ที่ปรึกษาหลัก
สาขาวิชา	คณิตศาสตร์ประยุกต์	
	และวิทยาการคณนา	
ปีการศึกษา	2562	

6072816323 : MAJOR APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCE

KEYWORDS : TIME SERIES / DEEP LEARNING / ESACF / ARIMA / SARIMA / ARCH

PAISIT KHANARSA : AUTOMATIC MODEL IDENTIFICATION FOR TIME SERIES

ANALYSIS USING DEEP LEARNING. ADVISOR : ASST. PROF. KRUNG SINAPIROM-

SARAN, Ph.D., 151 pp.

Most time series data can be characterized by a linear process via the autoregressive integrated moving average model requiring a three-component vector which are the autoregressive, differencing, and moving average orders before fitting coefficients. A model identification which determines those orders is analyzed via the partial autocorrelation function to identify the autoregressive order, the autocorrelation function to identify the moving average order and an extended sample autocorrelation function to identify both orders which is a challenging problem for statisticians. Accordingly, the auto-ARIMA model was proposed to automatically vary those orders and estimates their corresponding coefficients. This thesis proposes three architectures of convolutional neural networks. They are widened to build the seasonal autoregressive integrated moving average model and the autoregressive conditional heteroskedasticity model. From the experiments, the proposed deep learning models outperform the auto-ARIMA model in the cases of identifying ARIMA order and the SARIMA order via precision, recall and f1-scores.

Department : .. Mathematics and Student's Signature

..... Computer Science Advisor's Signature

Field of Study : .. Applied Mathematics and ..

..... Computational Science

Academic Year : .. 2019

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere thanks to my thesis advisor, Assistant Professor Krung Sinapiromsaran for his invaluable help and constant encouragement of this research.

In addition, I am grateful to AMCS students for suggestions and all their help. Moreover, I most gratefully acknowledge Science Achievement Scholarship of Thailand (SAST) for funding this research and the Applied Mathematics and Computational Science, Department of Mathematics and Computer Science, Faculty of Science, Chulalongkorn University for supporting the equipment in this work.

Finally, I would like to thank my parents for their increasing encouragement, support and attention throughout my life.



จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

CONTENTS

	Page
ABSTRACT IN THAI	iv
ABSTRACT IN ENGLISH	v
ACKNOWLEDGEMENTS	vi
CONTENTS	vii
LIST OF TABLES	x
LIST OF FIGURES	xi
CHAPTER	
1 INTRODUCTION	1
1.1 Motivation	1
1.2 Objectives of the dissertation	6
1.3 Structure of the dissertation	6
2 BACKGROUND KNOWLEDGE OF TIME SERIES ANALYSIS	8
2.1 Time series analysis	8
2.1.1 Stochastic process and time series definitions	8
2.1.2 The autocovariance and autocorrelation function	11
2.1.3 The partial autocorrelation function	13
2.1.4 White noise process	14
2.2 Time series models for a linear process and a nonlinear process	15
2.2.1 Time series models for a linear process	15
2.2.2 Identifying the ARIMA order via ACF and PACF	18
2.2.3 Identifying the ARIMA order via ESACF	20
2.2.4 Time series models for the nonlinear process	22
2.3 The concept of the auto-ARIMA model	23
2.3.1 The maximum likelihood concept	24
2.3.2 Automatic identifying orders of the auto-ARIMA model by AIC	25
2.4 The model measurements for the forecasting time series	27
2.5 The statistical test of the forecasting time series	31
3 BACKGROUND KNOWLEDGE OF DEEP LEARNING	37

CHAPTER	Page
3.1	The concept of deep learning 37
3.2	The backpropagation concept 40
3.3	A review on convolutional neural networks 44
3.4	The convolutional neural network concept and the pooling concept 47
3.5	The model measurements for the classification problems 48
4	METHODOLOGY TO IDENTIFY THE ARIMA ORDER 52
4.1	The self-identification deep learning model 52
4.1.1	The architecture of the SID model 52
4.1.2	Experimental results of the SID model 63
4.1.3	The conclusion of the SID model 67
4.2	The self-identification ResNet-ARIMA model 68
4.2.1	The architecture the SIRO model 68
4.2.2	Experimental results of the SIRO model 72
4.2.3	The conclusion of the SIRO model 76
4.3	The ACF-PACF-ESACF convolutional neural network ARIMA order identification model 77
4.3.1	The architecture of the APEA model 77
4.3.2	Experimental results of the APEA model 82
4.3.3	The conclusion of the APEA model 86
4.4	Visualization of the convolutional neural network filter 87
5	METHODOLOGY TO IDENTIFY THE SARIMA ORDER AND BUILD THE LINEAR TIME SERIES MODEL 92
5.1	The ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model 92
5.1.1	Constructing the ACF-PACF-ESACF convolutional neural net- work seasonal ARIMA order identification model 92
5.1.2	Experimental results of the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model 95

CHAPTER	Page
5.1.3 The conclusion of the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model	99
5.2 The enhancing SARIMA forecasting model via the deep learning algorithm	100
5.2.1 Constructing the enhancing SARIMA forecasting model via the deep learning algorithm	100
5.2.2 Experimental results of the enhancing SARIMA forecasting model via the deep learning algorithm	102
5.2.3 The conclusion of the enhancing SARIMA forecasting model via the deep learning algorithm	122
6 METHODOLOGY FOR APPLYING TO THE NONLINEAR TIME SERIES MODEL	123
6.1 Constructing the automatic ARCH forecasting via deep learning algorithm	123
6.2 Experimental results of the automatic ARCH forecasting via deep learning algorithm by applying to the financial time series data	125
6.3 The conclusion of the automatic ARCH forecasting via deep learning algorithm	142
7 CONCLUSIONS AND DISCUSSIONS	143
REFERENCES	146
BIOGRAPHY	151

LIST OF TABLES

Table	Page
2.1 Time series data X_1, X_2, \dots, X_{10}	12
2.2 General ESACF table for $ARMA(p, q)$	21
2.3 General ESACF table for the $ARMA(1, 1)$	22
2.4 The table of AIC generated from the grid search	27
2.5 MAPE, SMAPE, RMSE and MAE of forecasting time series data	30
3.1 The confusion matrix	48
3.2 The confusion matrix of model 1	49
3.3 The confusion matrix of model 2	50
4.1 Description of each channel in the models for identifying the ARIMA order . .	64
4.2 Description of each channel in the models for identifying the ARIMA order . .	73
4.3 Description of each input of each model for identifying the ARIMA order . . .	83
5.1 Description of each channel in the models for identifying the SARIMA order .	96
5.2 MAPE , SMAPE and RMSE between the enhancing SARIMA forecasting model via the deep learning algorithm and the auto-ARIMA model of the synthetic time series data	109
5.3 The coefficients and the order from the synthetic time series number 0-5	110
5.4 The coefficients and the order from the synthetic time series number 6-10 . . .	111
5.5 Description I of the real world datasets	112
5.6 Description II of the real world datasets	113
5.7 MAPE , SMAPE and RMSE between the enhancing SARIMA forecasting model via the deep learning algorithm and the auto-ARIMA model of the real world datasets	114
5.8 The coefficients and the order from the real world time series number 0-5 . . .	120
5.9 The coefficients and the order from the real world time series number 6-10 . .	121

LIST OF FIGURES

Figure	Page
2.1 The example of the realization of the uniform time series data	9
2.2 ACF of the $AR(2)$	18
2.3 PACF of the $AR(2)$	19
2.4 ACF of the $ARIMA(1, 1, 1)$	19
2.5 PACF of the $ARIMA(1, 1, 1)$	20
2.6 The plot of the actual time series data and the three forecasting data	28
2.7 Residual 1 plot	32
2.8 Residual 2 plot	32
2.9 The ACF plot of residual 1	33
2.10 The ACF plot of residual 2	33
2.11 The p -values plot of residual 1	36
2.12 The p -values plot of residual 2	36
3.1 The structure of ANN	38
3.2 The structure of the node in the hidden layer	38
3.3 The sigmoid function	39
3.4 The relu function	40
3.5 A example of the neural network	41
3.6 The structure of the deep learning model	46
3.7 The convolution concept	47
3.8 The max-pooling concept	48
4.1 The architecture of the pq -SID model for identifying the AR and MA order	55
4.2 The architecture of the d -SID model for identifying the AR and MA order	56
4.3 The process of the simulated time series data of the ARMA order and the example of the time series data.	57
4.4 The example of time series having differencing $d = 0, 1, 2$	60
4.5 The processes of simulated time series data of the differencing order.	61
4.6 The example of generating the ACF image.	62

Figure	Page
4.7 The example of generating the series image.	63
4.8 The precisions, the recalls and the f1-scores of the models identifying p	65
4.9 The precisions, the recalls and the f1-scores of the models identifying q	66
4.10 The precisions, the recalls and the f1-scores of the models identifying d	67
4.11 The architecture of the pq -SIRO model for identifying the AR and MA order .	70
4.12 The architecture of the d -SIRO model for identifying the AR and MA order . .	71
4.13 The precisions, the recalls and the f1-scores of the models by identifying order p	74
4.14 The precisions, the recalls and the f1-scores of the models identifying q	75
4.15 The precisions, the recalls and the f1-scores of the models identifying d	76
4.16 The architecture of the pq -APEA model for identifying the AR and MA order	79
4.17 The architecture of the d -APEA model for identifying the differencing order .	80
4.18 Samples of ACF, PACF, ESACF and differencing time series images	82
4.19 The precisions, the recalls and the f1-scores of the models identifying p	84
4.20 The precisions, the recalls and the f1-scores of the models identifying q	85
4.21 The precisions, the recalls and the f1-scores of the models identifying d	86
4.22 The color scales of “Viridis”.	87
4.23 The filter matrices of the CNN layer which the AR order is 0	88
4.24 The filter matrices of the CNN layer which the AR order is 1	89
4.25 The filter matrices of the CNN layer which the AR order is 2	90
4.26 The filter matrices of the CNN layer which the AR order is 3	91
5.1 The ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model	94
5.2 The precision, the recall and the f1-score of the models identifying P	97
5.3 The precision, the recall and the f1-score of the models identifying Q	98
5.4 The precision, the recall and the f1-score of the models identifying D	99
5.5 The enhancing SARIMA forecasting model via the deep learning algorithm . .	101
5.6 Graphs of out-sample data and forecasting data and Ljung-Box test from synthetic time series 1-5.	104

5.7	Graphs of out-sample data and forecasting data and Ljung-Box test from synthetic time series 1-5.	105
5.8	Graphs of out-sample data and forecasting data and Ljung-Box test from synthetic time series 1-5.	106
5.9	Graphs of out-sample data and forecasting data and Ljung-Box test from synthetic time series 1-5.	107
5.10	Graphs of out-sample data and forecasting data and Ljung-Box test from synthetic time series 1-5.	108
5.11	Graphs of out-sample data and forecasting data and Ljung Box test from the real world time series 1 and 2.	115
5.12	Graphs of out-sample data and forecasting data and Ljung-Box test from the real world time series 3 and 4.	116
5.13	Graphs of out-sample data and forecasting data and Ljung-Box test from the real world time series 5 and 6.	117
5.14	Graphs of out-sample data and forecasting data and Ljung-Box test from the real world time series 7 and 8.	118
5.15	Graphs of out-sample data and forecasting data and Ljung-Box test from the real world time series 9 and 10.	119
6.1	The automatic ARCH forecasting via deep learning algorithm	124
6.2	The daily closing price of the S&P500 index	126
6.3	The daily closing price of the NASDAQ index	126
6.4	The daily closing price of the Dow Jones index	127
6.5	(a)Residuals and (b)Squared residuals of the S&P500 index	128
6.6	(a)Residuals and (b)Squared residuals of the NASDAQ index	129
6.7	(a)Residuals and (b)Squared residuals of the Dow Jones index	130
6.8	The Standardized residuals of the ARCH model fitting with the S&P500 index	132
6.9	The Standardized residuals of the ARCH model fitting with the NASDAQ index	133
6.10	The Standardized residuals of the ARCH model fitting with the Dow Jones index	134
6.11	The ACF plot in the case of S&P500	136
6.12	The ACF plot in the case of NASDAQ	137

6.13	The ACF plot in the case of Dow Jones	138
6.14	Ljung-Box test in the case of S&P500	139
6.15	Ljung-Box test in the case of NASDAQ	140
6.16	Ljung-Box test in the case of Dow Jones	141



CHAPTER I

INTRODUCTION

1.1 Motivation

Forecasting a time series data is a challenging problem for professional researchers [1-3]. It helps to analyze many problems in various fields such as planning investments, analyzing the risk, etc. Nevertheless, studying the characteristics of the data from the time series, like prediction, is difficult due to the complicated relationships of the data which are sensitive to their own internal dynamics. [4, 5] Moreover, there are some time series according to nonlinear and non-stationary which are difficult to analyze. In other words, the time series has large swings and non-constant variances [6]. Therefore, it is difficult to predict the fluctuation of future values accurately and reliably. Consequently, the time series is needed to be analyzed via the concept of the time series analysis to understand the structure of the time series for forecasting the future values.

Generally, the time series analysis can be divided into two approaches which are the non-parametric approach and the parametric approach. The non-parametric approach includes the spectral analysis and estimating the covariance of the process without assuming any particular structure. The spectral analysis [7] is based on building the spectral density in the form of a cosine function. The spectral density is calculated from the weighted sum of cyclical components corresponding to a frequency in the interval $[-\pi, \pi]$. Then, the sample spectral density is estimated by sample autocovariance from the time series data, called the periodogram [8]. In 2011, Tchrakian et al. [9] applied the spectral analysis to forecast the real-time traffic flow. In the same year, Afshar and Bigdeli [10] used the singular spectral analysis to forecast the short term of the electricity market in Iran. In 2017, Du et al. [11] applied the spectral analysis and the discrete wavelet transform to build the artificial neural network model to forecast the Indian monthly rainfall.

However, this process requires complex mathematical background such as Fourier analysis [12]. Alternatively, the parametric approach is introduced. It assumes that the time series data comes from the stationary stochastic process which it can be described using some parameters. Normally, the stochastic time series analysis based on a linear process begins with the basic models such as the autoregressive integrated moving average model (ARIMA) and the seasonal autoregressive integrated moving average model (SARIMA) by Box and Jenkins in 1970 [13] and drives to more complex models such as the autoregressive conditional heteroskedasticity model (ARCH) by Engle in 1983 [14] and the generalized autoregressive conditional heteroskedasticity model (GARCH) by Bollerslev in 1990 [15].

The process of building the ARIMA model for forecasting time series composes of (1) the model identification and (2) the Box-Jenkins estimation to obtain the best model. The key components of these models are the autoregressive or AR component, the differencing component and the moving average or MA component. Statisticians who study the time series analysis define the MA order from the sample autocorrelation function (ACF) and the AR order from the sample partial autocorrelation function (PACF) via visualizing their plots [16]. Nonetheless, they may obscure one another. Therefore, the extended sample autocorrelation function or ESACF is introduced by Tsay and Tiao [17] and Chenoweth et al. in 2000 [18] for identifying the AR order and the MA order using a single plot.

The ACF plot, the PACF plot and the ESACF plot may be difficult for a statistician to extract the appropriate ARIMA order visually. Recently, to avert human analysis, analysts use the auto-ARIMA model to automatically determine the ARIMA order and the best coefficients according to the Akaike Information Criterion or AIC [19], the Bayesian Information Criterion or BIC [20] and the Hannan Quinn Information Criterion or HQIC [21]. The main advantage is that it requires no human interaction. However, its residuals may deviate from a white noise process which causes the forecast values to deviate so much from the actual ones. At present, these methods are still used in various fields, such as stochastic wind power generation by Chen et al. [22] in 2009, water quality

series by Faruk [23] in 2010 and daily and monthly average global solar radiation in Seoul, South Korea by Alsharif et al. in 2019 [24].

There are many researchers proposing methods to identify the appropriate ARIMA order because it is an important part of fitting the ARIMA model. Lee and Oh in 1996 [25] used the ANN-driven decision tree classifier to identify the ARMA order and suggested that most time series data uses the ARMA order less than 5. Chenoweth et al. in 2000 [18] used the extended sample autocorrelation function or ESACF to identify the orders of AR and MA at the same time. Then, Al-Qawasmi et al. in 2010 [26] applied the identification of the ARIMA order with the concept of a special covariance matrix and used the MEV criterion for selecting the best ARIMA model.

At present, deep learning is used to apply to several problems successfully including the time series analysis. In 2016, Amini and Karabasoglu [27] used the improved ARIMA model by optimizing the differencing order and the autoregressive order to predict the electric vehicle charging demand for stochastic power system operation. In 2017, Guarnaccia et al. [28] used the seasonal ARIMA time series models and the deterministic decomposition to predict the airport noise at Nice international airport in France. In 2018, Fukuoka et al. [29] used LSTM and CNN having one dimension of the filter matrix for the input time series data to predict the wind speed. In 2019, Kim et al. [30] proposed the LSTM-CNN model merging features of the time series data consisting of stock time series and stock chart images to forecast the stock prices of SPDR, S&P500, and ETF. In addition, the deep learning concept is applied to recognize the ARMA order for building the ARMA model by Tang and Röllin [31] in 2018. The method is constructed based on ResNet50 using the time series data directly as its input. Their results showed that, for the low-order ARMA model, ResNet50 outperformed the auto-ARIMA model in terms of speed and accuracy.

The main purpose of identifying the ARIMA order is to select the order and estimate the parameters to forecast the time series data accurately. The model identification via the Box-Jenkins method is a conventional method for identifying their orders but it requires

an analyst to visualize multiple plots. Although the problem can be automated using the auto-ARIMA model to identify the order, the results of the order may still be poor due to the leftover autocorrelation in residuals.

Even though the structure describing the ARIMA model is not difficult and simple to interpret, it may not be able to fit the time series having the nonlinear structure like the financial time series data. Consequently, the nonlinear time series data may not be suitable to be applied by a linear process. In the nonlinear time series analysis, many aspects are considered such as instability of variance or nonlinearity relationship. Many approaches which occur in this field attempt to capture the nonlinear characteristics of the time series data such as ARCH and GARCH family. The interesting conventional approach is the autoregressive conditional heteroskedasticity model or the ARCH model using for capturing the non-constant variance via the autoregressive process and it is commonly applied in modeling the financial time series data.

Consequently, to improve the limit of the order identification from the auto-ARIMA model, this research attempts to combine the concept of the time series analysis and deep learning together to identify the ARIMA order via training the ACF plot, the PACF plot, the ESACF plot and the time series images. this research starts by proposing the three methodologies to improve the model identification of the ARIMA order and the SARIMA order using the deep learning model and adapt this concept with the ARCH model for applying to the financial time series data having the non constant variance. Our research starts on the model identification of the AR and MA orders ignoring trends. So the training time series data will be simulated from the AR model and the MA model. This training time series data is used to train the deep learning model to identify the ARMA order. Then the training time series data simulated from the AR model and the MA model is accumulated for being the time series data having the effect of trends. This time series data is used to train the deep learning model to identify the differencing order. Hence, the first methodology begins with building the deep learning model, called the self-identification deep learning model or the SID model using the basic of the convolutional neural networks. It is split to pq -SID model to identify the ARMA order in the range of

0-5 and d -SID model to identify the differencing order in the range of 0-2. For beginning, the range of identifying the ARMA order is set according to the research of Lee and Oh [18] which they claimed that the most time series uses the ARMA order less than 5. The SID model is constructed by training the characteristic of ACF, PACF and series images. Then the SID model is improved using the ResNet architecture which it can improve the convergence of the weights of the deep learning model. The extended model is called the self-identification ResNet-ARIMA model or the SIRO model also splitting to the pq -SIRO model and the d -SIRO model. The SIRO model is constructed by training ACF, PACF and time series images taking differencing from $d = 0$ to 2. Moreover, to use ESACF as inputs, the ACF-PACF-ESACF convolutional neural network ARIMA order identification or the APEA model is introduced. it is split into the pq -APEA model and the d -APEA model. The pq -APEA model is constructed by training the characteristic of ACF, PACF, ESACF and differencing time series images to identify the ARMA order in the range of 0-7 while the d -APEA model is constructed by training ACF or PACF taking differencing from 0 to 3 to identify the differencing order in the range of 0-3.

Next, for the second methodology, the APEA model is extended to identify the SARIMA order, called the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model by using the concept of the spectral analysis to find the length of season and using the concept of aggregating the time series data to identify the seasonal order. Then the SARIMA order from the model is used to fit coefficients of the time series model via the Box-Jenkins method, called the enhancing SARIMA forecasting via the deep learning algorithm.

The last methodology is to adapt the first methodology and the second methodology to build the ARCH model for applying with the financial time series data, called the automatic ARCH forecasting via the deep learning algorithm. The algorithm uses the deep learning to identify the SARIMA order and build the SARIMA model. Then the residuals of forecasting the time series data are applied to the deep learning model for identifying the ARCH order to build the ARCH model.

1.2 Objectives of the dissertation

This dissertation aims to (1) present the deep learning model for identifying the orders of the ARIMA model, (2) present the algorithm for identifying the SARIMA order by applying the deep learning model and for building the SARIMA model to forecast the future values and (3) present the methodology for improving identifying the ARCH order and for building the ARCH model to forecast the financial time series data.

1.3 Structure of the dissertation

Generally, to forecast the time series data accurately, the time series data must be analyzed via the concept of the time series analysis. Moreover, identifying the ARIMA order and the SARIMA order using common criteria, such as AIC, may not find the appropriate orders. Consequently, this research uses the deep learning model to identify the ARIMA order and the SARIMA order and build the forecasting model to forecast the time series data.

- This research presents the deep learning model applying with ACF, PACF, ESACF and differencing time series as inputs to identify the ARIMA order.
- This research presents the algorithm to identify the SARIMA order by applying to the deep learning model and build the SARIMA model to forecast future values.
- This research uses the deep learning model to apply with the ARCH model for forecasting the financial time series data.

The remainders of this dissertation are organized as follows. Chapter 2 and Chapter 3 describe the basic knowledge of time series analysis and deep learning. Chapter 4 explains the details of the methodology to identify the ARIMA order including the SID model, the SIRA model and the the APEA model. Chapter 5 explains the details of the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model to identify the SARIMA order and the enhancing SARIMA forecasting via deep

learning algorithm to build the SARIMA model and forecast the future values. Chapter 6 explains the details of the automatic ARCH forecasting via deep learning algorithm to apply to the financial time series. Chapter 7 is the conclusion and discussion.



CHAPTER II

BACKGROUND KNOWLEDGE OF TIME SERIES ANALYSIS

This chapter states the necessary background knowledge of the time series used in this dissertation. Firstly, the background knowledge of the time series analysis is illustrated in Section 2.1. Secondly, the time series models for the linear process and the nonlinear process are explained in Section 2.2. Thirdly, the concept of the auto-ARIMA model is described in Section 2.3. Lastly, the model measurements of forecasting time series and the statistical test are introduced in Section 2.4 and Section 2.5.

2.1 Time series analysis

In this section, the fundamental concepts of the time series that are necessary for this dissertation are introduced consisting the stochastic process, the autocorrelation function and the partial autocorrelation function, respectively.

2.1.1 Stochastic process and time series definitions

The basic of stochastic process and its definition are stated below.

Definition 2.1. Let $\{X_t\}_{t \in T}$ be a stochastic process, where T be an index set and X_t is a random variable indexed by $t \in T$.

Definition 2.2. A discrete-time stochastic process is a family of time indexed random variable $\{X_t\}_{t \in T}$, where T is discrete time index set.

Next, the time series in this dissertation is defined by the stochastic process.

Definition 2.3. A time series is a realization or a sample function from a certain stochastic process.

For example, let $\{X_t\}_{t \in \{0,1,2,\dots,100\}}$ be a random variable where $X_t \sim U(-1, 1)$. Hence, the realization of $\{X_t\}_{t \in \{0,1,2,\dots,100\}}$ can be plotted as Figure 2.1.

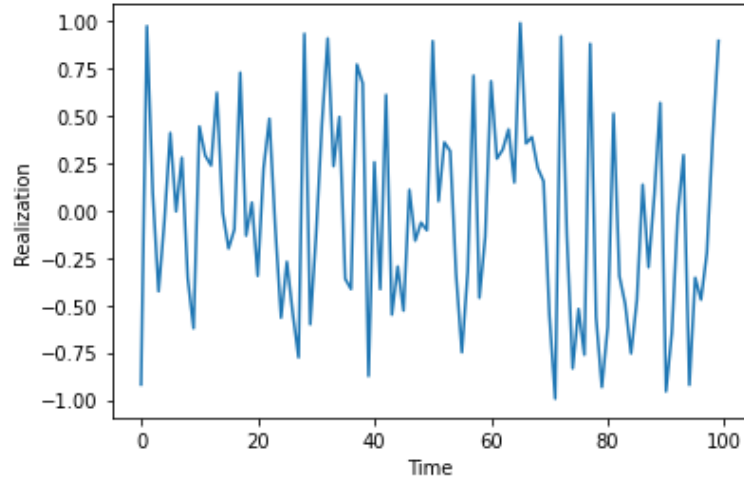


Figure 2.1: The example of the realization of the uniform time series data

Definition 2.4. Consider the index set as the set of all integers. A finite set of random variables from a stochastic process $\{X_t\}_{t \in \{\dots, -2, -1, 0, 1, 2, \dots\}}$ is represented by $\{X_{t_1}, X_{t_2}, \dots, X_{t_n}\}$. The n -dimensional distribution function is defined by

$$F_{X_{t_1}, X_{t_2}, \dots, X_{t_n}}(x_1, \dots, x_n) = P\{X_{t_1} \leq x_1, \dots, X_{t_n} \leq x_n\},$$

where each $x_i, i = 1, \dots, n$, is a real number.

Definition 2.5. A process is called first-order stationary in distribution if $F_{X_{t_1}}(x_1) = F_{X_{t_1+k}}(x_1)$ for any integers t_1 and k .

Moreover, a process is called second-order stationary in distribution if $F_{X_{t_1}, X_{t_2}}(x_1, x_2) = F_{X_{t_1+k}, X_{t_2+k}}(x_1, x_2)$ for any integers t_1, t_2 , and k .

For n th-order stationary in distribution,

$$F_{X_{t_1}, X_{t_2}, \dots, X_{t_n}}(x_1, x_2, \dots, x_n) = F_{X_{t_1+k}, X_{t_2+k}, \dots, X_{t_n+k}}(x_1, x_2, \dots, x_n)$$

for any integers t_1, t_2, \dots, t_n and k .

Definition 2.6. A process is called strictly stationary if

$$F_{X_{t_1}, X_{t_2}, \dots, X_{t_n}}(x_1, x_2, \dots, x_n) = F_{X_{t_1+k}, X_{t_2+k}, \dots, X_{t_n+k}}(x_1, x_2, \dots, x_n)$$

is true for any $n = 1, 2, \dots$.

Definition 2.7. For a given real-valued process $\{X_t\}_{t \in \{\dots, -2, -1, 0, 1, 2, \dots\}}$, the mean function of the process is

$$\mu_t = E(X_t),$$

the variance function of the process is

$$\sigma_t^2 = E(X_t - \mu_t)^2,$$

the covariance function between X_{t_1} and X_{t_2} is

$$\gamma(t_1, t_2) = E(X_{t_1} - \mu_{t_1})(X_{t_2} - \mu_{t_2}),$$

and the correlation function between X_{t_1} and X_{t_2} is

$$\rho(t_1, t_2) = \frac{\gamma(t_1, t_2)}{\sqrt{\sigma_{t_1}^2} \sqrt{\sigma_{t_2}^2}}.$$

Since the distribution function of the strictly stationary process is the same for all t , the mean function $\mu_t = \mu$ is a constant and the variance function $\sigma_t^2 = \sigma^2$ is a constant.

The covariance function is

$$\gamma(t_1, t_2) = \gamma(t - k, t) = \gamma(t, t + k) = \gamma_k,$$

and the correlation function is

$$\rho(t_1, t_2) = \rho(t - k, t) = \rho(t, t + k) = \rho_k,$$

where and $t_1 = t - k$ and $t_2 = t$.

Definition 2.8. For a given real-valued process $\{X_t\}_{t \in \{\dots, -2, -1, 0, 1, 2, \dots\}}$, the process is weakly stationary or covariance-stationary if

1. The first moment of X_t is independent of t or $E(X_t) = \mu$, where μ is a constant for all t .
2. The second moment of X_t is finite for all t or $E(X_t^2) < \infty$ for all t .
3. The cross moment or covariance of X_{t_1} and X_{t_2} depends only on $t_1 - t_2 = k$ or $\gamma(t_1, t_2) = \gamma_k$.

In this dissertation, any stationary process is referred to the weakly stationary process since most time series data will not satisfy the strong stationary one.

2.1.2 The autocovariance and autocorrelation function

Definition 2.9. For a weakly stationary process $\{X_t\}$, the mean function is $E(X_t) = \mu$ and the variance function $Var(X_t) = \sigma^2$, where μ and σ^2 are constant. The covariance between X_t and X_{t+k} is

$$\gamma_k = Cov(X_t, X_{t+k}) = E(X_t - \mu)(X_{t+k} - \mu)$$

and the correlation between X_t and X_{t+k} is

$$\rho_k = \frac{Cov(X_t, X_{t+k})}{\sqrt{Var(X_t)}\sqrt{Var(X_{t+k})}} = \frac{\gamma_k}{\gamma_0}$$

γ_k is called the autocovariance function and ρ_k is called the autocorrelation function or ACF.

Theorem 2.1. For a given observed time series $\{X_1, X_2, \dots, X_n\}$, the sample ACF is

defined as

$$\hat{\rho}_k = \frac{\gamma_k}{\gamma_0} = \frac{\sum_{t=1}^{n-k} (X_t - \bar{X})(X_{t+k} - \bar{X})}{\sum_{t=1}^n (X_t - \bar{X})^2},$$

$k = 0, 1, \dots, n - 1$ where $\bar{X} = \frac{\sum_{t=1}^n X_t}{n}$ is the sample mean of the series.

Example 1. : Let $\{X_1, X_2, \dots, X_{10}\}$ be the time series data. It is shown in Table 2.1.

t	1	2	3	4	5	6	7	8	9	10
X_t	13	8	12	11	5	4	14	9	15	13

Table 2.1: Time series data X_1, X_2, \dots, X_{10}

The sample mean of the time series data is

$$\bar{X} = \frac{\sum_{t=1}^n X_t}{10} = \frac{13 + 8 + 12 + 11 + 5 + 4 + 14 + 9 + 15 + 13}{10} = 10.4$$

Then, ACF of the time series data can be computed as

$$\begin{aligned} \hat{\rho}_1 &= \frac{\sum_{t=1}^9 (X_t - \bar{X})(X_{t+1} - \bar{X})}{\sum_{t=1}^{10} (X_t - \bar{X})^2} \\ &= \frac{(13 - 10.4)(8 - 10.4) + (8 - 10.4)(12 - 10.4) + \dots + (15 - 10.4)(13 - 10.4)}{(13 - 10.4)^2 + (8 - 10.4)^2 + \dots + (13 - 10.4)^2} \\ &= -0.003 \end{aligned}$$

$$\begin{aligned}
\hat{\rho}_2 &= \frac{\sum_{t=1}^8 (X_t - \bar{X})(X_{t+2} - \bar{X})}{\sum_{t=1}^{10} (X_t - \bar{X})^2} \\
&= \frac{(13 - 10.4)(12 - 10.4) + (8 - 10.4)(11 - 10.4) + \cdots + (9 - 10.4)(13 - 10.4)}{(13 - 10.4)^2 + (8 - 10.4)^2 + \cdots + (13 - 10.4)^2} \\
&= -0.057
\end{aligned}$$

$$\begin{aligned}
\hat{\rho}_3 &= \frac{\sum_{t=1}^7 (X_t - \bar{X})(X_{t+3} - \bar{X})}{\sum_{t=1}^{10} (X_t - \bar{X})^2} \\
&= \frac{(13 - 10.4)(11 - 10.4) + (8 - 10.4)(5 - 10.4) + \cdots + (14 - 10.4)(13 - 10.4)}{(13 - 10.4)^2 + (8 - 10.4)^2 + \cdots + (13 - 10.4)^2} \\
&= -0.047
\end{aligned}$$

Other $\hat{\rho}_k$ can be calculated correspondingly.

2.1.3 The partial autocorrelation function

Definition 2.10. For a weakly stationary process $\{X_t\}$, let the mean function be $E(X_t) = \mu$ and the variance function $Var(X_t) = \sigma^2$, where μ and σ^2 are constant. The partial autocorrelation function ϕ_k between X_t and X_{t+k} is

$$\phi_k = Corr(X_t, X_{t+k} | X_{t+1}, \dots, X_{t+k-1}).$$

Theorem 2.2. For a given observed time series $\{X_1, X_2, \dots, X_n\}$, sample PACF for lag k is defined as

$$\phi_{k+1} \approx \hat{\phi}_{k+1, k+1} = \frac{\hat{\rho}_{k+1} - \sum_{j=1}^k \hat{\phi}_{kj} \hat{\rho}_{k+1-j}}{1 - \sum_{j=1}^k \hat{\phi}_{kj} \hat{\rho}_j}$$

and

$$\hat{\phi}_{k+1,j} = \hat{\phi}_{kj} - \hat{\phi}_{k+1,k+1}\hat{\phi}_{k,k+1-j},$$

for $j = 1, 2, \dots, k$.

From example 1 in Section 2.1.2, PACF of the time series data can be computed as

$$\hat{\phi}_{11} = \hat{\rho}_1 = -0.003$$

$$\hat{\phi}_{22} = \frac{\hat{\rho}_2 - \hat{\rho}_1^2}{1 - \hat{\rho}_1^2} = \frac{-0.057 - (-0.003)^2}{1 - (-0.003)^2} = -0.057$$

$$\hat{\phi}_{21} = \hat{\phi}_{11} - \hat{\phi}_{22}\hat{\phi}_{11} = -0.003 - (-0.057)(-0.003) = -0.003$$

$$\begin{aligned} \hat{\phi}_{33} &= \frac{\hat{\rho}_3 - \hat{\phi}_{21}\hat{\rho}_2 - \hat{\phi}_{22}\hat{\rho}_1}{1 - \hat{\phi}_{21}\hat{\rho}_2 - \hat{\phi}_{22}\hat{\rho}_1} = \frac{-0.047 - (-0.003)(-0.057) - (-0.057)(-0.003)}{1 - (-0.003)(-0.057) - (-0.057)(-0.003)} \\ &= -0.047 \end{aligned}$$

Other $\hat{\phi}_{kk}$ can be calculated correspondingly.

2.1.4 White noise process

A process $\{\epsilon_t\}$ is called a white noise process if the process is a sequence of uncorrelated random variables from a fixed distribution. Normally, the white noise process $\{\epsilon_t\}$ has the constant mean $E(\epsilon_t) = 0$ and the constant variance $Var(\epsilon_t) = \sigma_\epsilon^2$ and $\gamma_k = Cov(\epsilon_t, \epsilon_{t+k})$ for all $k \neq 0$.

By definition, a white noise process $\{\epsilon_t\}$ is stationary with the autocovariance function

$$\gamma_k = \begin{cases} \sigma_\epsilon^2 & \text{if } k = 0 \\ 0 & \text{if } k \neq 0 \end{cases},$$

the autocorrelation function

$$\rho_k = \begin{cases} 1 & \text{if } k = 0 \\ 0 & \text{if } k \neq 0 \end{cases},$$

and the partial autocorrelation function

$$\phi_{kk} = \begin{cases} 1 & \text{if } k = 0 \\ 0 & \text{if } k \neq 0 \end{cases}.$$

A white noise process is Gaussian if its joint distribution is normal.

2.2 Time series models for a linear process and a nonlinear process

In this section, the basic of the time series models for a linear process and a nonlinear process are demonstrated. The linear process includes the autoregressive model, the moving average model, the autoregressive moving average model, the the autoregressive integrated moving average model and the seasonal autoregressive integrated moving average model. The nonlinear process includes the autoregressive conditional heteroskedastic model and the generalized autoregressive conditional heteroskedastic model. The details are described as follows.

2.2.1 Time series models for a linear process

The concept of the time series models based on a linear process is an important part for understanding the time series data.

The autoregressive process or the AR process of the time series analysis of p order can be explained by

$$x_t = \sum_{i=1}^p \phi_i x_{t-i} + \epsilon_t. \quad (2.1)$$

Next, the backward shift operator is defined as $x_{t-1} = Bx_t$. For example, $x_{t-2} = Bx_{t-1} = B(Bx_t) = B^2x_t$. Then Equation 2.1 can be rewritten as

$$x_t - \sum_{i=1}^p \phi_i x_{t-i} = \epsilon_t \quad (2.2)$$

$$x_t - \phi_1 x_{t-1} - \phi_2 x_{t-2} - \dots - \phi_p x_{t-p} = \epsilon_t \quad (2.3)$$

$$x_t - \phi_1 B x_t - \phi_2 B^2 x_t - \dots - \phi_p B^p x_t = \epsilon_t \quad (2.4)$$

$$(1 - \phi_1 B - \phi_2 B^2 - \dots - \phi_p B^p) x_t = \epsilon_t \quad (2.5)$$

$$\phi_p(B) x_t = \epsilon_t, \quad (2.6)$$

where x_t and ϵ_t are the time series data and random error at time t . $\phi_1, \phi_2, \dots, \phi_p$ are the coefficients of the AR process and $\phi_p(B) = 1 - \phi_1(B) - \phi_2(B^2) - \dots - \phi_p(B^p)$. To explain the stationary of the AR process, the root of $\phi_p(B) = 0$ where B is treated as the variable of this polynomial must be outside of the unit circle [16] to prevent the divergence of the coefficients in the model.

The moving average process or the MA process of q order can be explained by

$$x_t = \epsilon_t - \sum_{i=1}^q \theta_i \epsilon_{t-i} \quad (2.7)$$

or

$$x_t = \theta_q(B) \epsilon_t, \quad (2.8)$$

where $\theta_1, \theta_2, \dots, \theta_q$ are the coefficients of the MA process and $\theta_q(B) = 1 - \theta_1(B) - \theta_2(B^2) - \dots - \theta_q(B^q)$. In the concept of MA process, the stationary is explained in the name of invertibility according that the root of $\theta_q(B) = 0$ where B is treated as the variable of this polynomial must be outside of the unit circle [16].

Then, the extension of the AR process and the MA process is built as the autoregressive moving average or the ARMA process, expressed as follows

$$\phi_p(B)x_t = \theta_q(B)\epsilon_t \quad (2.9)$$

To confirm that the ARMA model has the property of stationary and invertibility, the roots of $\phi_p(B) = 0$ and $\theta_q(B) = 0$ have to be outside of the unit circle [16], respectively.

To deal with the time series trend, the differencing part is also applied to the ARMA process creating the autoregressive integrated moving average or ARIMA process, expressed as follows

$$\phi_p(B)\nabla^d x_t = \theta_q(B)\epsilon_t \quad (2.10)$$

where d is the number of differencing and $\nabla x_t = x_t - x_{t-1}$.

Generally, the seasonal effect is applied to the ARIMA process defining the seasonal period to be s . The process is called the seasonal autoregressive integrated moving average or the SARIMA process of order $(p, d, q) \times (P, D, Q)$.

$$\Phi_P(B^s)\phi_p(B)(1-B)^d(1-B^s)^D x_t = \Theta_Q(B^s)\theta_q(B)\epsilon_t \quad (2.11)$$

where

$$\phi_p(B) = 1 - \phi_1(B) - \phi_2(B^2) - \dots - \phi_p(B^p)$$

$$\theta_q(B) = 1 - \theta_1(B) - \theta_2(B^2) - \dots - \theta_q(B^q)$$

$$\Phi_P(B^s) = 1 - \Phi_1(B^s) - \Phi_2(B^{2s}) - \dots - \Phi_P(B^{Ps})$$

$$\Theta_Q(B^s) = 1 - \Theta_1(B^s) - \Theta_2(B^{2s}) - \dots - \Theta_Q(B^{Qs}).$$

P is the order of seasonal autoregressive term, Q is the order of moving average term and D is the number of seasonal differencing which $\Phi_1, \Phi_2, \dots, \Phi_P$ represent the

coefficients of seasonal AR and $\Theta_1, \Theta_2, \dots, \Theta_Q$ represent the coefficients of seasonal MA.

2.2.2 Identifying the ARIMA order via ACF and PACF

To identify the ARIMA order, the concepts of ACF and PACF are presented which can be calculated by formulae from the previous section. The AR order is identified by the cut off at the first PACF lag. Similarly for the MA order is recognized by the cut off at the first ACF lag. The differencing order will be determined by considering the lowest positive integer of the ACF lag that its plot decays rapidly to zero, either from above or below. For example, the sample ACF and PACF of the $AR(2)$ are shown in Figure 2.2 and Figure 2.3 and the sample ACF and PACF of the $ARIMA(1, 1, 1)$ are shown in Figure 2.4 and Figure 2.5.

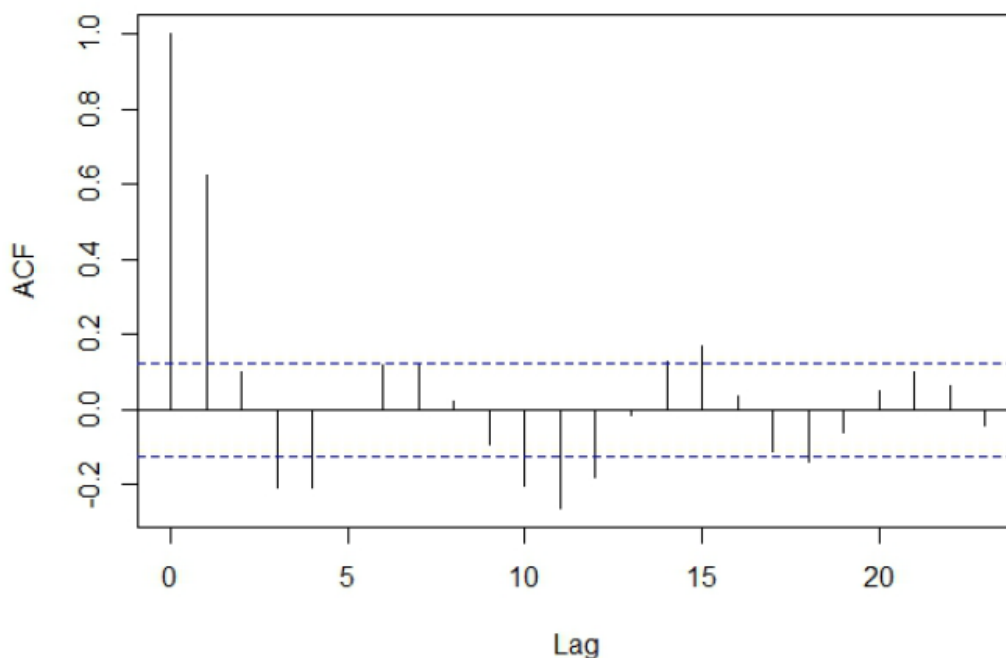


Figure 2.2: ACF of the $AR(2)$

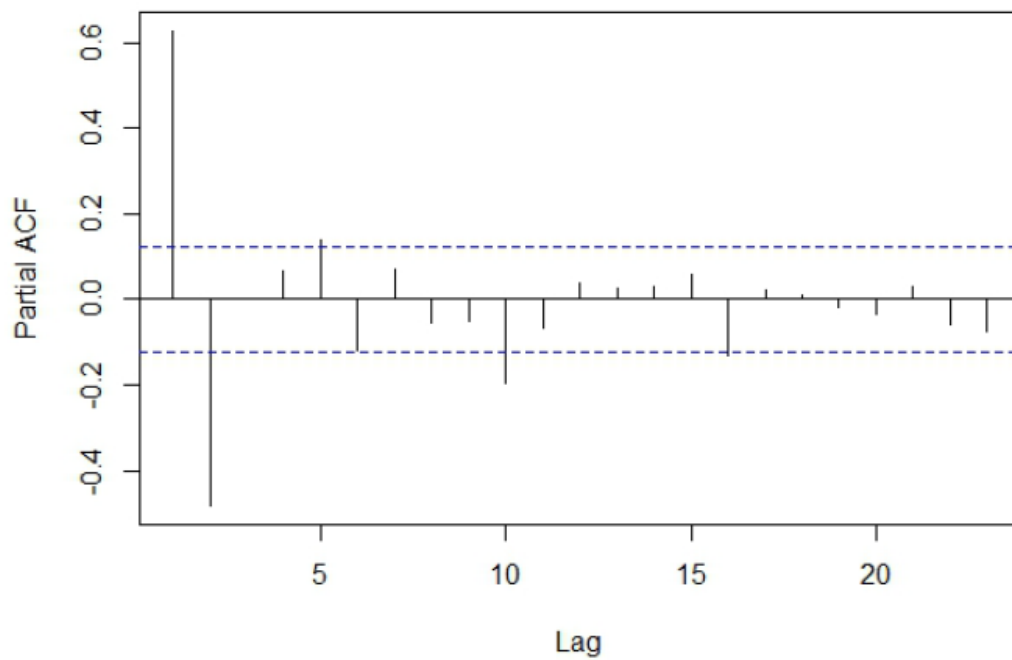


Figure 2.3: PACF of the $AR(2)$

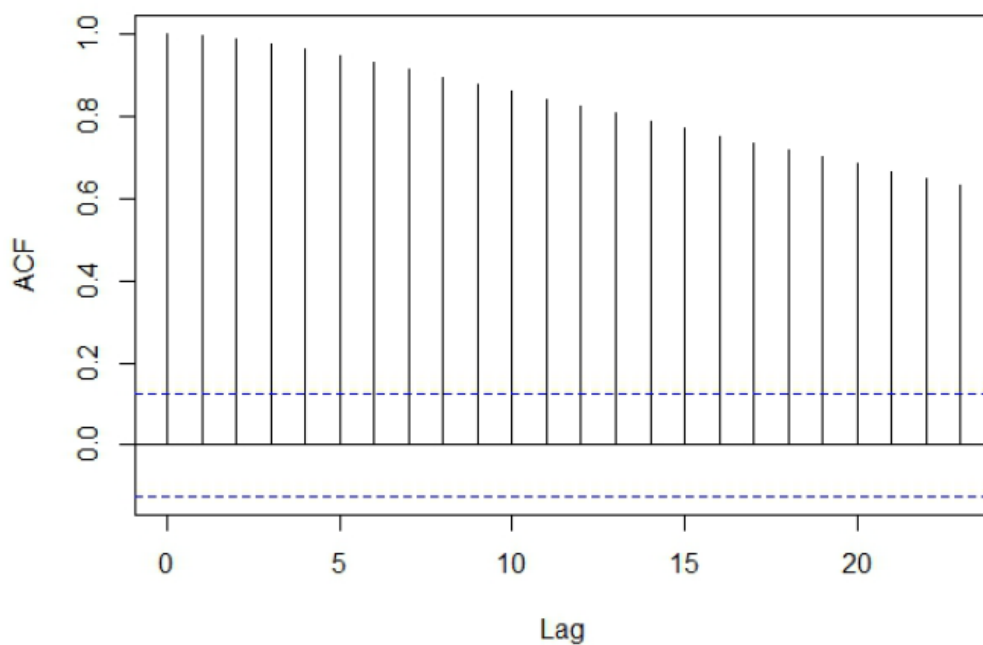


Figure 2.4: ACF of the $ARIMA(1,1,1)$

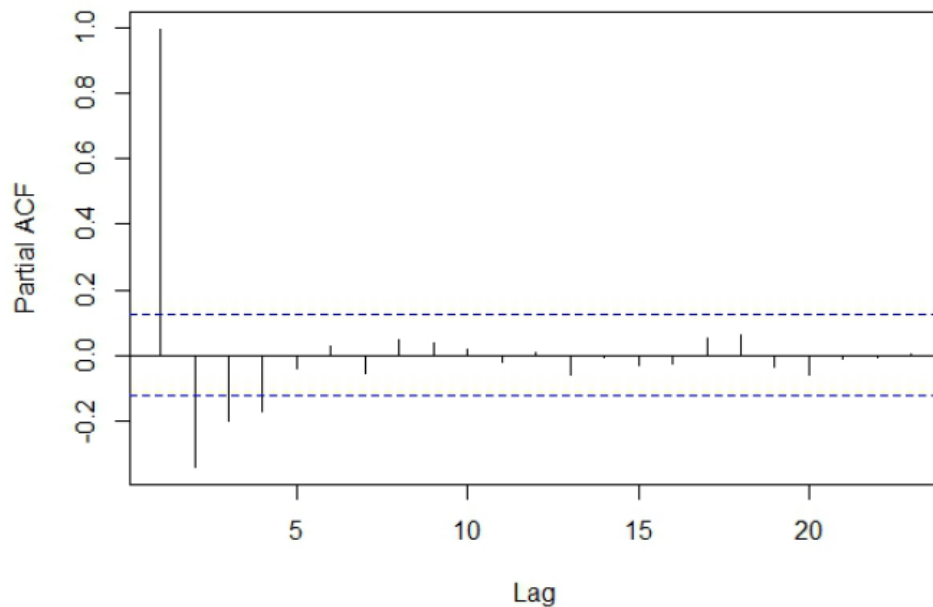


Figure 2.5: PACF of the $ARIMA(1, 1, 1)$

2.2.3 Identifying the ARIMA order via ESACF

Moreover, there is a tool which is used to identify the AR order and the MA order at the same time. The extended sample autocorrelation function or ESACF which was proposed by Tsay and Tiao in 1984 [17]. Generating ESACF is the process based on iterated least squares estimates of the autoregressive order or the AR order. The ESACF value is represented by $r_j^{(m)}$ computing via sample ACF of residual which occurs from the time series fitted by $AR(m)$ where m^{th} -row, $m = 0, 1, 2, 3, \dots$, is the number of the AR order which is used to fit with the time series data and j^{th} , $j = 1, 2, 3, \dots$, is the number of the ACF lag.

<i>AR/MA</i>	0	1	2	3	...
0	$r_1^{(0)}$	$r_2^{(0)}$	$r_3^{(0)}$	$r_4^{(0)}$...
1	$r_1^{(1)}$	$r_2^{(1)}$	$r_3^{(1)}$	$r_4^{(1)}$...
2	$r_1^{(2)}$	$r_2^{(2)}$	$r_3^{(2)}$	$r_4^{(2)}$...
3	$r_1^{(3)}$	$r_2^{(3)}$	$r_3^{(3)}$	$r_4^{(3)}$...
\vdots	\vdots	\vdots	\vdots	\vdots	\ddots

Table 2.2: General ESACF table for the $ARMA(p, q)$

The ESACF values are represented via Table 2.2. for considering the pattern to identify the ARMA order. From Table 2.2, the first row corresponding with $r_j^{(0)}$ for $j = 1, 2, 3, \dots$ gives standard sample ACF with lag j of time series data. The next row and subsequence row give sample ACF which is from residuals of time series data fitted by the AR model. The ARMA order can be identified via Table 2.2 by letting the row be the number of the AR order and the column be the number of the MA order.

Moreover, the ESACF values in Table 2.2 can be transformed to the symbols with “O” representing values greater than or less than ± 2 standard deviations and “X” for values within $\text{mean} \pm 2$ standard deviations. The pattern of ARMA(1,1) represented via the ESACF table is shown in Table 2.3.

<i>AR/MA</i>	0	1	2	3	4	5	...
0	<i>X</i>	<i>X</i>	<i>X</i>	<i>X</i>	<i>X</i>	<i>X</i>	...
1	<i>X</i>	<i>O</i>	<i>O</i>	<i>O</i>	<i>O</i>	<i>O</i>	...
2	<i>X</i>	<i>X</i>	<i>O</i>	<i>O</i>	<i>O</i>	<i>O</i>	...
3	<i>X</i>	<i>X</i>	<i>X</i>	<i>O</i>	<i>O</i>	<i>O</i>	...
4	<i>X</i>	<i>X</i>	<i>X</i>	<i>X</i>	<i>O</i>	<i>O</i>	...
5	<i>X</i>	<i>X</i>	<i>X</i>	<i>X</i>	<i>X</i>	<i>O</i>	...
	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 2.3: General ESACE table for the *ARMA*(1, 1)

2.2.4 Time series models for the nonlinear process

To build a time series model for a nonlinear process, an analyst applies the concept of the autoregressive conditional heteroskedasticity process and the generalized autoregressive conditional heteroskedasticity process. Commonly, both processes are used to capture the non-constant variance of the time series data. The details of these processes are described as follows.

The autoregressive conditional heteroskedasticity process or the ARCH process of order q of the time series data can be explained below. Let ϵ_t be the error terms, σ_t be the standard deviation of the error terms and $z_t \stackrel{iid}{\sim} N(0, 1)$ be a Gaussian white noise process. The ARCH model can be defined as follows

$$\epsilon_t = \sigma_t z_t \quad (2.12)$$

and

$$\sigma_t^2 = \alpha_0 + \alpha_1 \epsilon_{t-1}^2 + \dots + \alpha_q \epsilon_{t-q}^2. \quad (2.13)$$

Given all information $I_{t-1} = \{\epsilon_{t-1}, \epsilon_{t-2}, \dots, \epsilon_1\}$, the conditional variance of ϵ_t can

be written as

$$\text{Var}(\epsilon_t|I_t) = E(\epsilon_t^2|I_t) \quad (2.14)$$

$$= E(\sigma_t^2 z_t^2|I_t) \quad (2.15)$$

$$= E(\sigma_t^2)E(z_t^2|I_t) \quad (2.16)$$

$$= E(\sigma_t^2) \cdot 1 \quad (2.17)$$

$$= E(\sigma_t^2) \quad (2.18)$$

$$= E(\alpha_0 + \alpha_1 \epsilon_{t-1}^2 + \dots + \alpha_q \epsilon_{t-q}^2).$$

Hence, the error terms ϵ_t can be forecasted by the following AR process of order q [16].

Then, if the ARMA process is applied to variance σ_t , the process is called generalized autoregressive conditional heteroskedasticity or GARCH.

The GARCH process of p and q orders can be defined as follows.

$$\epsilon_t = \sigma_t z_t \quad (2.19)$$

where

$$\sigma_t^2 = \alpha_0 + \alpha_1 \epsilon_{t-1}^2 + \dots + \alpha_q \epsilon_{t-q}^2 + \beta_1 \sigma_{t-1}^2 + \beta_2 \sigma_{t-2}^2 + \dots + \beta_p \sigma_{t-p}^2. \quad (2.20)$$

2.3 The concept of the auto-ARIMA model

The auto-ARIMA model [32] is the popular method for fitting the best ARIMA model based on some criteria. The concepts of using the auto-ARIMA model are divided into the likelihood concept part and the automatic identifying order (p, d, q) of the ARIMA model by the criteria part. The details are shown as follows.

2.3.1 The maximum likelihood concept

Next, the concept of the maximum likelihood method is described as follows.

Let x_1, x_2, \dots, x_T be independent random samples from population X which belongs to a family of distribution functions governing by unknown parameter θ of the form $f(x; \theta)$ where $\theta \in \Gamma$.

The likelihood function, $L(\theta; x_1, x_2, \dots, x_t)$ or $L(\theta)$, can be written as

$$L(\theta) = \prod_{i=1}^T f(x_i; \theta). \quad (2.21)$$

The $L(\theta)$ can be simplified to the log-likelihood function as follows.

$$\ln L(\theta) = \sum_{i=1}^T \ln f(x_i; \theta). \quad (2.22)$$

Note $\hat{\theta}$ that maximizes the likelihood function $L(\theta)$ is called the maximum likelihood estimator of θ . It can be written as

$$\hat{\theta} = \arg \max_{\theta \in \Gamma} \ln L(\theta) \quad (2.23)$$

where Γ is the parameter space of θ .

The next example demonstrates the transformation of parameters of the AR(1) model.

Consider the AR(1) model:

$$x_t = c + \phi x_{t-1} + \epsilon_t, \epsilon_t \underset{iid}{\sim} N(0, \sigma^2)$$

or

$$\epsilon_t = x_t - (c + \phi x_{t-1}), \epsilon_t \underset{iid}{\sim} N(0, \sigma^2)$$

for $t = 1, 2, 3, \dots, T$ and $\theta = (c, \phi, \sigma^2)$, $|\phi| < 1$, where ϵ_t satisfies a normal distribution with zero mean and constant variance and has the properties of independent and identically distributed.

Given ϵ_t having a normal distribution, its probability density function is

$$f(\epsilon_t; \theta) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{\epsilon_t}{\sigma}\right)^2} \quad (2.24)$$

$$= \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{(x_t - (c + \phi x_{t-1}) - 0)}{\sigma}\right)^2} \quad (2.25)$$

$$= \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{(x_t - (c + \phi x_{t-1}))^2}{\sigma}\right)}.$$

Hence, the log likelihood function of the AR(1) model can be derived as

$$\begin{aligned} \ln L(\theta) &= \sum_{i=1}^T \ln \left(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{(x_t - (c + \phi x_{t-1}))^2}{\sigma}\right)} \right) \quad (2.26) \\ &= \frac{T-1}{2} \ln(2\pi) - \frac{T-1}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^T (x_t - (c + \phi x_{t-1}))^2. \end{aligned}$$

The log-likelihood function is a non-linear function of the parameters θ which all coefficients of the AR(1) model can be found by maximizing $\ln L(\theta)$. there are various methods to solve the non-linear function, such as Newton's method and the Steepest Descent method.

2.3.2 Automatic identifying orders of the auto-ARIMA model by AIC

The main concept to automatically identify the ARIMA order is to use a criterion that can distinguish a good and a bad forecasting model. One such criteria is the Akaike

information criterion or AIC defining as follows

$$AIC = -2\ln(L) + 2(p + q + k). \quad (2.27)$$

where p is the autoregressive order and q is the moving average order. $k = 1$, if there is an intercept or constant in the ARIMA model and 0 otherwise. $\ln(L)$ is the maximized log likelihood function fitted by the model.

AIC in Equation (2.28) comprises of two parts including the error from the log likelihood function and the number of parameters. The best model for this method should have the minimum AIC value which it means that the errors of the model should be satisfied with the white noise process which causes the log likelihood function value to be high. Nevertheless, AIC also consists of a penalty which is an increasing function of the number of estimated parameters. The penalty in AIC is represented via the parameters part to prevent overfitting. AIC is useful in choosing the ARIMA model automatically. The best model is obtained by minimizing the AIC varying the AR order and the MA order.

The popular method to identify the minimum AIC is to perform the grid search. From the example, thirty six ARMA models varying p order from 0-5 and q order from 0-5 are generated. There are ARMA(1,1), ARMA(1,2), ARMA(1,3), ..., ARMA(5,4) and ARMA(5,5). The table of AICs performing using the grid search is shown in Table 2.4.

		The AR(p) order					
		0	1	2	3	4	5
The MA(q) order	0	3588.66	3588.47	3589.88	3591.62	3592.18	3593.31
	1	3588.61	3584.68	3586.26	3599.26	3590.17	3592.00
	2	3590.03	3586.26	3588.32	3590.25	3590.73	3594.10
	3	3591.88	3589.08	3583.76	3593.01	3589.64	3591.00
	4	3592.88	3590.16	3592.25	3594.10	3583.88	3586.88
	5	3594.05	3590.79	3594.07	3596.02	3586.78	3587.79

Table 2.4: The table of AIC generated from the grid search

From Table 2.4, the minimum AIC of the ARMA model is at order $p = 2$ and order $q = 3$. It means that the ARMA(2,3) model is the best model having less parameters and maintaining the minimal residual.

2.4 The model measurements for the forecasting time series

To assess the model performance, the mean absolute percentage error, the symmetric mean absolute percentage error, the root mean square error and the mean absolute error are used as the model measurements. Their formulae are described next.

Let x_i , \hat{x}_i be the actual data and the forecasting data, respectively and N be the number of the forecasted data.

The mean absolute percentage error or MAPE is the average of the relative difference between the actual time series value and its forecast. It can be written as

$$MAPE = \frac{1}{N} \sum_{i=1}^N \frac{|x_i - \hat{x}_i|}{|x_i|}.$$

The symmetric mean absolute percentage error or SMAPE is the average of the difference between the actual time series value and its forecast divided by the middle value of those two absolute values. It can be written as

$$SMAPE = \frac{1}{N} \sum_{i=1}^N \frac{|x_i - \hat{x}_i|}{\frac{|x_i| + |\hat{x}_i|}{2}}$$

The root mean square error or RMSE is the square root of the average of the squared difference between the actual time series value and its forecast. It can be written as

$$RMSE = \sqrt{\sum_{i=1}^N \frac{(x_i - \hat{x}_i)^2}{N}}$$

The mean absolute error or MAE can be the average of the absolute difference between the actual time series value and its forecast. It can be written as

$$MAE = \sum_{i=1}^N \frac{|x_i - \hat{x}_i|}{N}$$

The next example shows how to use these model measurements to evaluate the forecasting time series. Consider ten forecasting values from three time series models as shown in Figure 2.6.

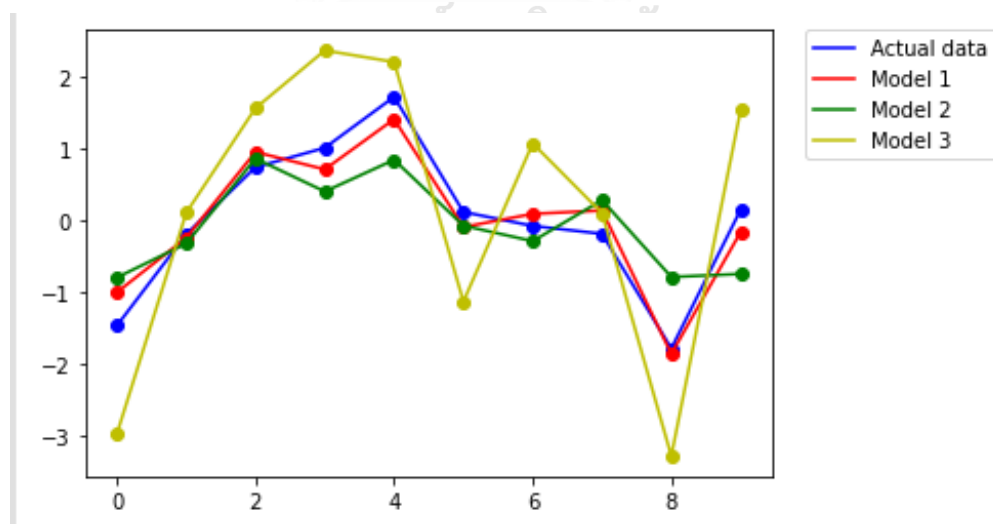


Figure 2.6: The plot of the actual time series data and the three forecasting data

From Figure 2.6, the actual values is $A = \{x_i | i = 1, 2, \dots, 10\} = \{-1.0023, -0.2429, 0.9521, 0.7095, 1.4091, -0.0929, 0.0907, 0.1378, -1.8593, -0.163\}$, the forecasting values of Model 1 is $F1 = \{\hat{x}_i^{(1)} | i = 1, 2, \dots, 10\} = \{-1.0023, -0.2429, 0.9521, 0.7095, 1.4091, -0.0929, 0.0907, 0.1378, -1.8593, -0.163\}$, the forecasting values of Model 2 is $F2 = \{\hat{x}_i^{(2)} | i = 1, 2, \dots, 10\} = \{-0.7959, -0.3235, 0.8604, 0.4019, 0.8428, -0.0733, -0.2893, 0.2813, -0.7882, -0.7461\}$ and the forecasting values of Model 3 is $F3 = \{\hat{x}_i^{(3)} | i = 1, 2, \dots, 10\} = \{-2.9722, 0.1051, 1.5629, 2.366, 2.2047, -1.1296, 1.0722, 0.1012, -3.2793, 1.5384\}$.

To measure the accuracy of forecasting the time series data, MAPE, SMAPE, RMSE and MAE of Model 1 can be calculated as follows.

$$MAPE = \frac{1}{10} \sum_{i=1}^{10} \frac{|x_i - \hat{x}_i^{(1)}|}{|x_i|} \quad (2.28)$$

$$= \frac{1}{10} \left(\frac{|-1.0023 - (-1.0023)|}{|-1.0023|} + \frac{|-0.2429 - (-0.2429)|}{|-0.2429|} + \dots \right) \quad (2.29)$$

$$= 0.92$$

$$SMAPE = \frac{1}{10} \sum_{i=1}^{10} \frac{|x_i - \hat{x}_i^{(1)}|}{\frac{|x_i| + |\hat{x}_i^{(1)}|}{2}} \quad (2.30)$$

$$= \frac{1}{10} \left(\frac{|-1.0023 - (-1.0023)|}{\frac{|-1.0023| + |-1.0023|}{2}} + \frac{|-0.2429 - (-0.2429)|}{\frac{|-0.2429| + |-0.2429|}{2}} + \dots \right) \quad (2.31)$$

$$= 0.94$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{10} (x_i - \hat{x}_i^{(1)})^2}{10}} \quad (2.32)$$

$$= \sqrt{\left(\frac{(-1.0023 - (-1.0023))^2}{10} + \frac{(-0.2429 - (-0.2429))^2}{10} + \dots \right)} \quad (2.33)$$

$$= 0.27$$

$$MAE = \frac{\sum_{i=1}^{10} |x_i - \hat{x}_i^{(1)}|}{10} \quad (2.34)$$

$$= \sqrt{\left(\frac{|-1.0023 - (-1.0023)|}{10} + \frac{|-0.2429 - (-0.2429)|}{10} + \dots \right)} \quad (2.35)$$

$$= 0.27$$

For Model 2 and Model 3, MAPE, SMAPE, RMSE and MAE can be calculated similar to Model 1. By following these model measurements, MAPE, SMAPE, RMSE and MAE of Model 1, 2 and 3 are shown in Table 2.5.

จุฬาลงกรณ์มหาวิทยาลัย

	MAPE	SMAPE	RMSE	MAE
Model 1	0.92	0.94	0.27	0.24
Model 2	1.59	1.07	0.61	0.52
Model 3	4.27	1.27	1.11	1.0

Table 2.5: MAPE, SMAPE, RMSE and MAE of forecasting time series data

The best forecasting model should provide the smallest value of MAPE, SMAPE, RMSE and MAE, From Table 2.5, it can be concluded that Model 1 is the best in forecasting the time series data.

2.5 The statistical test of the forecasting time series

To confirm the performance of the forecasting time series model, the Ljung-Box test is introduced to test the correlation within the residuals after fitting the time series model. The Ljung-Box test, introduced by Greta M. Ljung and George E. P. Box, is a statistical test of whether autocorrelations of time series data are different from zero. The Ljung-Box test is widely used in econometrics and other applications of time series analysis for testing the residuals of forecasting the time series. The Ljung-Box test can be defined as:

- H_0 : The data are independently distributed or the correlation in the population from the sample is 0.
- H_1 : The data are not independently distributed; their correlations are not equal to 0.

The test statistic of the Ljung-Box test is defined as

$$Q(h) = n(n+2) \sum_{k=1}^h \frac{\hat{\nu}_k^2}{n-k}$$

where n is the sample size, $\hat{\nu}_k$ is the sample autocorrelation of the residual at lag k , and h is the number of lags being tested. For significance level α , the null hypothesis H_0 is rejected when $Q(h) > \chi_{1-\alpha, h}^2$. For the alternative way to test the hypothesis, the p -value can be used which it is generated from the Chi-squared table. H_0 is rejected when the p -value $< \alpha$.

Next, the example of the Ljung-Box test to test the residual of forecasting time series is shown. From the example, there are two residuals occurring from forecasting by two time series models as shown in Figure 2.7 and Figure 2.8.

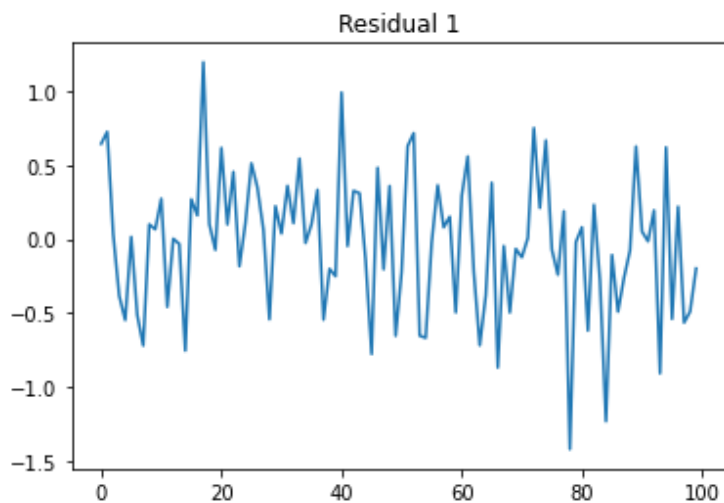


Figure 2.7: Residual 1 plot

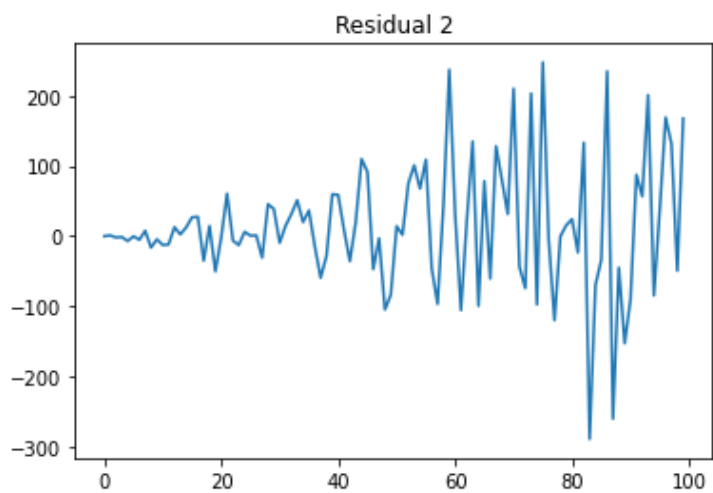


Figure 2.8: Residual 2 plot

Both residuals can be evaluated independent by calculating sample ACF as shown in Section 2.1. The ACF plots of two residuals are shown in Figure 2.9 and Figure 2.10.

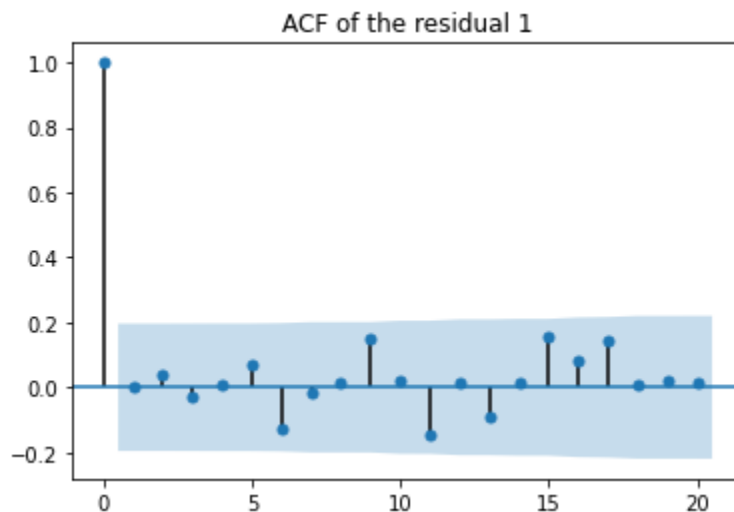


Figure 2.9: The ACF plot of residual 1

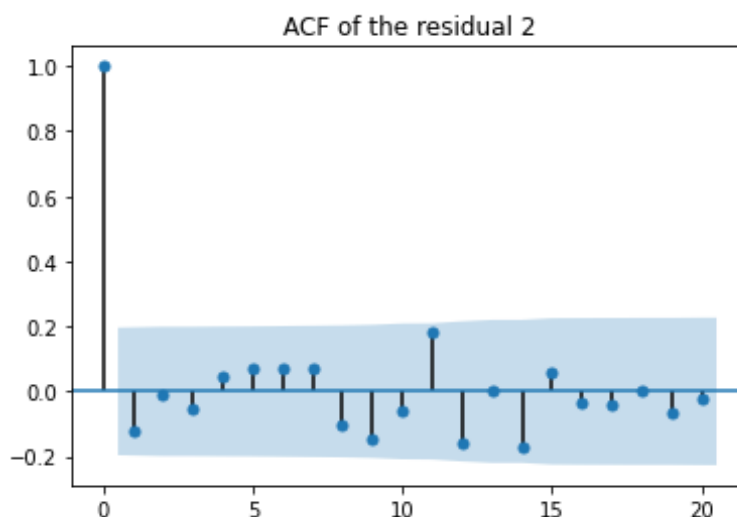


Figure 2.10: The ACF plot of residual 2

From Figure 2.7 and Figure 2.8, sample ACF of residual 1 and 2 for 20 lags is $ACF^{(1)} = \{\nu_k^{(1)} | k = 0, 2, \dots, 19\} = \{1.0, 0.0002, 0.0358, -0.0313, 0.0059, 0.072, -0.1284, -0.0168, 0.0149, 0.1485, 0.0216, -0.1478, 0.0132, -0.0896, 0.0162, 0.1588, 0.081, 0.1415, 0.0063, 0.0201, 0.0127\}$ and $ACF^{(2)} = \{\nu_k^{(2)} | k = 0, 2, \dots, 19\} = \{1.0, -0.1201, -0.0081, -0.0557, 0.0474, 0.071, 0.069, 0.0715, -0.1016, -0.1456, -0.0571, 0.1835, -0.1594, 0.0041, -0.1719, 0.0574, -0.0356, -0.0403, -0.0, -0.069, -0.0255\}$, respectively.

Next, the statistical test of Ljung Box is used to test independent of residual 1 and 2 from sample ACF. For the statistical values, $(Q^{(1)}(h))$, of the Ljung Box test from residual 1 can be calculated as follows.

$$\begin{aligned} Q^{(1)}(1) &= 100(100 + 2) \sum_{k=1}^1 \frac{(\nu_k^{(1)})^2}{100 - k} \\ &= 0 \end{aligned} \tag{2.36}$$

$$\begin{aligned} Q^{(1)}(2) &= 100(100 + 2) \sum_{k=1}^2 \frac{(\nu_k^{(1)})^2}{100 - 2} \\ &= 0.13325 \end{aligned} \tag{2.37}$$

$$\begin{aligned} Q^{(1)}(3) &= 100(100 + 2) \sum_{k=1}^3 \frac{(\nu_k^{(1)})^2}{100 - 3} \\ &= 0.236 \end{aligned} \tag{2.38}$$

For the statistical values, $(Q^{(2)}(h))$, of the Ljung Box test from residual 2 can be calculated as follows.

$$\begin{aligned} Q^{(2)}(1) &= 100(100 + 2) \sum_{k=1}^1 \frac{(\nu_k^{(2)})^2}{100 - k} \\ &= 1.41307 \end{aligned} \tag{2.39}$$

$$\begin{aligned}
 Q^{(2)}(2) &= 100(100 + 2) \sum_{k=1}^2 \frac{(\nu_k^{(2)})^2}{100 - 2} \\
 &= 2.91687
 \end{aligned}
 \tag{2.40}$$

$$\begin{aligned}
 Q^{(2)}(3) &= 100(100 + 2) \sum_{k=1}^3 \frac{(\nu_k^{(2)})^2}{100 - 3} \\
 &= 10.46637
 \end{aligned}
 \tag{2.41}$$

Each statistical value, $Q(h)$, can be checked by the Chi-squared test with significance level α and h degrees of freedom. The null hypothesis is rejected at $Q^{(2)}(3)$ since $Q^{(2)}(3) > \chi_{1-\alpha,3}^2$. It means that residual 2 is dependent when sample ACF is accumulated after lag 3. It can be concluded that the model that generates residual 2 is not good for forecasting.

For the alternative test of the hypothesis, the p -value generating by the Chi-squared table is introduced. H_0 is rejected when the p -value < 0.05 . The p -values of each $Q(h)$ are plot in Figure 2.11 and Figure 2.12.

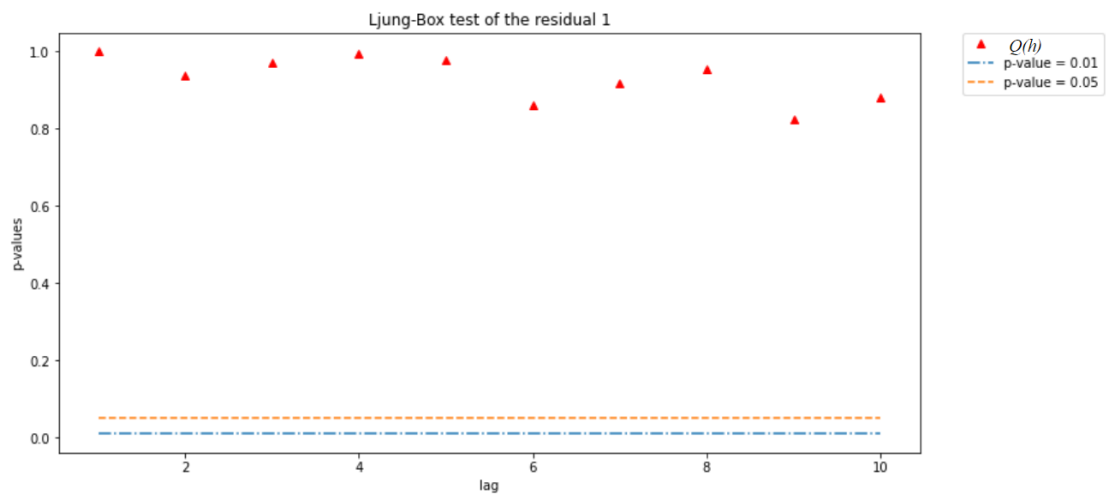


Figure 2.11: The p -values plot of residual 1

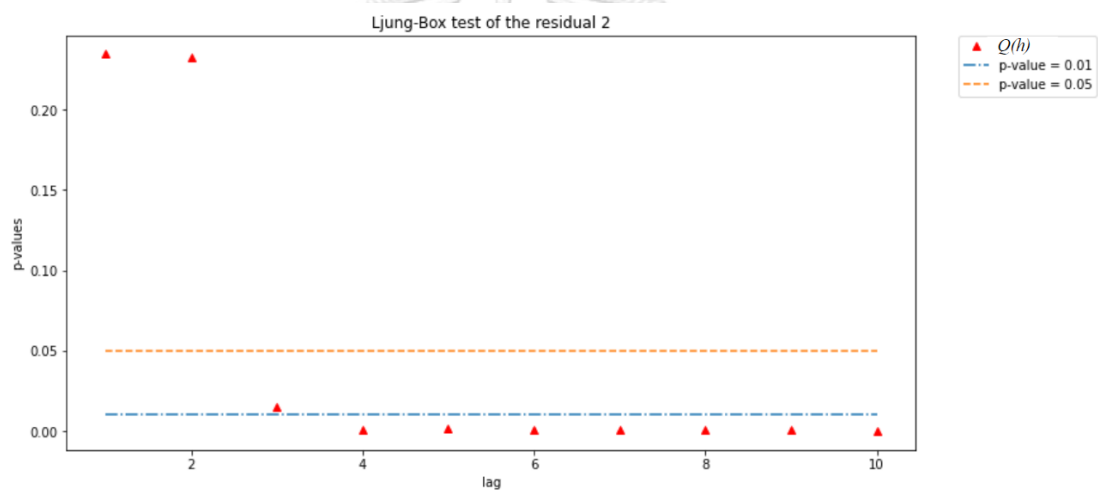


Figure 2.12: The p -values plot of residual 2

From Figure 2.11 and Figure 2.12, It is shown that p -values from residual 1 is above 0.05 for every lag of ACF whereas p -values from residual 2 is below 0.05 after lag 3 of ACF. It can be concluded that the model giving residual 1 is more powerful in forecasting than the model giving residual 2.

CHAPTER III

BACKGROUND KNOWLEDGE OF DEEP LEARNING

This chapter demonstrates the necessary background knowledge of deep learning using in this dissertation. Firstly, the concept of deep learning is demonstrated in Section 3.1. Secondly, the process of updating weights by the backpropagation is explained in Section 3.2. Thirdly, a review on convolutional neural networks is demonstrated in Section 3.3. Fourthly, the convolutional neural network concept and the pooling concept are explained in Section 3.4. Lastly, the model measurements for the classification problems are introduced in Section 3.5.

3.1 The concept of deep learning

Deep learning is a part of machine learning methods based on an artificial neural network or ANN which is inspired by information processing and distributed communication nodes in biological brain systems. The definition of deep learning is introduced by Deng and Yu [33] in 2014. The quote definition of deep learning is “a class of machine learning algorithms that uses multiple layers to progressively extract higher level features from the raw input. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces”. Moreover the deep learning models arise from the combination of ANN and the concept of the convolutional neural networks or CNN. In deep learning, each layer attempts to learn for transforming the input data to extract the features which are in the inputs. In general, deep learning is applied to the image classification which the input data is a matrix of pixels.

The basic structure of ANN comprises of three parts which are the input layer, the

hidden layer and the output layer. The nodes in each layer of ANN are connected by edges along their weights. The basic structure of ANN is shown in Figure 3.1.

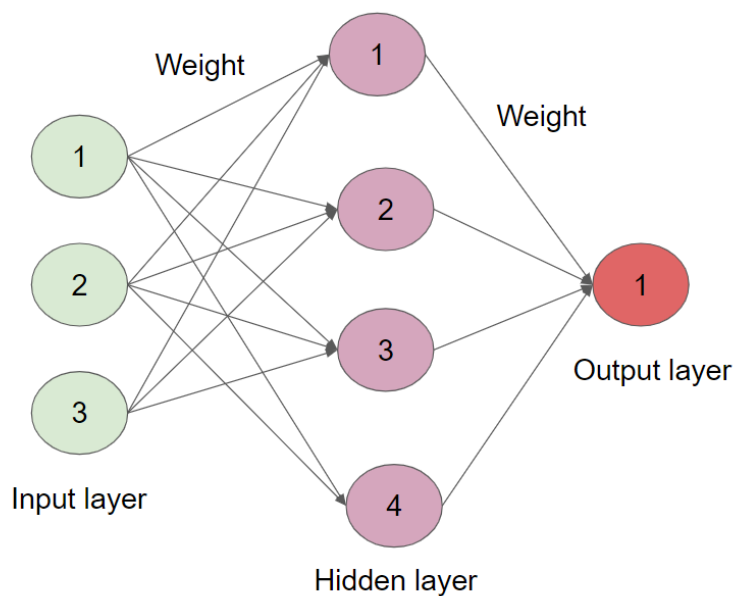


Figure 3.1: The structure of ANN

The structure in each node in the hidden layer is demonstrated via the example in Figure 3.2. All inputs x_1, x_2, x_3 are sent to the node in the hidden layer while the sum of the products between inputs and weights is sent to the activation function before submitting the output y .

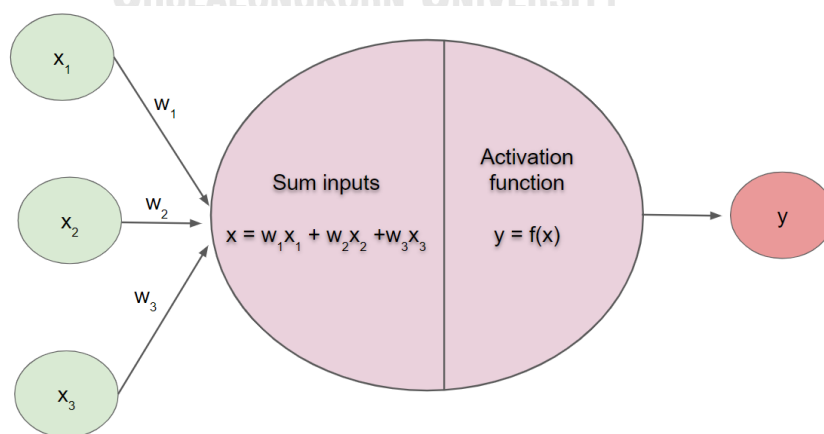


Figure 3.2: The structure of a node in the hidden layer

In an artificial neural network, the activation function is used to define the output. Next, two popular activation functions are described which are the sigmoid function and the relu function.

The sigmoid function is a function having a characteristic “S”-shaped curve defined as follows

$$f(x) = \frac{1}{1 + e^{-x}}$$

The domain of the sigmoid function is all real numbers. The values of the sigmoid function in range 0-1 are shown in Figure 3.3.

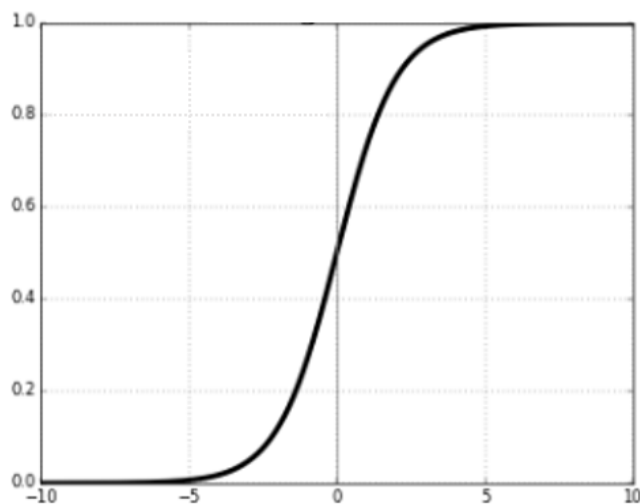


Figure 3.3: The sigmoid function

Next, the relu function is defined as follows

$$f(x) = \max\{0, x\},$$

$f(x)$ is set to be 0 when x is a negative value and $f(x)$ is equal to x when x is a positive value or 0 as shown in Figure 3.4.

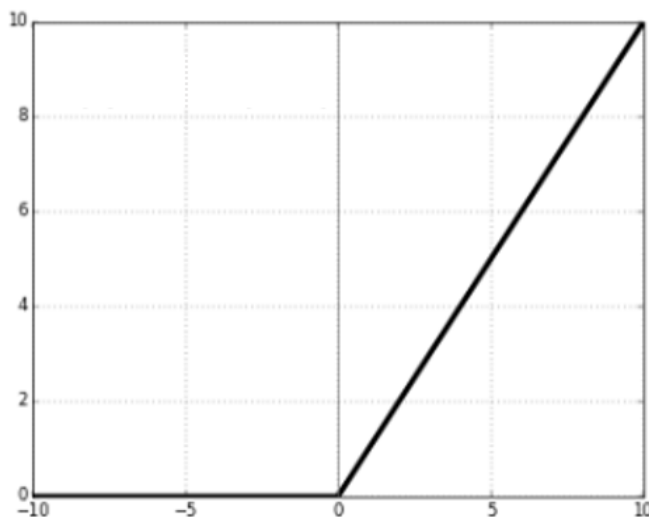


Figure 3.4: The relu function

3.2 The backpropagation concept

This section covers the method for training an artificial neural network to update weights in the deep learning model. The method is called the backpropagation which uses the gradient of the error function with respect to the neural network's weights to update the new weights. The whole process is shown via the example next.

The neural network is defined by two inputs, one hidden node and one output. The input values of x_1 and x_2 are set to be 0.1 and 0.3, respectively. The initial weights w_1 and w_2 are set to be 0.2 and 0.6, respectively and the target of output is 0.5. The process is shown in Figure 3.5.

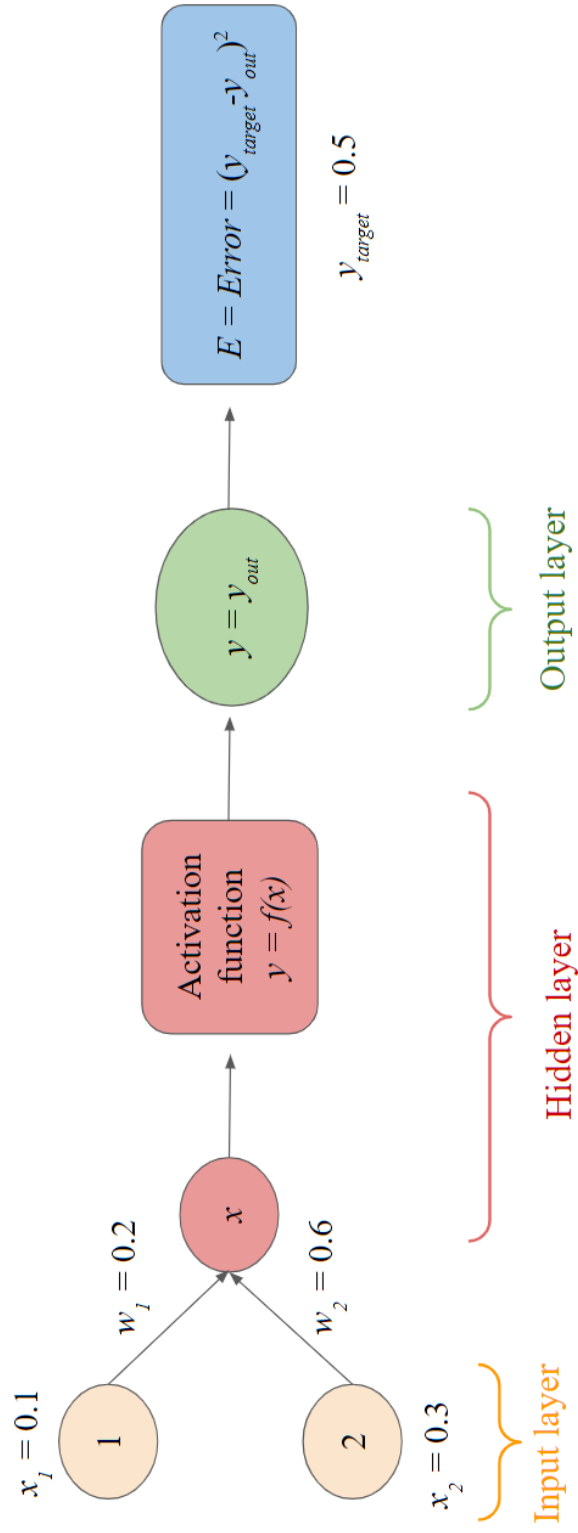


Figure 3.5: A example of the neural network

The goal of backpropagation is to optimize the weights of this neural network. This process is divided into three parts including “The Forward Pass”, “Calculating the Total Error”, and “The Backward Pass”.

The Forward Pass

From Figure 3.5, it begins with calculating x in the hidden layer as follows:

$$x = w_1 \times x_1 + w_2 \times x_2 \quad (3.1)$$

$$= 0.2 \times 0.1 + 0.6 \times 0.3 \quad (3.2)$$

$$= 0.2$$

Then, the output y_{out} can be calculated by the activation function. For this example, the sigmoid function is applied as follows:

$$y_{out} = f(x) \quad (3.3)$$

$$= f(0.2) \quad (3.4)$$

$$= \frac{1}{1 + e^{-0.2}} \quad (3.5)$$

$$= 0.55$$

After this step, the output y_{out} is obtained from the hidden layer.

Calculating the Total Error

After the previous step, the output y_{out} is used to calculate the error as follows:

$$E = Error \quad (3.6)$$

$$= (y_{target} - y_{out})^2 \quad (3.7)$$

$$= (0.5 - 0.55)^2 \quad (3.8)$$

$$= 0.0025$$

The Backward Pass

The goal with backpropagation is to update each weight in the neural network. For this example, the weights w_1 and w_2 are updated as follows:

Consider w_1 , the effect of w_1 to the error can be considered by the partial derivative of E or $\frac{\partial E}{\partial w_1}$ where E is the composite function of y_{out} and x which are differentiable. By applying this with the chain rule, $\frac{\partial E}{\partial w_1}$ can be calculated as follows:

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial y_{out}} \times \frac{\partial y_{out}}{\partial x} \times \frac{\partial x}{\partial w_1}$$

where

จุฬาลงกรณ์มหาวิทยาลัย
CHULALONGKORN UNIVERSITY

$$\frac{\partial E}{\partial y_{out}} = \frac{\partial (y_{target} - y_{out})^2}{\partial y_{out}} = -2(y_{target} - y_{out}),$$

$$\frac{\partial y_{out}}{\partial x} = \frac{\partial y_{out}}{\partial x} = \frac{\partial \frac{1}{1+e^x}}{\partial x} = x(1-x)$$

and

$$\frac{\partial x}{\partial w_1} = \frac{\partial (w_1 \times x_1 + w_2 \times x_2)}{\partial w_1} = x_1$$

Hence, $\frac{\partial E}{\partial w_1}$ can be calculated as follows:

$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial y_{out}} \times \frac{\partial y_{out}}{\partial x} \times \frac{\partial x}{\partial w_1} \quad (3.9)$$

$$= -2(y_{target} - y_{out}) \times x(1 - x) \times x_1 \quad (3.10)$$

$$= -2(0.5 - 0.55) \times 0.2(1 - 0.2) \times 0.1 \quad (3.11)$$

$$= 0.0016$$

To decrease the error, the weights of the neural network have to be updated.

$$w^{new} = w^{old} - \alpha \frac{\partial E}{\partial w^{old}}$$

where α represents the learning rate. For this example, α is set to be 0.5. Therefore, weight w_1 can be updated as follows:

$$w_1^{new} = w_1^{old} - \alpha \frac{\partial E}{\partial w_1^{old}} = 0.2 - 0.5 \times 0.0016 = 0.1992$$

Then, weight w_2 can be calculated the same way as weight w_1 . The process of updating w_1 and w_2 is repeated by the backpropagation concept until the error of the neural network converges to 0.

3.3 A review on convolutional neural networks

Afterward, a convolutional neural network or CNN is applied to the deep learning model to increase the efficiency of images classification. There are several convolutional neural networks or CNN architectures appearing in the current research.

In 1983, LeCun et al. proposed the LeNet architecture to classify digits as review in [34]. Emerging architectures are proposed due to a large benchmark collection from ImageNet [35]. AlexNet was introduced by Krizhevsky et al. in 2012 [34]. This architecture consists of 5 convolutional layers using 96 filter matrices of size 11×11 and 3 fully connected layers. Later, VGGNet was proposed by Simonyan and Zisserman in 2014 [36]. The number of layers of VGGNet can vary depending on the type of the VGGNet models. For example, VGG16 has 16 layers of convolutional layers and max-pooling layers. The size of filter matrices of this model is 3×3 and the model uses only non-overlapping max-pooling. Then GoogLeNet was proposed in 2014 by Google [37]. This model is built from the inspiration of LeNet but added the new concept of CNN, called inception module to improve feature extraction of the model. Moreover, ResNet was introduced by Kaiming He et al [34] in 2015 which it uses the concept of skip connection which is gated recurrent units in RNN and applies batch normalization in the convolutional layers. However, their basic structure of CNN is similar. The basic structure of CNN is called the convolutional block consisting of the convolutional layer and the pooling layer. Finally, the fully connected layer is used according to the concept of the artificial neural network for computing the outputs. The whole process of the deep learning model applying with the CNN is shown Figure 3.6.

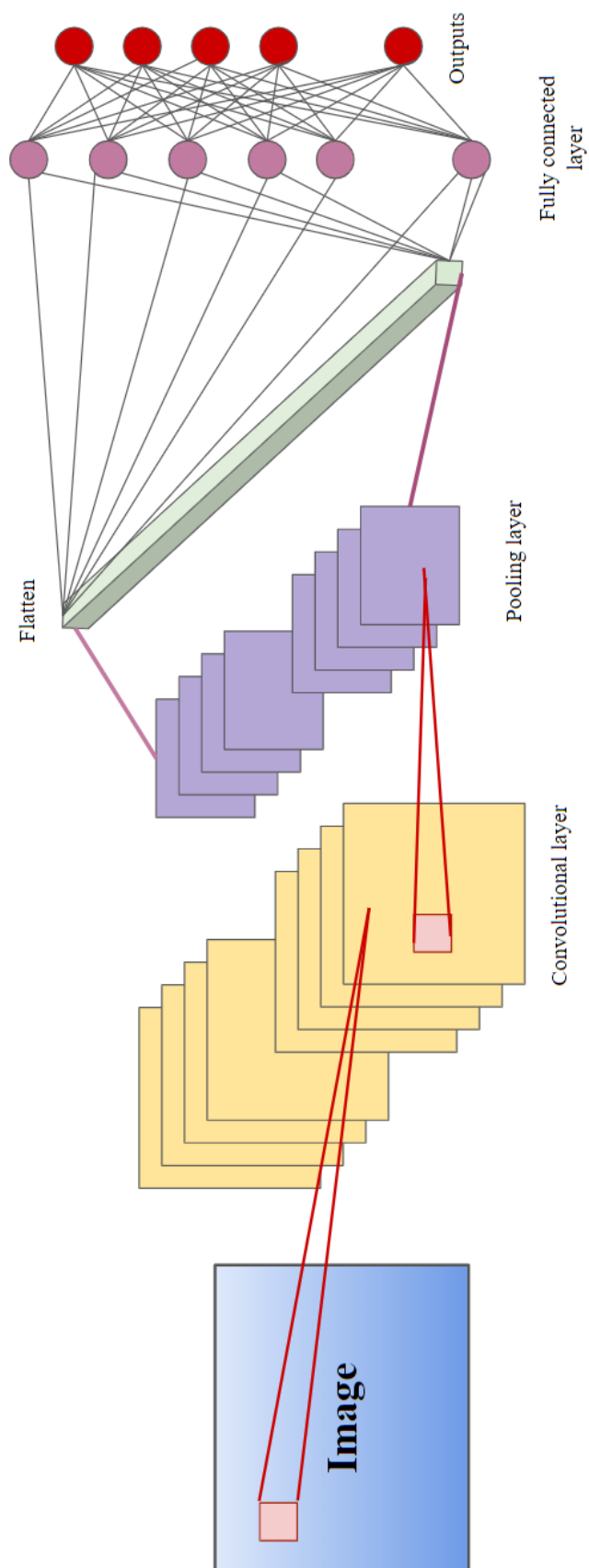


Figure 3.6: The structure of the deep learning model

Presently, CNN has been used in the classification of time series and time series forecasting. In 2018, Liu et al [38] used the multivariate convolutional neural network models for time series classification of the prognostics and health management. In the same year, Koprinska et al [39] used the convolutional neural network model for forecasting four solar and electricity time series from Australia, Portugal and Spain. In 2020, Wang et al [40] introduced the new recurrent convolutional neural network model to forecast photovoltaic solar power and electricity load for the next day. This research uses the concept of CNN to identify the ARIMA order and the SARIMA order using ESACF, PACF, ACF and differencing time series as inputs.

3.4 The convolutional neural network concept and the pooling concept

The concept of convolutional layer is to learn features from the input images which is composed of several convolutional kernels or filter matrices. Consider the following example in Figure 3.7.

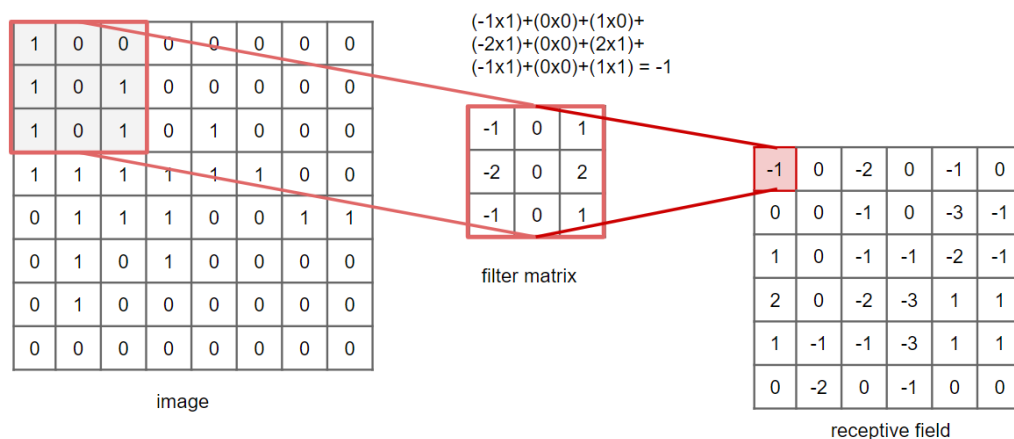


Figure 3.7: The convolution concept

From Figure 3.7, the values in the image matrix are multiplied with the filter matrix of size 3×3 with known elements and add them together as the output, called a receptive field.

Then, the concept of pooling is applied for reducing the resolution of the images

and pulling out the distinctive features of the image. In practice, the concept of max-pooling is popularly used as shown in Figure 3.8 which will replace a block of the matrix by the largest values on the receptive field. The output of the max-pooling layer is called a feature map. In this example, the max-pooling size is set to 2×2 .

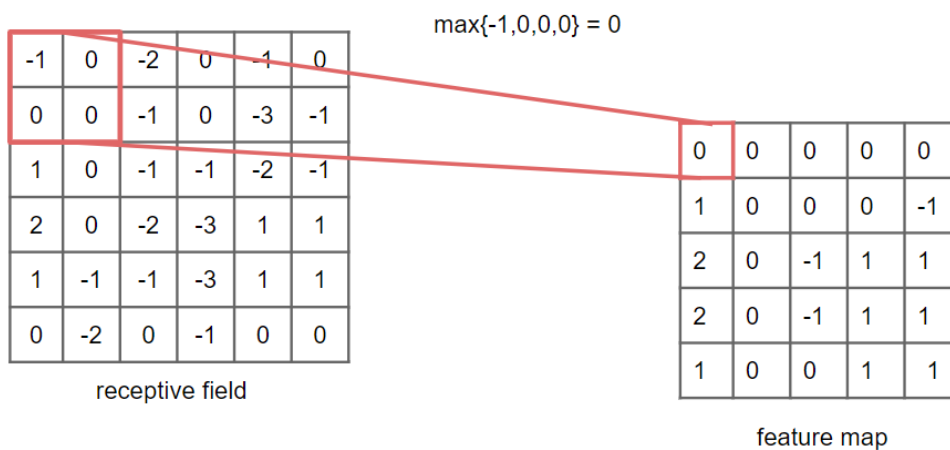


Figure 3.8: The max-pooling concept

3.5 The model measurements for the classification problems

The model measurements for the classification problems using in this dissertation consist of the precision, the recall and the f1-score describing as follows.

Before describing the formulae of the precision, the recall and the F1-score, the confusion matrix is introduced as follows.

		Predicted class	
		Class = Yes	Class = No
Actual class	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

Table 3.1: The confusion matrix

where

- True Positives or TP: When the value of the actual class is yes and the value of the predicted class is also yes.
- True Negatives or TN: When the value of the actual class is no and the value of the predicted class is also no.
- False Positives or FP: When the actual class is no and the predicted class is yes.
- False Negatives or FN: When the actual class is yes but the predicted class is no.

Therefore, the precision, the recall and the f1-score can be calculated as follows

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$f1 - score = 2 \times \frac{precision \times recall}{precision + recall}$$

For the example, the confusion matrices of model 1 and 2 are set as Table 3.2 and

3.3.

		Predicted class	
		Class = Yes	Class = No
Actual class	Class = Yes	30	10
	Class = No	5	55

Table 3.2: The confusion matrix of model 1

		Predicted class	
		Class = Yes	Class = No
Actual class	Class = Yes	35	5
	Class = No	25	35

Table 3.3: The confusion matrix of model 2

The precision, the recall and the f1-score of model 1 can be computed as follows

$$precision = \frac{TP}{TP + FP} = \frac{30}{30 + 5} = 0.87$$

$$recall = \frac{TP}{TP + FN} = \frac{30}{30 + 10} = 0.75$$

$$f1 - score = 2 \times \frac{precision \times recall}{precision + recall} = 2 \times \frac{0.87 \times 0.75}{0.87 + 0.75} = 0.80$$

The precision, the recall and the f1-score of model 2 can be computed as follows

$$precision = \frac{TP}{TP + FP} = \frac{35}{35 + 25} = 0.58$$

$$recall = \frac{TP}{TP + FN} = \frac{35}{35 + 10} = 0.77$$

$$f1 - score = 2 \times \frac{precision \times recall}{precision + recall} = 2 \times \frac{0.58 \times 0.77}{0.58 + 0.77} = 0.66$$

A good model for the classification problem should have both the high precision and the high recall. From the experimental results of this example, model 2 has the high recall but has the low precision while model 1 has the high scores of both the precision and the recall. Therefore, it can be concluded that model 1 is more effective in classification than model 2. Moreover, when considering the average score between the precision and the recall by using the f1-score, model 1 clearly has the higher f1-score.



CHAPTER IV

METHODOLOGY TO IDENTIFY THE ARIMA ORDER

This chapter explains the process to build the deep learning models for identifying the ARIMA order. The first section covers the deep learning architecture and the training process of the self-identification deep learning model or the SID model using simulated time series data. Then, the SID model is extended to the self-identification ResNet-ARIMA model or the SIRA model by using the architecture of ResNet which is explained in the second section. The third section is to describe the ACF-PACF-ESACF convolutional neural network ARIMA order identification model or the APEA model using ESACF as a part of inputs.

4.1 The self-identification deep learning model

This section covers the motivation and the process of constructing the SID model and its experimental results. The details are shown as follows.

4.1.1 The architecture of the SID model

In the time series analysis, the ARIMA model is normally used for forecasting the time series data. It is built via two important steps which are (1) the model identification to recognize the ARIMA order and (2) the fit coefficients of the ARIMA model. To apply the deep learning model to the model identification, the deep learning model must learn a large number of the time series inputs having the correct ARIMA order. The simulated ARIMA time series data are synthesized for training the deep learning model. Nevertheless, using the raw time series data as the inputs may not help the deep learning model to identify the ARIMA order accurately. Therefore, the ACF plot and the PACF

plot are used. In this research, the raw time series data are used to generate the ACF plot and the PACF plot before submitting to the deep learning model.

In constructing a deep learning model, the first step begins with assigning the input data for training the deep learning model and their labels. In this research, the simulated ARMA time series data are used for training the pq -SID model to learn the ARMA order. After the pq -SID model is constructed, the training time series data are reused for differencing to learn the differencing order. Consequently, this subsection explains the architecture of the SID model and its training process using simulated time series data. The SID model is split into two parts which are the pq -SID model using for identifying the ARMA order from 0-5 and the d -SID model using for identifying the differencing order from 0-2.

Identifying the ARMA order requires the PACF plot and the ACF plot instead of the time series data alone. In the time series analysis, PACF and ACF are useful for identifying the ARMA order where PACF is used for identifying the AR order and the ACF is used for identifying the MA order. Therefore, the inputs of the pq -SID model comprise of three channels of 50×50 black and white images. The first channel is an image of the ACF plot, the second channel is an image of the PACF plot and the third channel is the time series image. The pq -SID model consists of 6 convolutional layers applying the max-pooling layer over two adjacent convolutional layers. The convolutional layers have 64 of 3×3 filter matrices with the stride = 1 and the max-pooling layer has a size matrix of 2×2 and the stride is set to 2. It is followed by two fully connected layers having 512 and 1024 nodes, respectively. Finally, the final layer is softmax applied with all possible orders of p from 0-5 and q from 0-5.

For the case of identifying the differencing order, the deep learning architecture of the d -SID model is used only the inputs of differencing. Since, in identifying the differencing order, ACF can be used to indicate the trends within the time series data. Therefore, the d -SID model consists of 3 channels of the ACF plots from time series taking differencing to time series from $d = 0$ to 2. For the final layer, softmax is applied.

The architectures of the pq -SID model and the d -SID model are shown in Figure 4.1 and Figure 4.2.



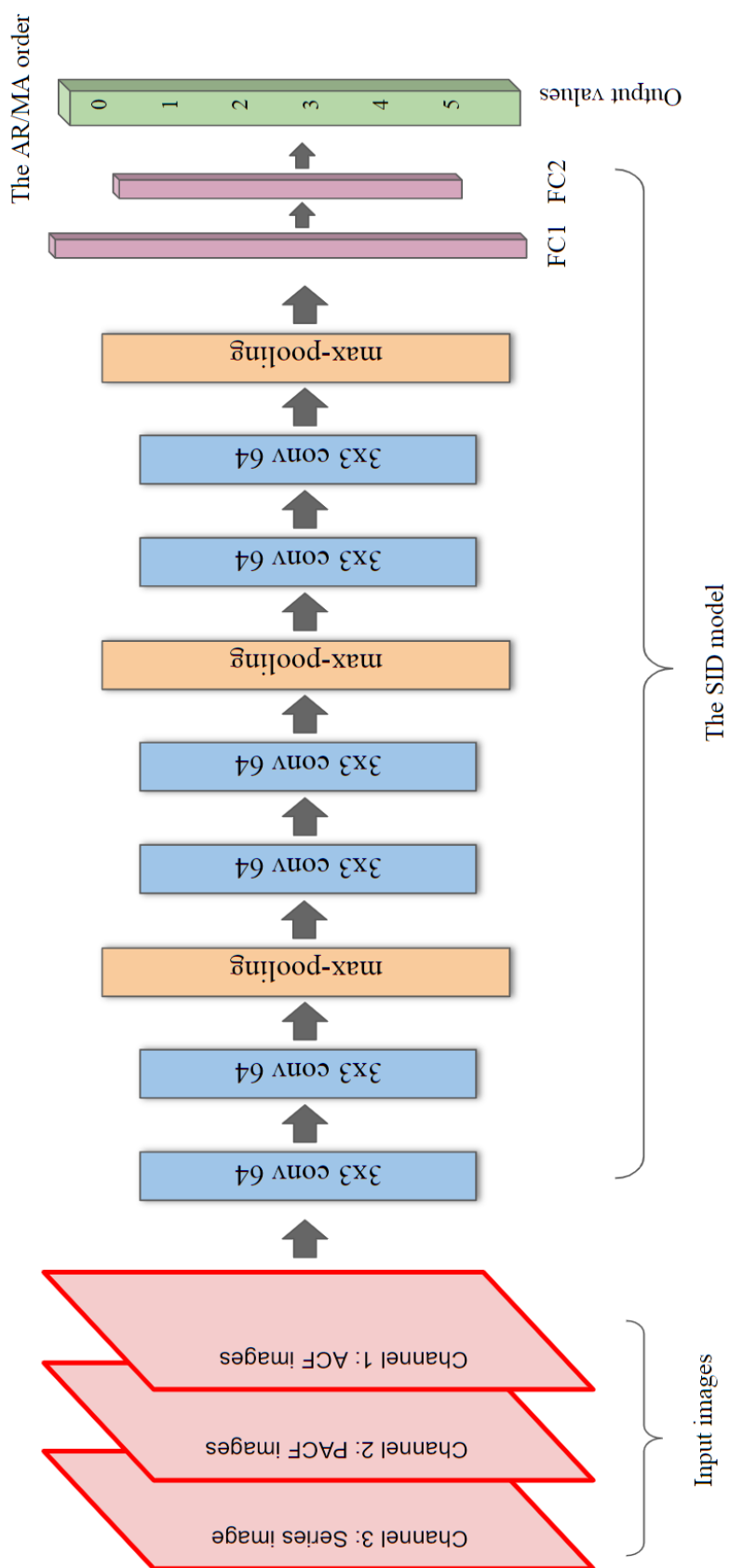


Figure 4.1: The architecture of the pq -SID model for identifying the AR and MA order

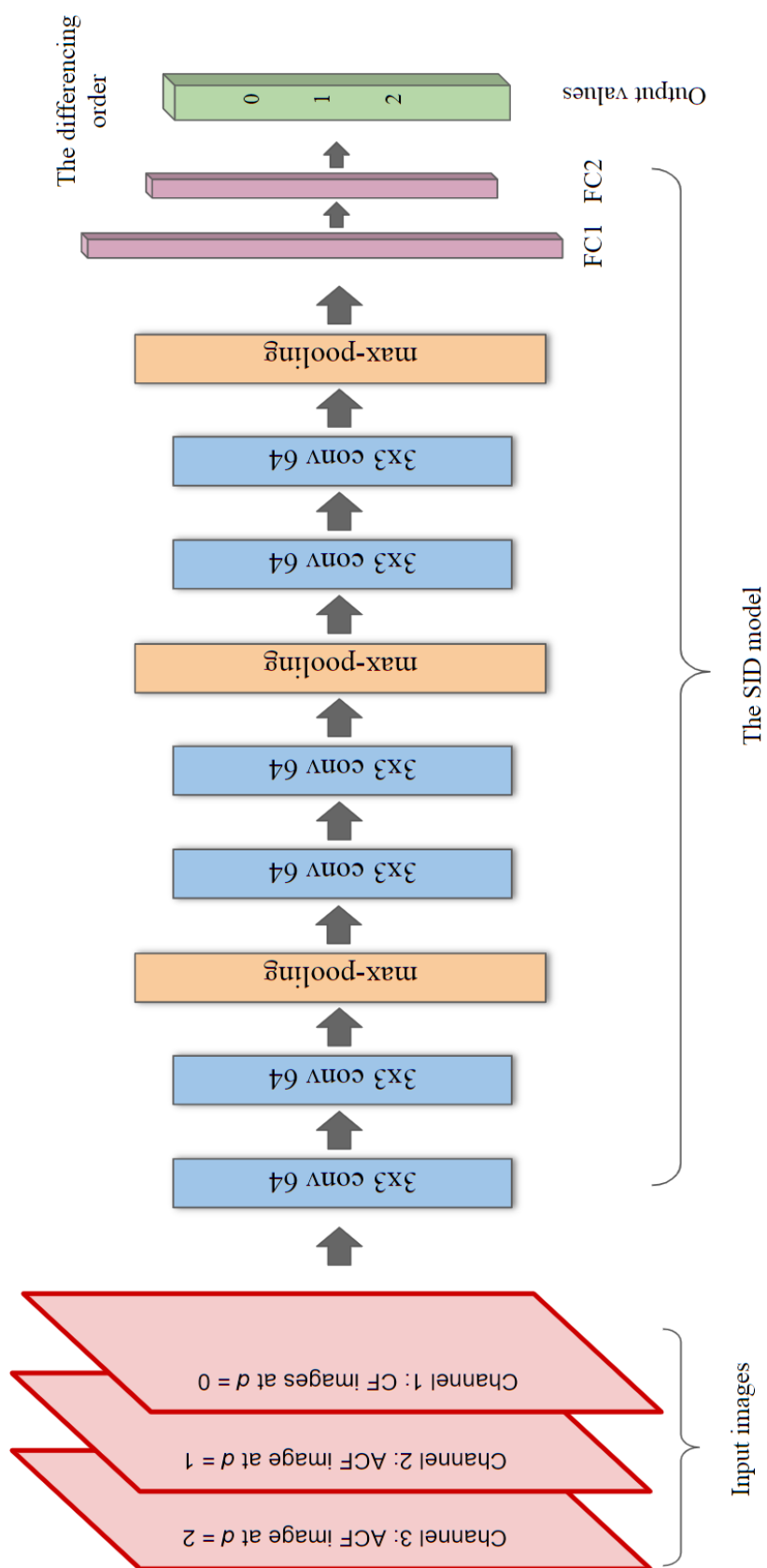


Figure 4.2: The architecture of the d -SID model for identifying the differencing order

In the training process of these SID models, all simulating time series data are converted to their corresponding images which are split to the training data, the validating data and the testing data. The training data is used for updating weights of the deep learning model. The validating data is used for evaluating the accuracy occurring in training the deep learning model. The weights in the deep learning model are updated until the accuracy of the training data and the validating data converge. After finishing training the deep learning model, the testing data is used to measure the performance of the obtained SID model.

In this research, the simulated time series data for training the pq -SID model are generated from the ARMA process by varying p from 0 to 5 and q from 0 to 5. Moreover, all coefficients of the process must be randomly generated to satisfy the stationary property and the invertibility property from the uniform distribution within range -1 to 1. If the stationary property or invertibility property is not satisfied, all coefficients will be regenerated. The number of the time series data in the training data, the validating data and the testing data are 7200, 3600 and 720, respectively. The process and the example of time series are shown in Figure 4.3.

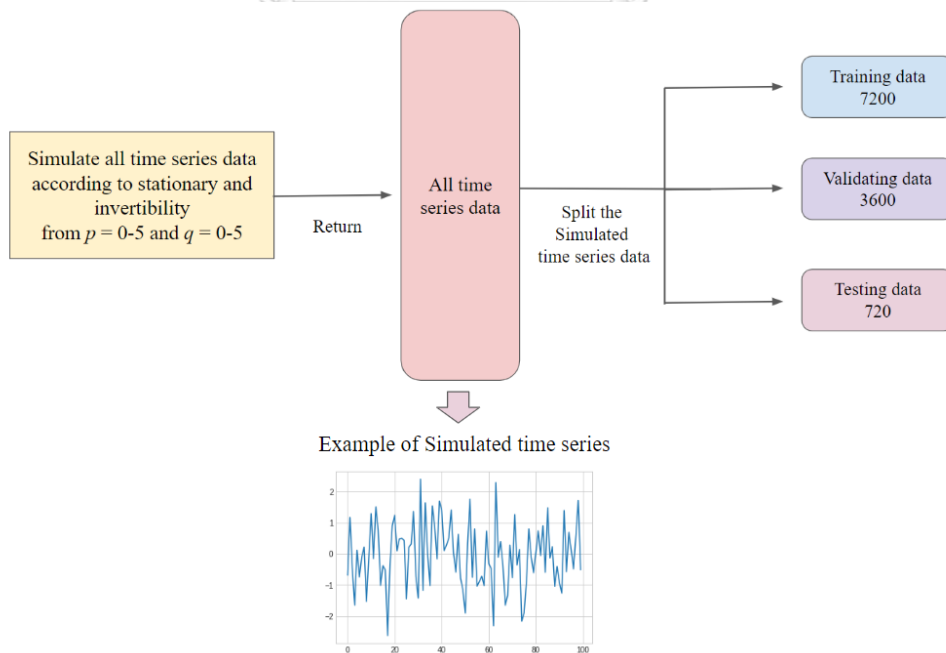


Figure 4.3: The process of the simulated time series data of the ARMA order and the example of the time series data.

In the process of simulated time series for the differencing order to train the d -SID model, the time series data are simulated by Equation (4.1)

$$y_{t+1} = x_t + y_t \quad (4.1)$$

where x_t is the simulated data from the ARMA process and y_t is the simulated data having differencing $d = 1$. The initial value y_0 is random from the uniform distribution within range -1 to 1.

Next, to prove that y_t satisfies to the ARIMA($p, 1, q$) process. Let x_t be the time series satisfying the ARIMA($p, 0, q$) process and y_t be the time series satisfying with Equation (4.1). Hence, x_t can be written as

$$\phi_p(B)x_t = \theta_q(B)\epsilon_t \quad (4.2)$$

where x_t and ϵ_t are the time series data and random error at time t . From Equation (4.1), it can be rewritten as

$$x_t = y_{t+1} - y_t = \nabla y_t \quad (4.3)$$

Substitute ∇y_t into the x_t in Equation (4.2).

$$\phi_p(B)\nabla y_t = \theta_q(B)\epsilon_t \quad (4.4)$$

From Equation (4.4), y_t is the time series satisfying with the ARIMA($p, 1, q$).

To calculate y_t having differencing $d = 2$, x_t is set to be the simulated data having differencing $d = 1$. The example of simulating time series data having $d = 1$ is shown as

follows.

Let $X = \{x_t | t = 0, 1, 2, \dots, 19\} = \{0.63, -0.45, -0.38, 0.02, 0.22, 0.66, -0.14, 0.86, -0.27, -0.05, 0.33, 0.83, 0.06, -0.41, 0.46, -0.08, -0.32, 0.69, 0.43, -0.83\}$ be the simulated time series data from the ARMA process and $Y = \{y_t | t = 0, 1, 2, \dots, 19\}$ be the simulated data having differencing $d = 1$ and the initial value y_0 be 0.66. The elements in Y can be calculated as follows.

$$y_1 = x_0 + y_0 \quad (4.5)$$

$$= 0.63 + 0.66 \quad (4.6)$$

$$= 1.29$$

$$y_2 = x_1 + y_1 \quad (4.7)$$

$$= -0.45 + 1.29 \quad (4.8)$$

$$= 0.84$$

$$y_3 = x_2 + y_2 \quad (4.9)$$

$$= -0.38 + 0.84 \quad (4.10)$$

$$= 0.46$$

For the rest of elements in Y can be generated similarly. Hence, Y is $\{0.66, 1.29, 0.84, 0.46, 0.48, 0.7, 1.36, 1.22, 2.08, 1.81, 1.76, 2.09, 2.92, 2.98, 2.57, 3.03, 2.95, 2.63, 3.32, 3.75, 2.92\}$. The time series plot of this example is shown in Figure 4.4.

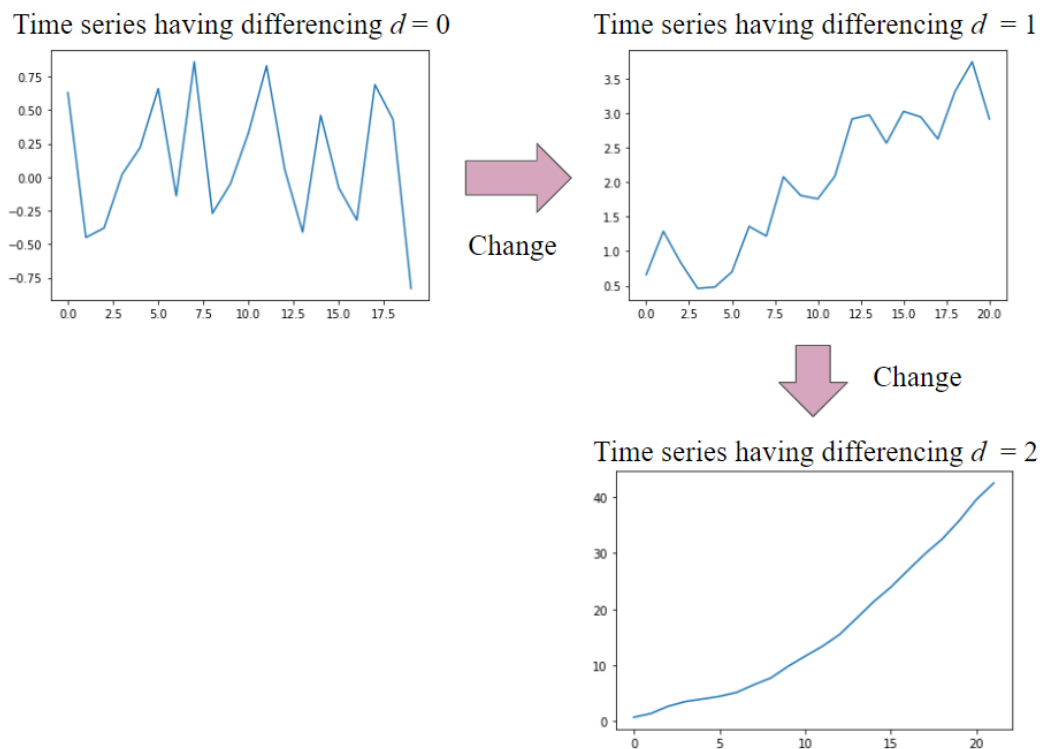


Figure 4.4: The example of time series having differencing $d = 0, 1, 2$.

For this case, the process is repeated two times to simulate time series data having the differencing order d from 0 to 2. Hence, the number of the time series data in the training data, the validating data and the testing data are 21600, 10800 and 2160, respectively. The processes are shown in Figure 4.5.

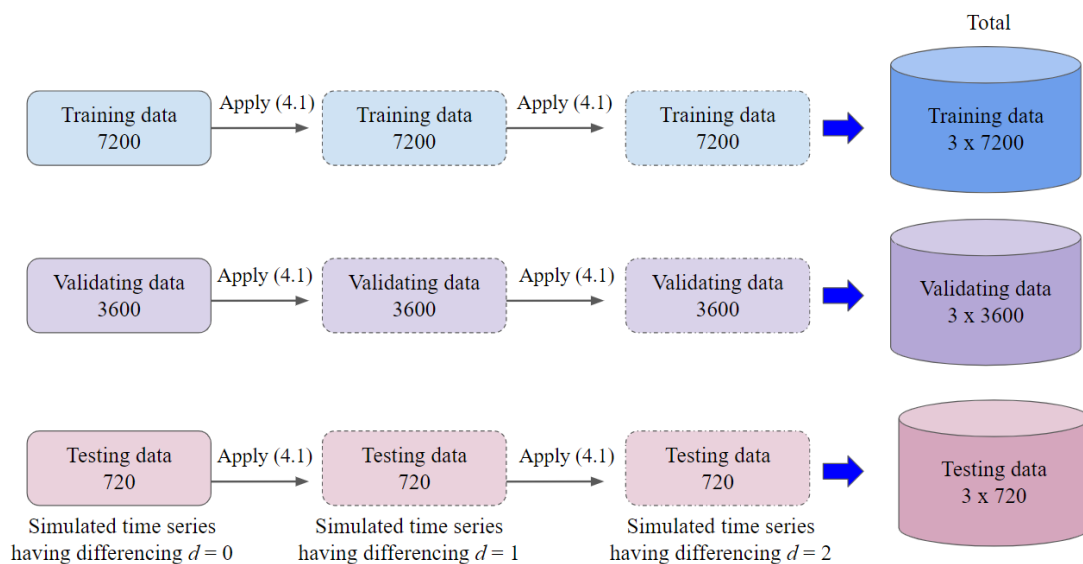


Figure 4.5: The processes of simulated time series data of the differencing orders.

After the simulating time series data is generated, ACF and PACF are generated and are converted to the ACF image, the PACF image and the time series image which are black and white. The generating processes of ACF and PACF images are described in Figure 4.6 and Algorithm 1. For the ACF images, the process begins to calculate the ACF values of the time series data. Then, the ACF values are put into two sets which are P (the set of positive values) and N (the set of negative values). Next, P is used to build the upper image of the ACF plot by marking the point according to the size of the ACF value in P. For N, it is used to build the lower image. For the PACF images can be done similarly as the ACF plot. For converting the time series data to the time series image, the process is described in Figure 4.7 and Algorithm 2. The time series data is transformed into the two-dimensional image according to the size of the image as shown in Figure 4.7. For this example, the size of the image is set to be 10×10 . Then, the two-dimensional image of the time series data is normalized to be in the range 0 to 1.

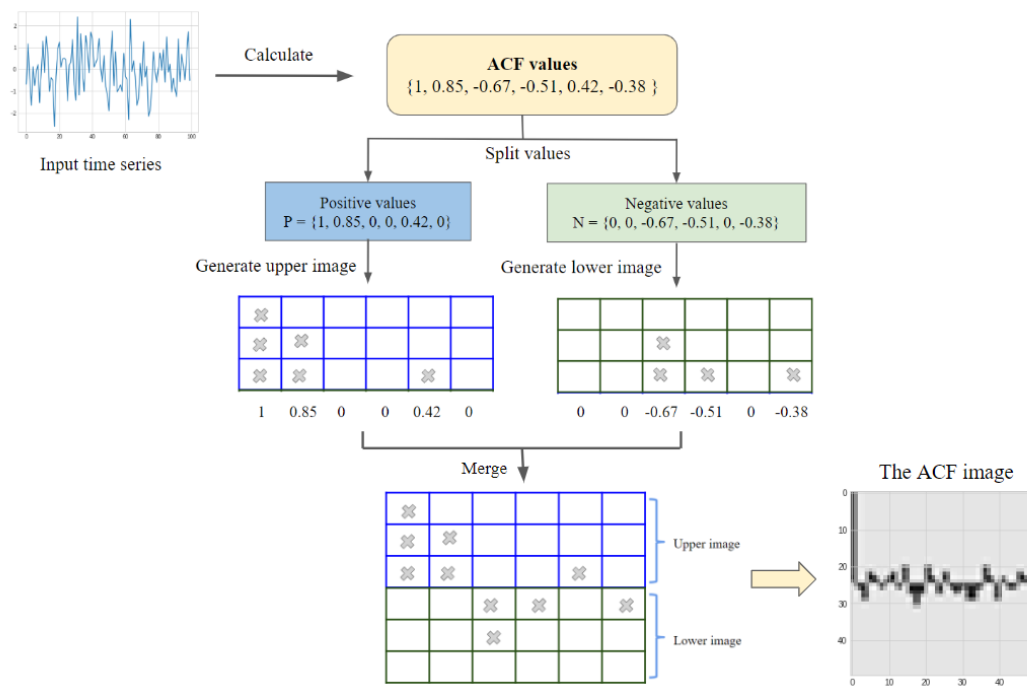


Figure 4.6: The example of generating the ACF image.

Algorithm 1 : Pseudo code for generating images of ACF/PACF

Input: ACF/PACF, k = the maximum lags of ACF/PACF

- 1: Define the image size as $k \times k$
 - 2: Split ACF/PACF values to P if the ACF/PACF value is non-negative and to N otherwise.
 - 3: Generate the upper image(positive part)
 - 4: **for** lag $i=1$ to k **do**
 - 5: Plot points with height of P[i]
 - 6: Generate the lower image(negative part) similar to step 3-5 using N instead of P
 - 7: Merge the upper image and the lower image together
-

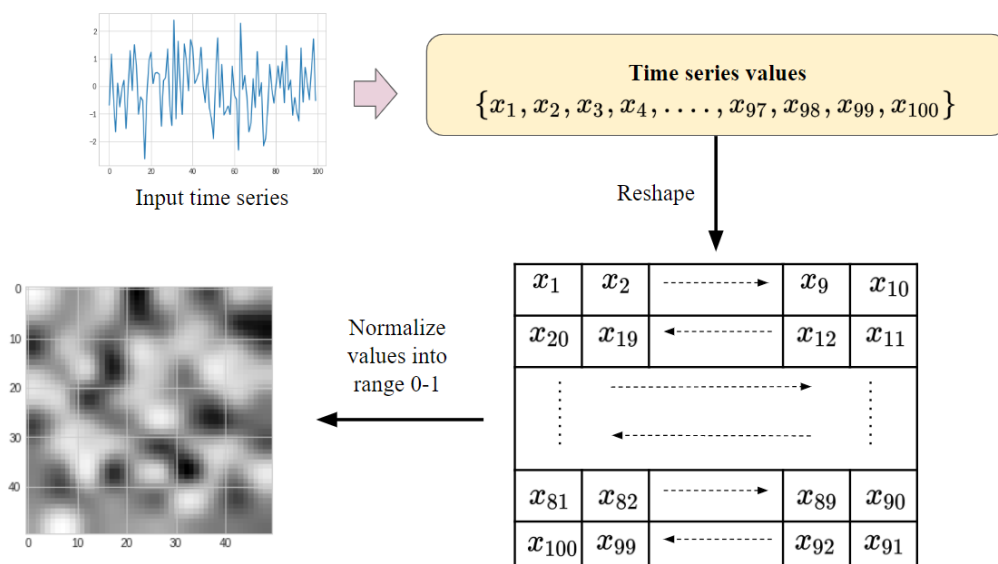


Figure 4.7: The example of generating the series image.

Algorithm 2 : Pseudo code for generating time series images

Input: time series data, image size = k

- 1: Define the image size as $k \times k$
 - 2: Generate the image
 - 3: Reshape 1 dimension of the time series having length of $k^2 \times 1$ to 2 dimension image having size of $k \times k$
 - 4: Normalize the image into range 0-1
-

4.1.2 Experimental results of the SID model

This section shows the experimental results of identifying the ARIMA order via the SID model which are divided into three cases: (1) the cases of identifying the AR order from 0-5, (2) the cases of identifying the MA order from 0-5 and (3) the case of identifying the differencing order from 0-2. The settings of each model having different channels are demonstrated in Table 4.1.

	Model	Case: identify p Channel(s)	Case: identify q Channel(s)	Case: identify d Channel(s)
Model S1	SID	1 channel: PACF images	1 channel: ACF images	3 channels: ACF images with $d = 0, 1$ and 2
Model S2	SID	2 channels: PACF and ACF images	2 channels: PACF and ACF images	None
Model S3	SID	3 channels: PACF, ACF and time series images	3 channels: PACF, ACF and time series images	None
Model R	ResNet50	Series	Series	None
Model L	auto-ARIMA (AIC criteria)	Series	Series	Series

Table 4.1: Description of each channel in the models for identifying the ARIMA order

The different input images come from their uses for identifying the ARMA order in the time series analysis. S1 uses the PACF images because PACF is used to identify the p order in the ARIMA model. In addition, S2 also uses the ACF image as the input since it also identifies the q order with the different characteristics. Moreover, S3 uses the time series images as inputs to add more information to the deep learning model. Two candidate models which are the ResNet50 model and the auto-ARIMA model are referred using their abbreviations as shown in Table 4.1. The SID model is called S1, S2 and S3 according to the inputs. R and L are represented as the ResNet50 model and the auto-ARIMA model.

Results of identifying the p order of the SID model

Figure 4.8 summarizes the scores of all models corresponding to the precision, the recall and the f1-score, respectively. From the results, S1, S2 and S3 are the best scores for all measurements having scores close to 1. The performance of S1, S2 and S3 is quite similar but S1 is the best. Both R and L give inferior performance.

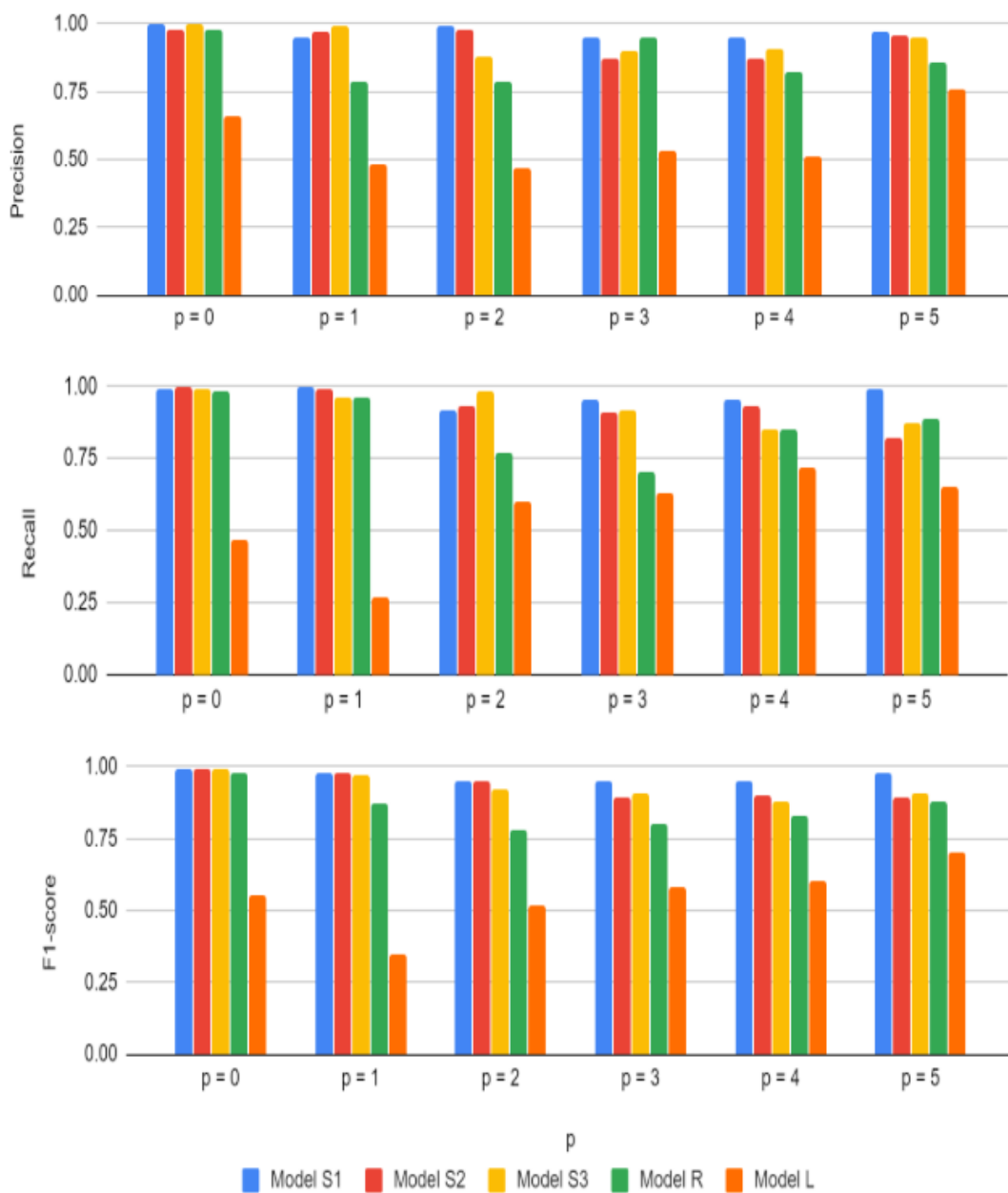


Figure 4.8: The precisions, the recalls and the f1-scores of the models identifying p

Results of identifying the q order of the SID model

From Figure 4.9, S2 and S3 give the best the precision, the recall and the f1-score. All scores of S2 are highest with respect to other models. R has trouble predicting q with the low values of the precision, the recall, the f1-score. L is better than R yet it is inferior than S1, S2, S3 for all measures.

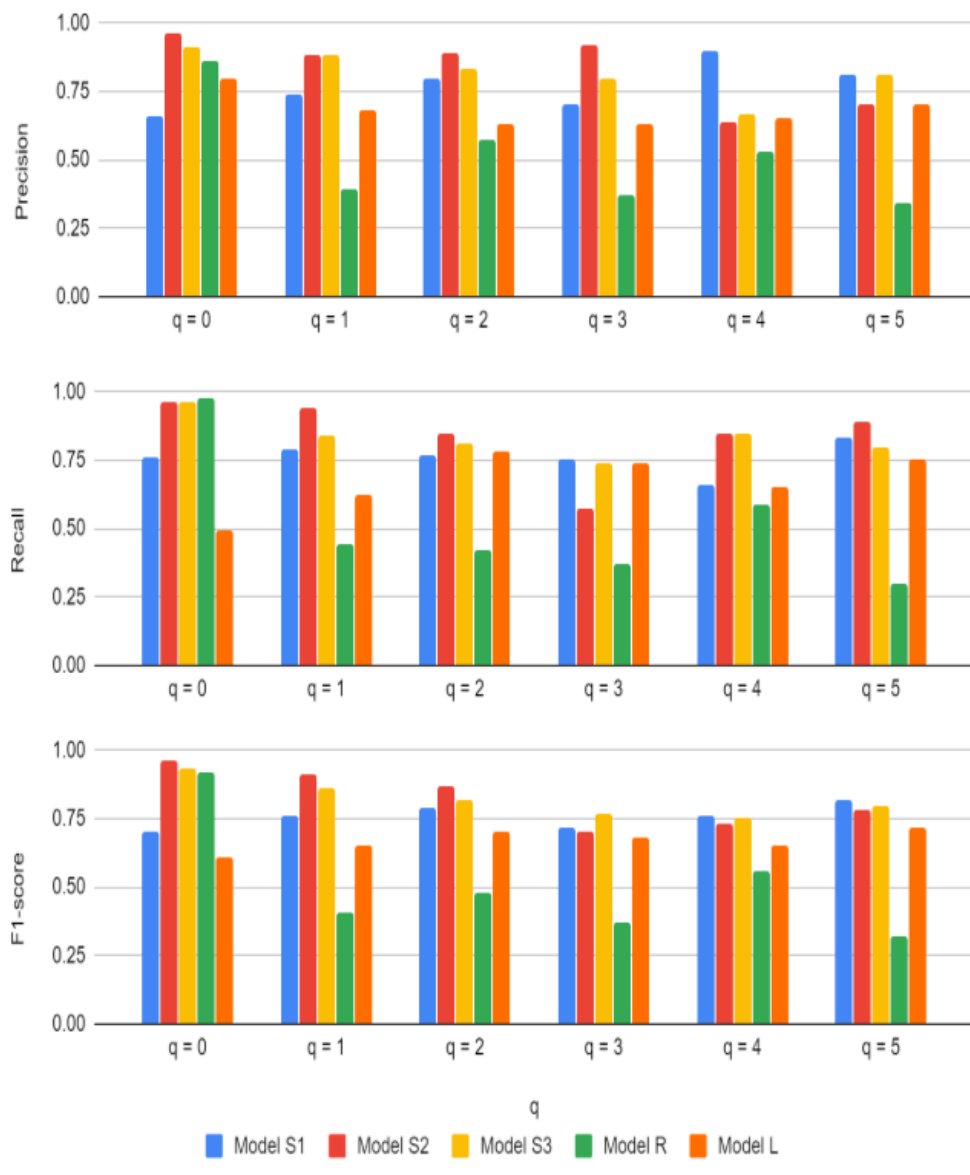


Figure 4.9: The precisions, the recalls and the f1-scores of the models identifying q

Results of identifying the d order of the SID model

Figure 4.10 shows the precision, the recall and the f1-score between S1 and L which it is clear that S1 gives the best values of the precision, the recall and the f1-score.

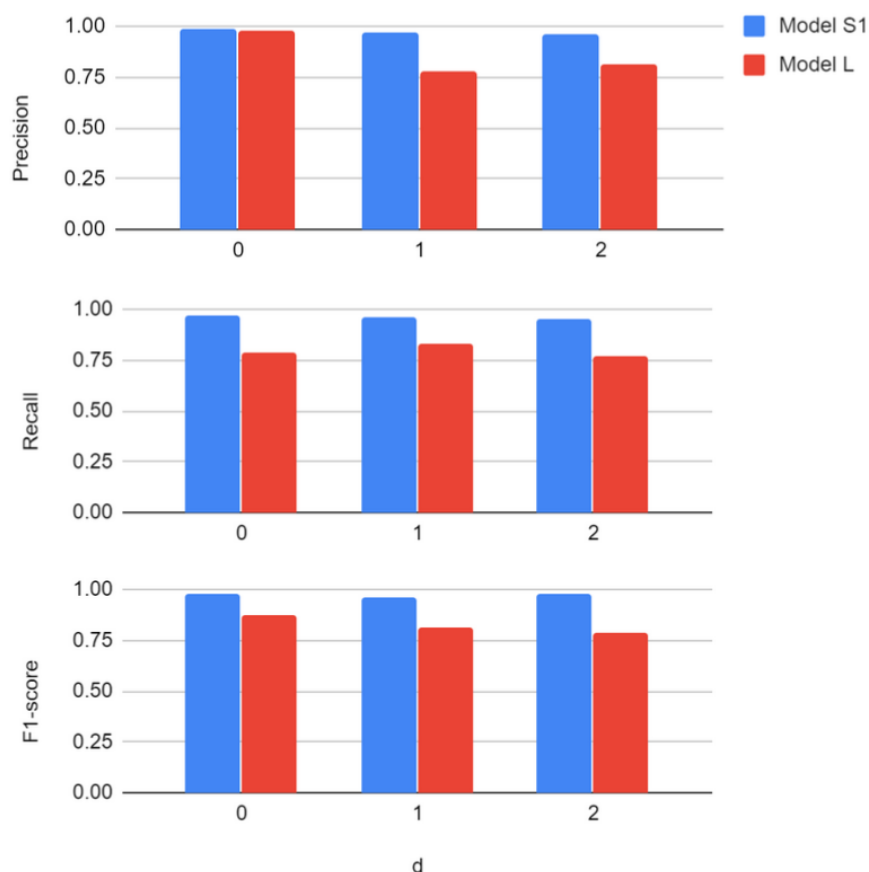


Figure 4.10: The precisions, the recalls and the f1-scores of the models identifying d

4.1.3 The conclusion of the SID model

For the conclusion of the SID model, the SID is constructed by training the PACF images and the ACF images as the inputs to identify the ARIMA order. The SID model is separated into the pq -SID model to identify the ARMA order and the d -SID model to identify the differencing order. The results of the SID model demonstrates the suitable ARIMA order measuring via the scores of the precision, the recall and the f1-score whereas the auto-ARIMA model fails to give the suitable ARIMA order. The outcome from this experiments confirms that using of ACF and PACF via visualizing as images is still more appropriate for model identification than the auto-ARIMA model and the ResNet50 model using the time series as inputs directly. Next, to improve the performance of the SID model, the SID model is adapted to the very deep network based on the architecture of ResNet. The details are shown in the next section.

4.2 The self-identification ResNet-ARIMA model

This section demonstrates the the process of constructing the SIRO model which is extended from the SID model using the architecture of ResNet to identify the ARIMA order and its experimental results. The details are shown as follows.

4.2.1 The architecture the SIRO model

In the deep learning architecture, ResNet is one of the popular deep learning architecture. It is built to help learning of very deep network which has the problem of updated weights. For a deep learning model with a long chains of the convolutional layers, it has the vanishing gradient problem where the gradient in the backpropagation process for updating the weights is getting smaller and closer to 0 inside the network. To fix this problem, the ResNet architecture is introduced using the concept of skipping the CNN layers. This section explains the architecture of the SIRO model based on the ResNet architecture. The model is split into two parts including the pq -SIRO model using for identifying the ARMA order from 0-5 and the d -SIRO model using for identifying the differencing order from 0-2.

The pq -SIRO model for identifying the ARMA order is constructed based on the ResNet architecture. The inputs of the pq -SIRO model comprise of five channels of 50×50 black and white images. The first channel is an image of the ACF plot, the second channel is an image of the PACF plot and the rest are images of the time series image taking differencing from $d = 0$ to 2. The pq -SIRO model consists of 14 convolutional layers applying the concept of skip connections over two adjacent convolutional layers. The first six convolutional layers have 64 of 3×3 filter matrices with the stride = 1 and the next eight convolutional layers have 128 of 3×3 filter matrices with the stride = 1. The next layer is the max-pooling layer of the filter of size 2×2 and the stride is set to 2. It is followed by a fully connected layer having 512 nodes, respectively. Finally, the final layer is softmax applied with all possible orders p from 0-5 and q from 0-5.

For the case of identifying the differencing order, the deep learning architecture of

the d -SIRO model is similar to the case of identifying the ARMA order but the inputs are different. The d -SIRO model of this case consists of 3 channels of the ACF plots from time series taking differencing to time series from $d = 0$ to 2 or the time images from time series taking differencing to time series from $d = 0$ to 2. For the final layer, softmax is applied with possible d from 0-2. The architectures of the pq -SIRO model and the d -SIRO model are shown in Figure 4.11 and Figure 4.12.



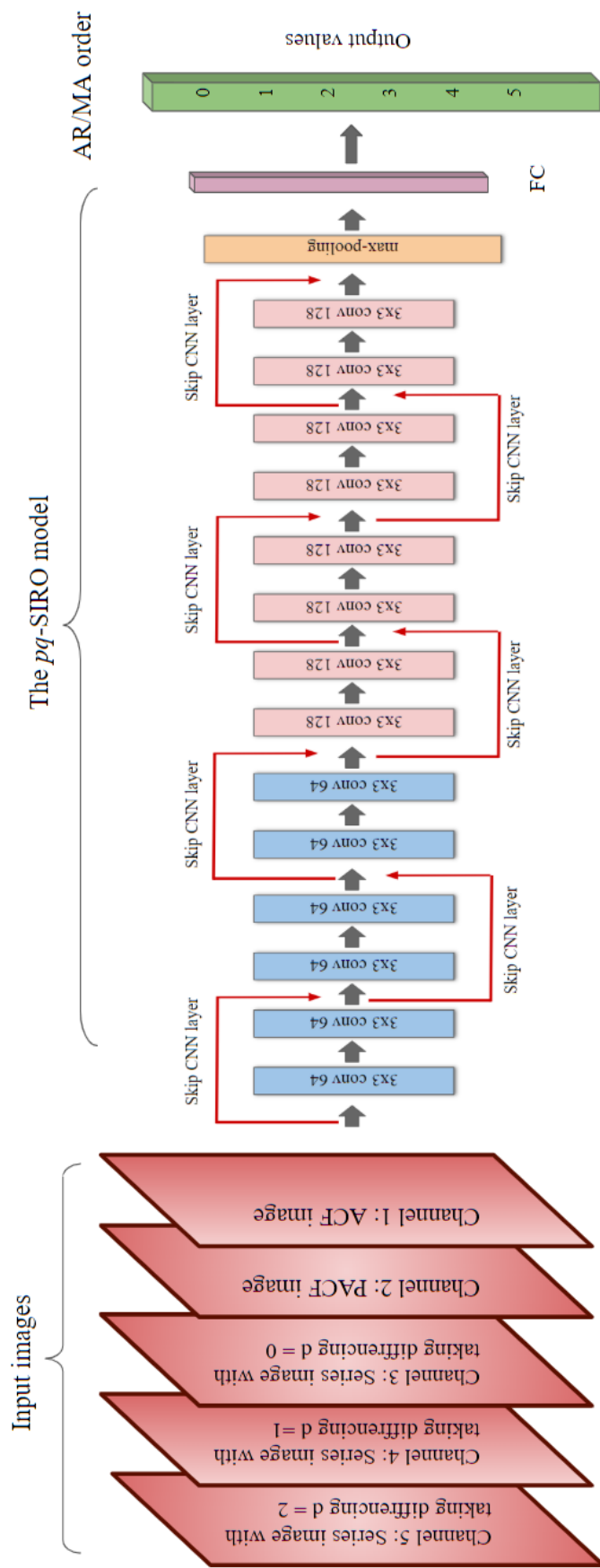


Figure 4.11: The architecture of the pq -SIRO model for identifying the AR and MA order

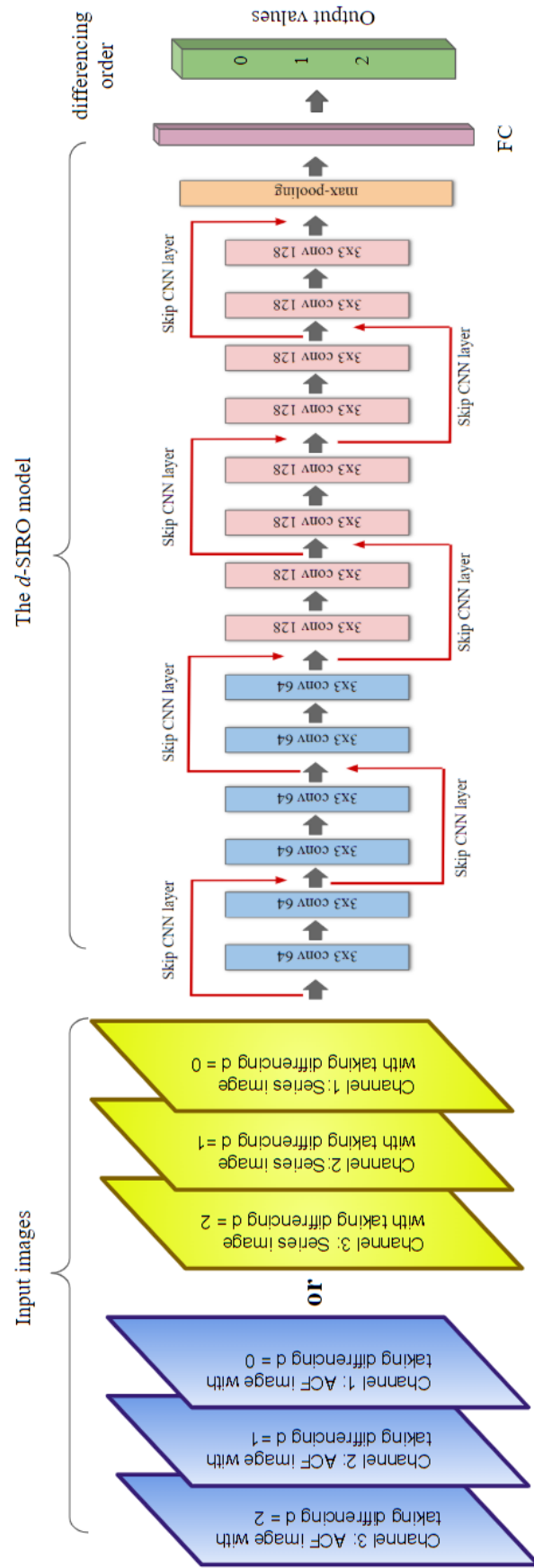


Figure 4.12: The architecture of the d -SIRO model for identifying the differencing order

In this research, the simulated time series data are generated from the ARMA process as in Section 4.1 by varying p from 0 to 5 and q from 0 to 5. All coefficients of the process must be randomly generated to satisfy the stationary property and the invertibility property from the uniform distribution within -1 to 1. The number of the time series data in the training data, the validating data and the testing data are 7200, 3600 and 720, respectively.

In the process of simulated time series for the differencing order, the simulated time series data from the ARMA process is accumulated by varying d from 0 to 2, respectively. The number of the time series data in the training data, the validating data and the testing data are 21600, 10800 and 2160. The processes of simulating time series and generating images are the same as Section 4.1.1.

4.2.2 Experimental results of the SIRO model

This section is to explain the experimental results of identifying the ARIMA order via the SIRO model which are divided into three cases including the cases of identifying the AR order from 0-5, the cases of identifying the MA order from 0-5 and the case of identifying the differencing order from 0-2. The settings of each model having different channels are demonstrated in Table 4.2.

	Model	Case: identify p Channel(s)	Case: identify q Channel(s)	Case: identify d Channel(s)
Model S1	SIRO	1 channel: PACF images	1 channel: ACF images	3 channels: ACF images with $d = 0, 1$ and 2
Model S2	SIRO	2 channels: PACF and ACF images	2 channels: PACF and ACF images	3 channels: time series images with $d = 0, 1, 2$
Model S3	SIRO	3 channels: PACF, ACF and time series images with $d = 0$	3 channels: PACF, ACF and time series images with $d = 0$	None
Model S4	SIRO	4 channels: PACF, ACF and time series images with $d = 0, 1$	4 channels: PACF, ACF and time series images with $d = 0, 1$	None
Model S5	SIRO	5 channels: PACF, ACF and time series images with $d = 0, 1, 2$	5 channels: PACF, ACF and time series images with $d = 0, 1, 2$	None
Model R	ResNet50	Series	Series	None
Model L	auto-ARIMA (AIC criteria)	Series	Series	Series

Table 4.2: Description of each channel in the models for identifying the ARIMA order

To explain the experimental results, the SIRO model and the candidate models which are the ResNet50 model and the auto-ARIMA model are called via the abbreviations as shown in Table 4.2. The SIRO model is called via S1, S2, S3, S4 and S5 according to the setting inputs. R and L are represented to the ResNet50 model and the auto-ARIMA model.

Results of identifying the p order the SIRO model

The measures of all models including the precision, the recall and the f1-score are shown in Figure 4.13. From this experiment, the models having the top-five scores are S1, S2, S3, S4 and S5. All scores are close to 1. Although the results of S1, S2, S3, S4 and

S5 provide similar scores of the precision, the recall and the f1-score, S1 gives the best performance. For the worst case, L provides the scores around 0.5 to 0.6 in the precision, the recall and the f1-score. For the case of model R, it is quite better than L, but this model gives lower scores than S1, S2, S3, S4 and S5 when identifying higher order.

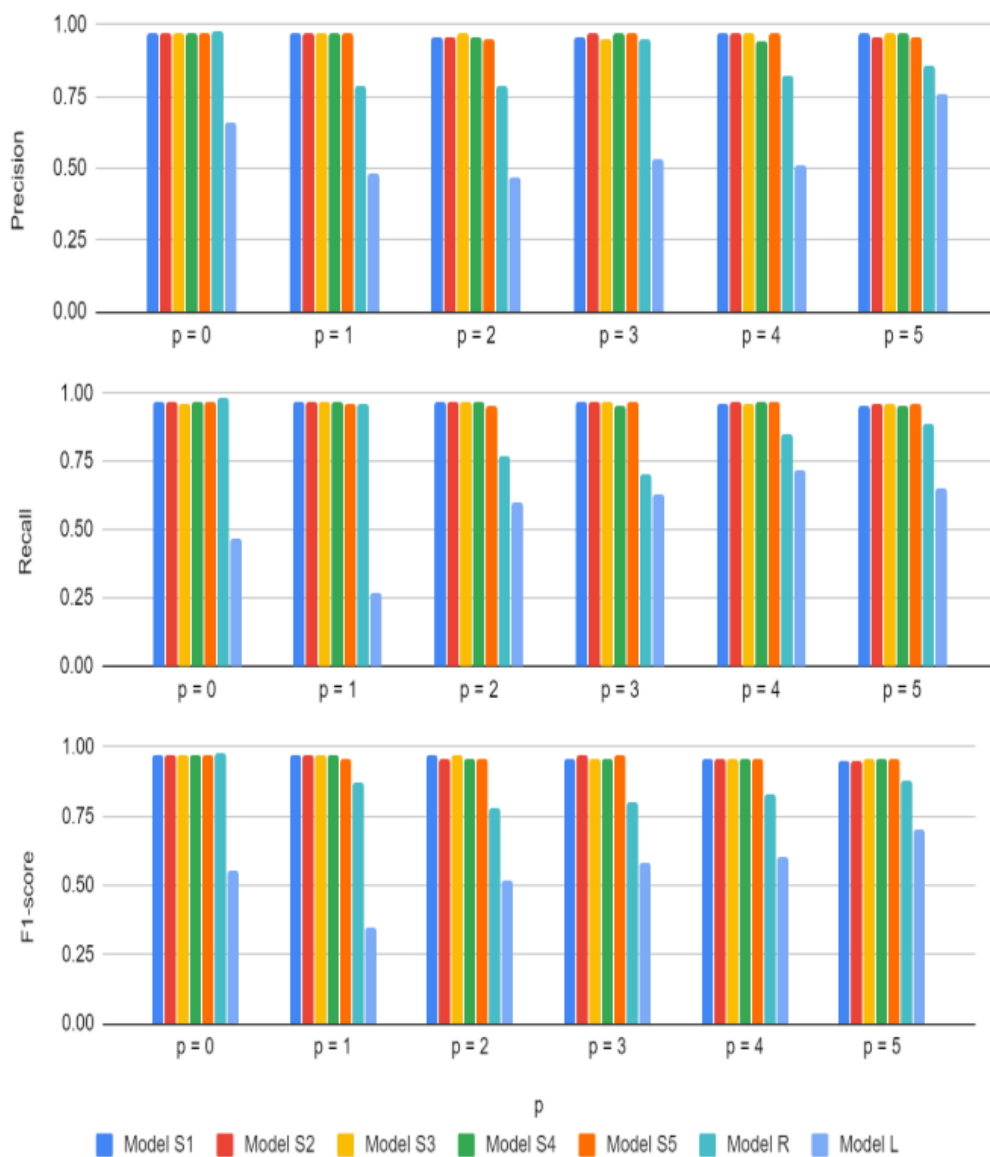


Figure 4.13: The precisions, the recalls and the f1-scores of the models identifying p

Results of identifying the q order of the SIRO model

For the case of identifying the q order, the scores of the precision, the recall and the f1-score are demonstrated in Figure 4.14. The results show that model S1, S2 and S3 give

the best scores of the precision, the recall and the f1-score. For the result of model R, all scores have trouble predicting indicated by their scores close to 0.5 in the precision, the recall and the f1-score. For the auto-ARIMA model case, L gives the better performance than model R, however it is still worse than S1, S2, S3, S4 and S5.

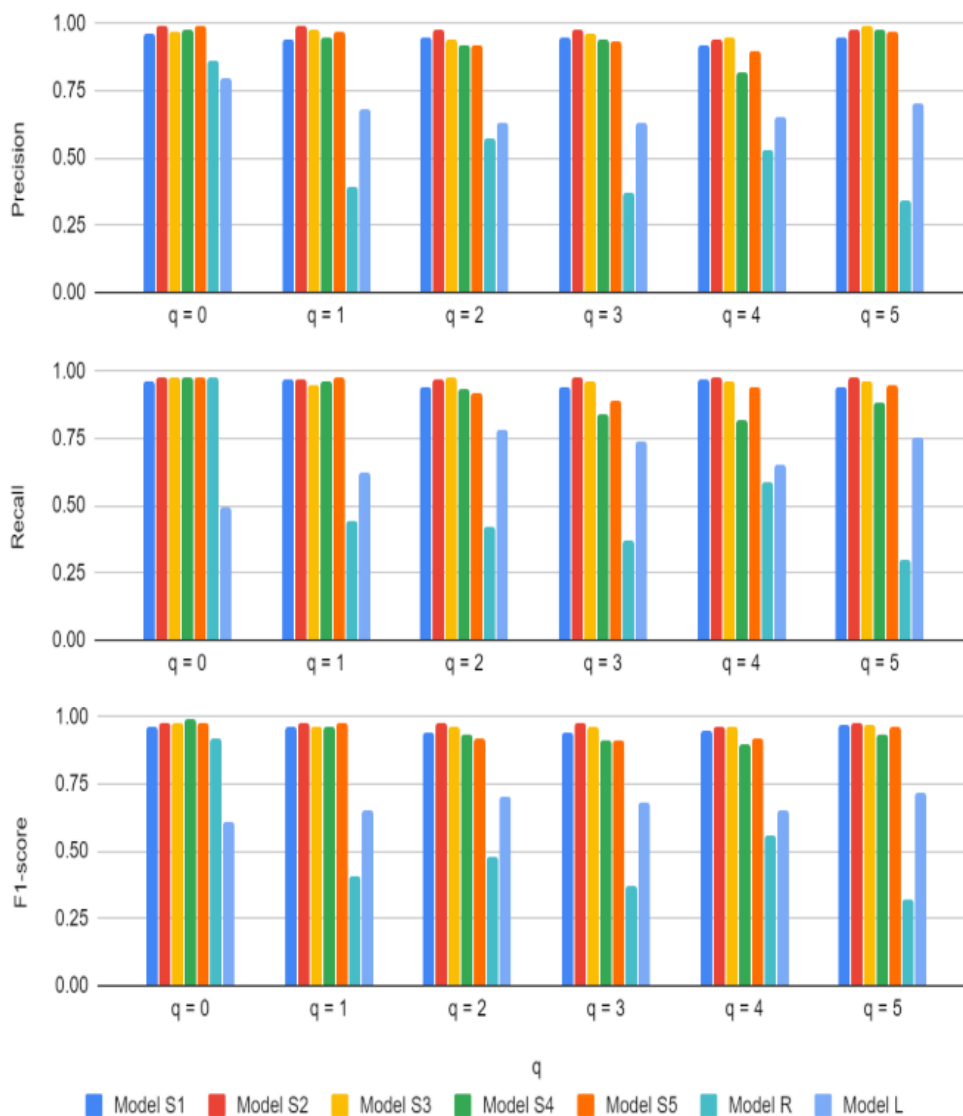


Figure 4.14: The precisions, the recalls and the f1-scores of the models identifying q

Results of identifying the d order of the SIRO model

For the case of identifying the differencing order in Figure 4.15, it is obvious that S1 provides the best scores of the precision, the recall and the f1-score in all cases. In the

auto-ARIMA model case, L has the best the precision when $d = 0$ whereas the recall and the f1-score of L is worse than the recall and the f1-score of S1 and S2.

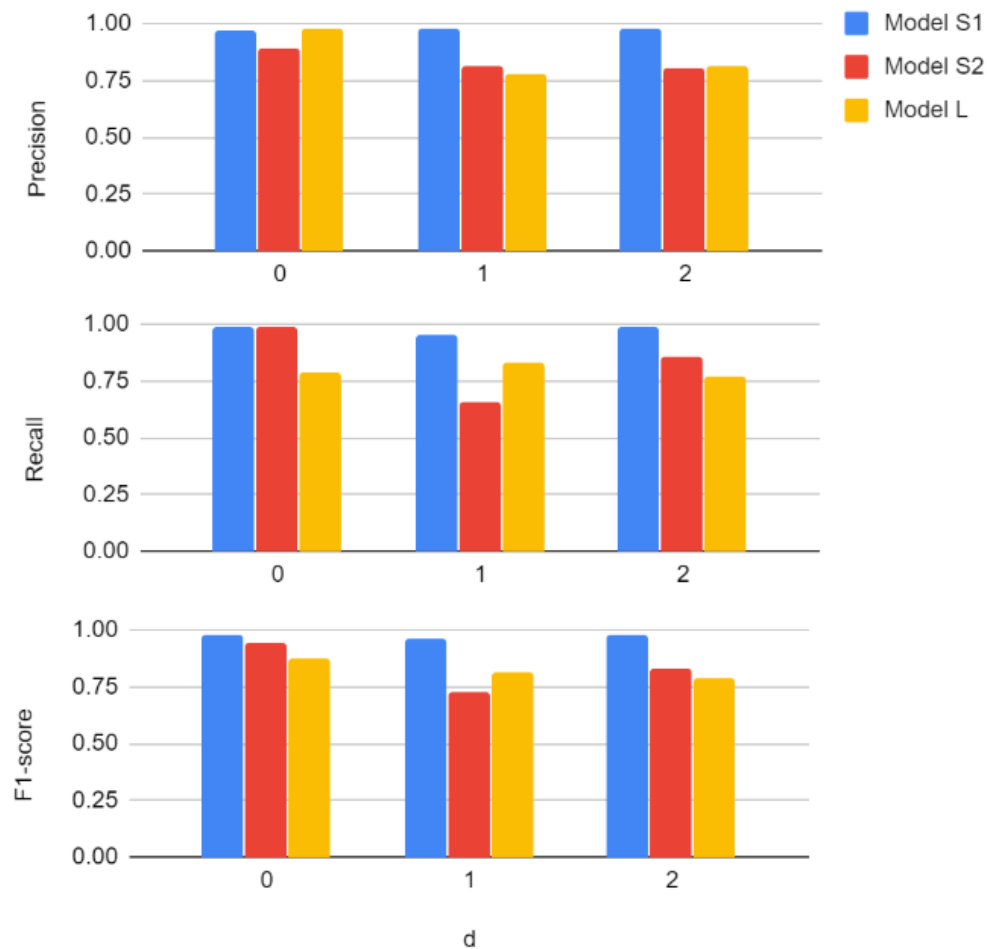


Figure 4.15: The precisions, the recalls and the f1-scores of the models identifying d

4.2.3 The conclusion of the SIRO model

For the conclusion of the SIRO model, the SIRO is constructed based on the ResNet architecture by training the PACF images and the ACF images as the inputs to identify the ARIMA order like the SID model. The SIRO model is separated into the pq -SIRO model to identify the ARMA order and the d -SIRO model to identify the differencing order. In the experimental results of the SIRO model, it outperforms the auto-ARIMA model and the previous deep learning model using ResNet50 in the scores of the precision, recall and f1-score. The results are suggested that the SIRO model is able to extract the

features of the ACF images, the PACF images and the series images which are tough for analysts to identify the suitable ARIMA order. Next, the new tool of the time series analysis is introduced for using to be the new input of the deep learning model. The details are shown in the next section.

4.3 The ACF-PACF-ESACF convolutional neural network ARIMA order identification model

This section demonstrates the process of constructing the APEA model using ESACF as inputs and its experimental results. The details are shown as follows.

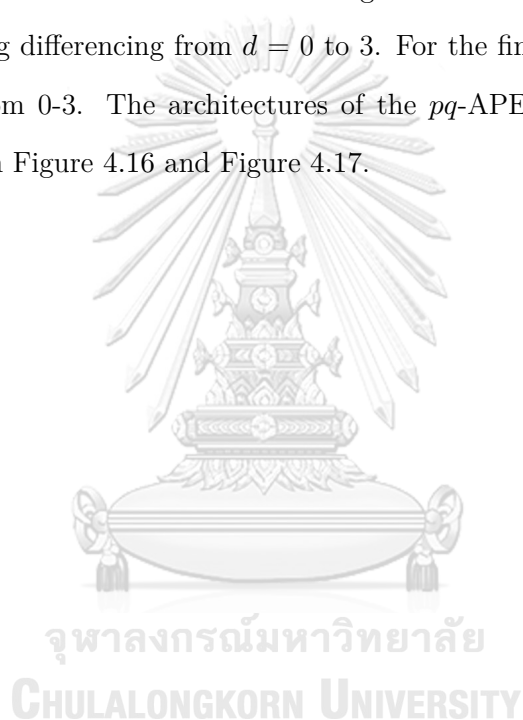
4.3.1 The architecture of the APEA model

In the time series analysis, the PACF plot using for identifying the AR order and the ACF plot using for identifying the MA order are very useful to identify the ARMA order. Nevertheless, the PACF plot and the ACF plot obscure one another. Therefore, ESACF is introduced to identify the AR order and the MA order using a single plot. Consequently, this section explains the architecture of the APEA model using ESACF as inputs of the deep learning model. The model is split into two parts: (1) the pq -APEA model using for identifying the ARMA order from 0-7 and (2) the d -APEA model using for identifying the differencing order from 0-3.

In this research, the pq -APE model for identifying the ARMA order is built based on the CNN model. The inputs of the pq -APEA model consist of four channels of 50×50 black and white images. The first channel is an image of the ACF plot, the second channel is an image of the PACF plot, the third channel is an image of the ESACF plot and the last channel is the time series image from differencing. the pq -APE model comprises of 2 convolutional layers. The first convolutional layers have 64 of 3×3 filter matrices with the stride = 1 and the next convolutional layers have 128 of 3×3 filter matrices. Each of the convolutional layer is connected with the max-pooling layer of the filter of size 2×2 and the stride is set to 2. After the CNN model, it is followed by a fully connected layers having 512 nodes. Finally, in the case of identifying the ARMA order, the softmax layer

is applied with all possible of orders p and q , which are 64 classes varying p from 0-7 and q from 0-7.

For the case of identifying the differencing order, the deep learning architecture of the d -APEA model is similar to the pq -APEA model. The inputs of the d -APEA model consists of 4 channels of the ACF plots from time series images taking differencing from $d = 0$ to 3. Moreover, to increase the efficiency of the deep learning model to identify the differencing order, PACF is introduced as the input data to add more information to the deep learning model for decision the differencing order. The PACF plots are from time series images taking differencing from $d = 0$ to 3. For the final layer, softmax is applied with possible d from 0-3. The architectures of the pq -APEA model and the d -APEA model are shown in Figure 4.16 and Figure 4.17.



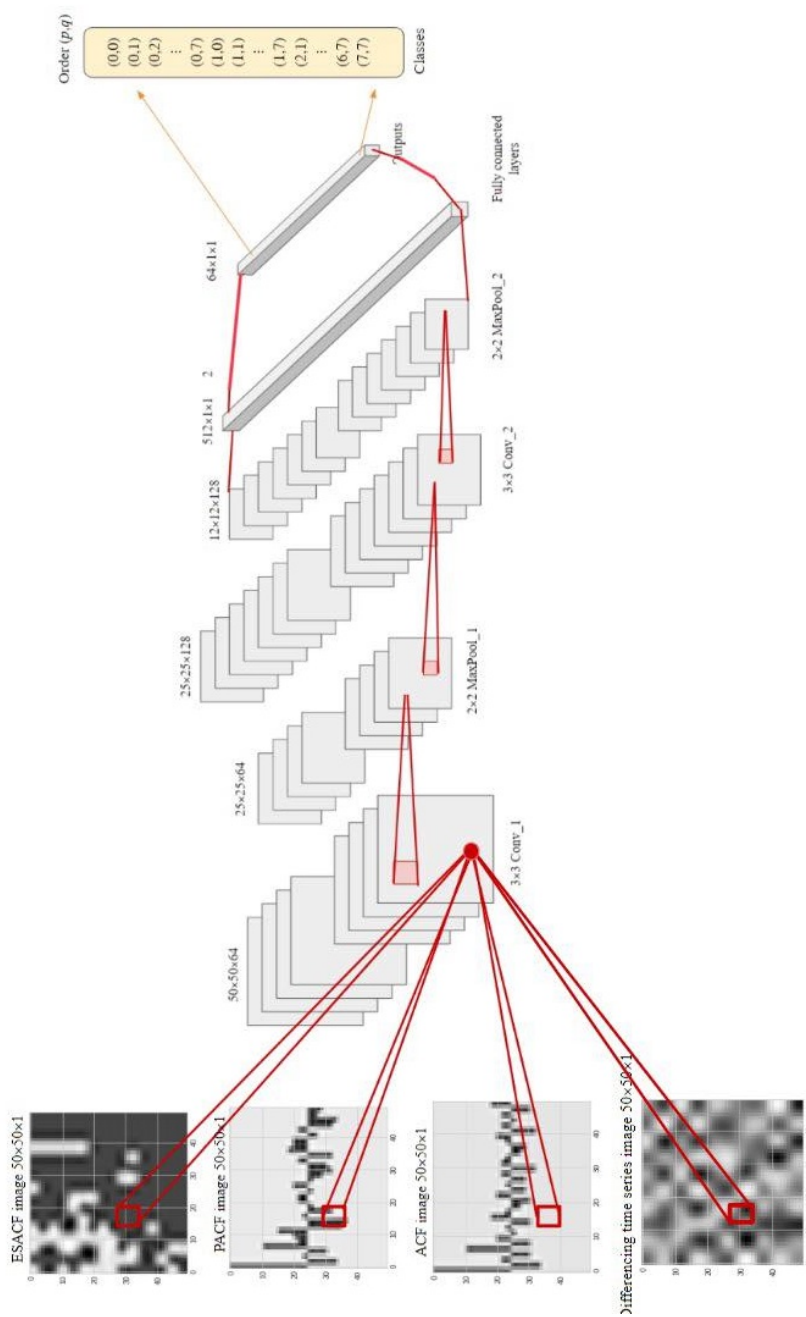


Figure 4.16: pq -APEA model for identifying the AR and MA order

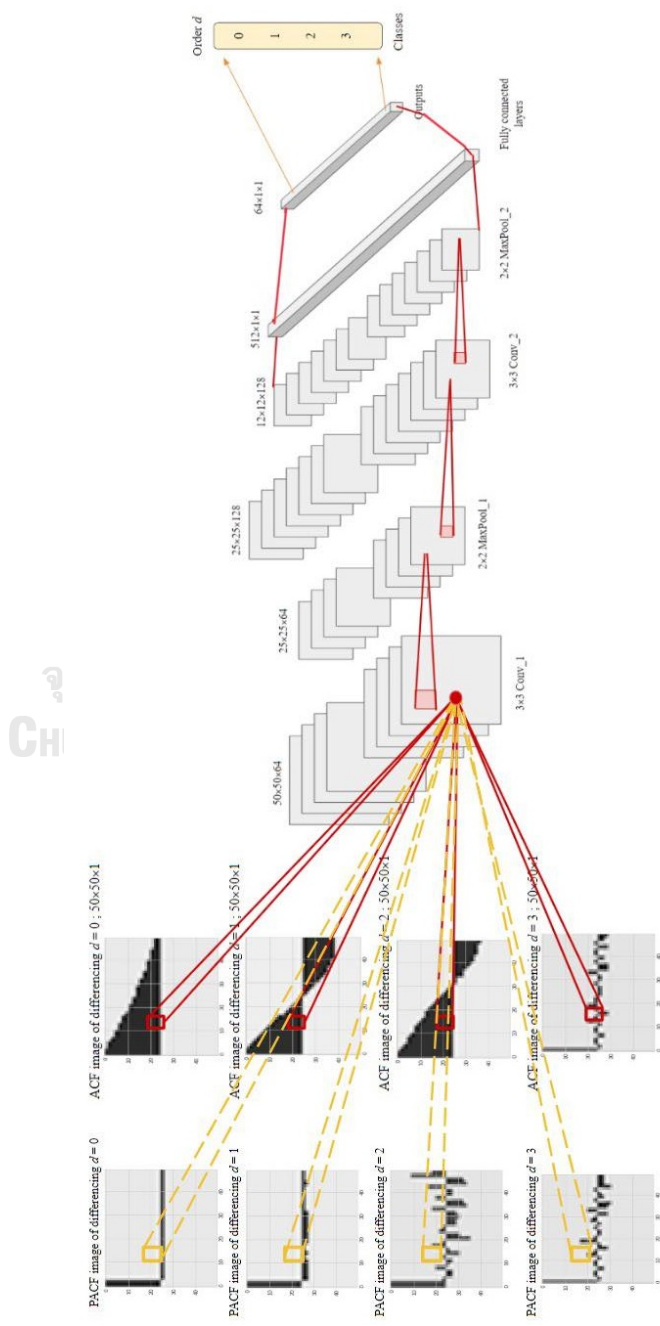
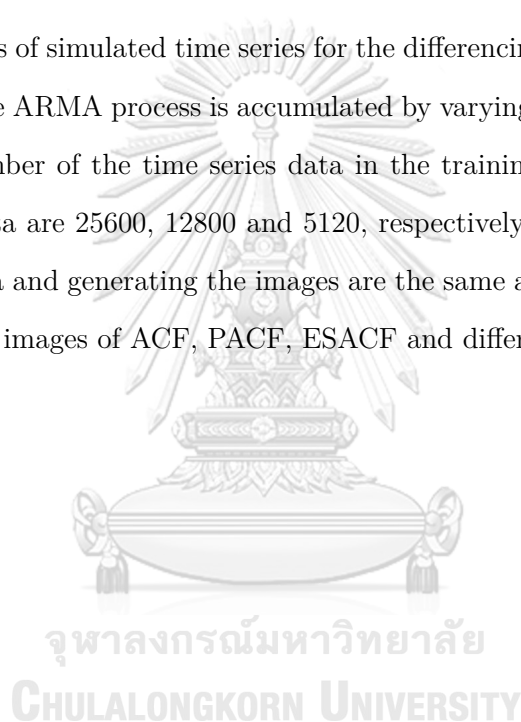


Figure 4.17: d -APEA model for identifying the differencing order

In this research, the simulated time series data are generated from the ARMA process varying p from 0 to 7 and q from 0 to 7. Moreover, all coefficients of the process must satisfy the stationary and invertibility properties. They are randomly generated from the uniform distribution within -1 to 1. If the stationary property or invertibility property is not satisfied, all coefficients will be regenerated. The number of the time series data in the training data repeating 100 times for each class, the validating data repeating 50 times for each class and the testing data repeating 20 times for each class are 6400, 3200 and 1280, respectively.

In the process of simulated time series for the differencing order, the simulated time series data from the ARMA process is accumulated by varying d from 0 to 3, respectively. Therefore, the number of the time series data in the training data, the validating data and the testing data are 25600, 12800 and 5120, respectively. The process of simulating the time series data and generating the images are the same as Section 4.1.1. Figure 4.18 shows the example images of ACF, PACF, ESACF and differencing time series.



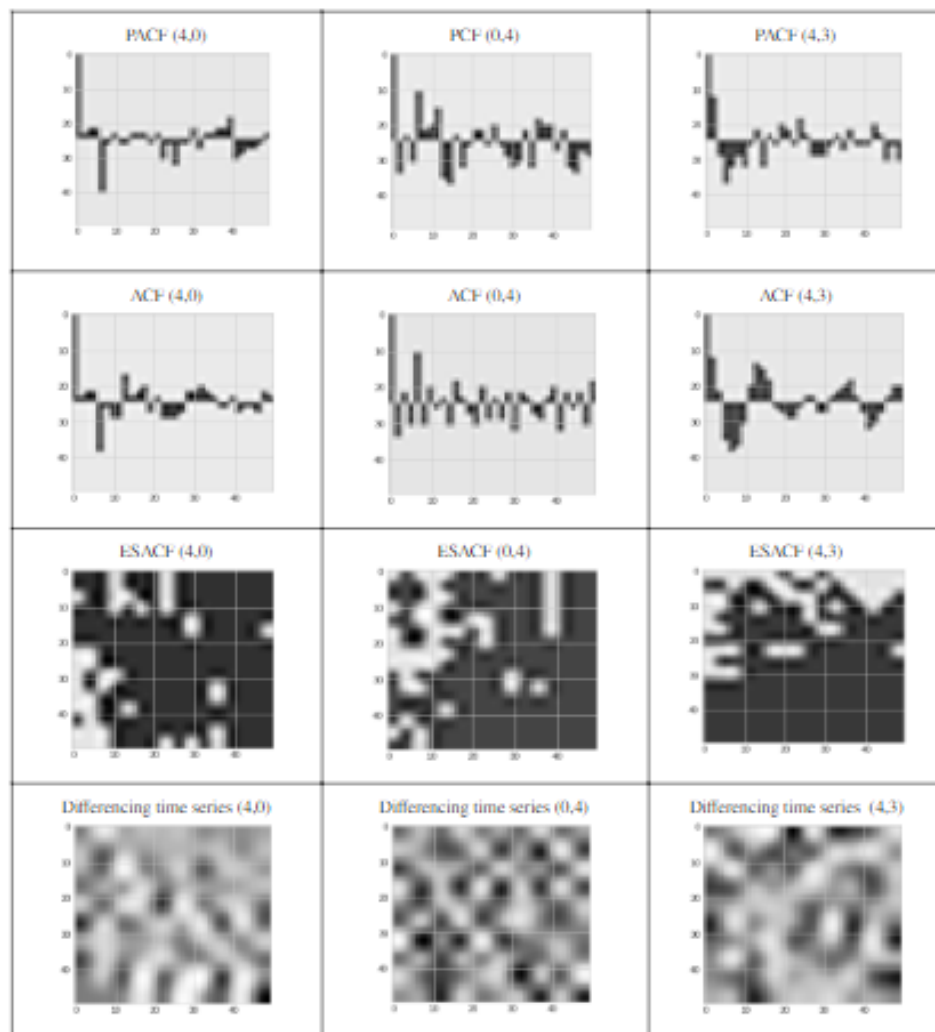


Figure 4.18: Samples of ACF, PACF, ESACF and differencing time series images

4.3.2 Experimental results of the APEA model

The experimental results of identifying the ARIMA order via the APEA model are divided into three cases including the cases of identifying the AR order from 0-7, the cases of identifying the MA order from 0-7 and the case of identifying the differencing order from 0-3. The settings of each model having different channels are demonstrated in Table 4.3.

	Model	Case: identifying order p and q Channel(s)	Case: identifying order d Channel(s)
Model A1	APEA	1 channel: ESACF images	4 channels: ACF images with $d = 0, 1, 2$ and 3
Model A2	APEA	2 channels: PACF and ACF images	4 channels: PACF images with $d = 0, 1, 2$ and 3
Model A3	APEA	2 channels: ESACF and differencing time series images	8 channels: PACF and ACF images with $d = 0, 1, 2$ and 3
Model A4	APEA	3 channels: ESACF, PACF and ACF images	None
Model A5	APEA	3 channels: PACF, ACF and differencing time series images	None
Model A6	APEA	4 channels: ESACF, PACF, ACF and differencing time series images	None
Model R	ResNet50	Series	None
Model L	auto-ARIMA (AIC criteria)	Series	Series

Table 4.3: Description of each input of each model for identifying the ARIMA order

To explain the experimental results, the APEA model and the candidate models which are the ResNet50 model and the auto-ARIMA model are called via the abbreviations as shown in Table 4.3. The APEA model is called via A1, A2, A3, A4, A5 and A6 according to the setting inputs. R and L are represented to the ResNet50 model and the auto-ARIMA model.

Results of identifying the p order of the APEA model

Figure 4.19 summarizes the precisions, the recalls and the f1-scores of all models. From this figure, A4 using 3 channels of ESACF, ACF and PACF and A6 using 4 channels of ESACF, ACF, PACF and differencing time series images are the top-two best scores having scores close to 0.8. The performance of A1, A2 and A3 are similarly good but they are still not the best using only ESACF or PACF and ACF. R and L show the poor performance.

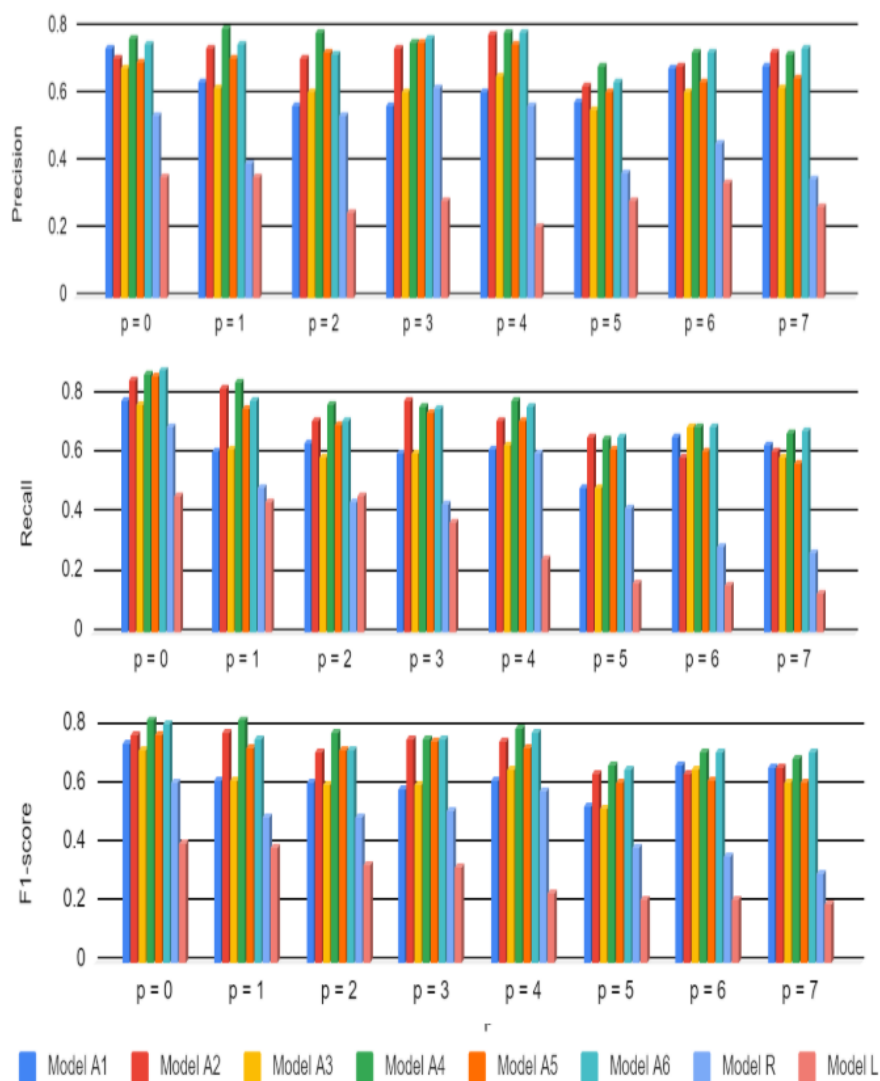


Figure 4.19: The precisions, the recalls and the f1-scores of the models identifying p

Results of identifying the q order of the APEA model

From Figure 4.20, A4 using 3 channels of ESACF, PACF and ACF shows the best performance of the precisions, the recalls and the f1-scores. Although A6 using 4 channels of ESACF, PACF, ACF and differencing time series images provides similar performance to A4, A4 performs better for the high order. For the other models, A1, A2, A3 and A5, the performance is quite similar which decrease when the order is increase. For R and L, the scores of both models are quite bad.

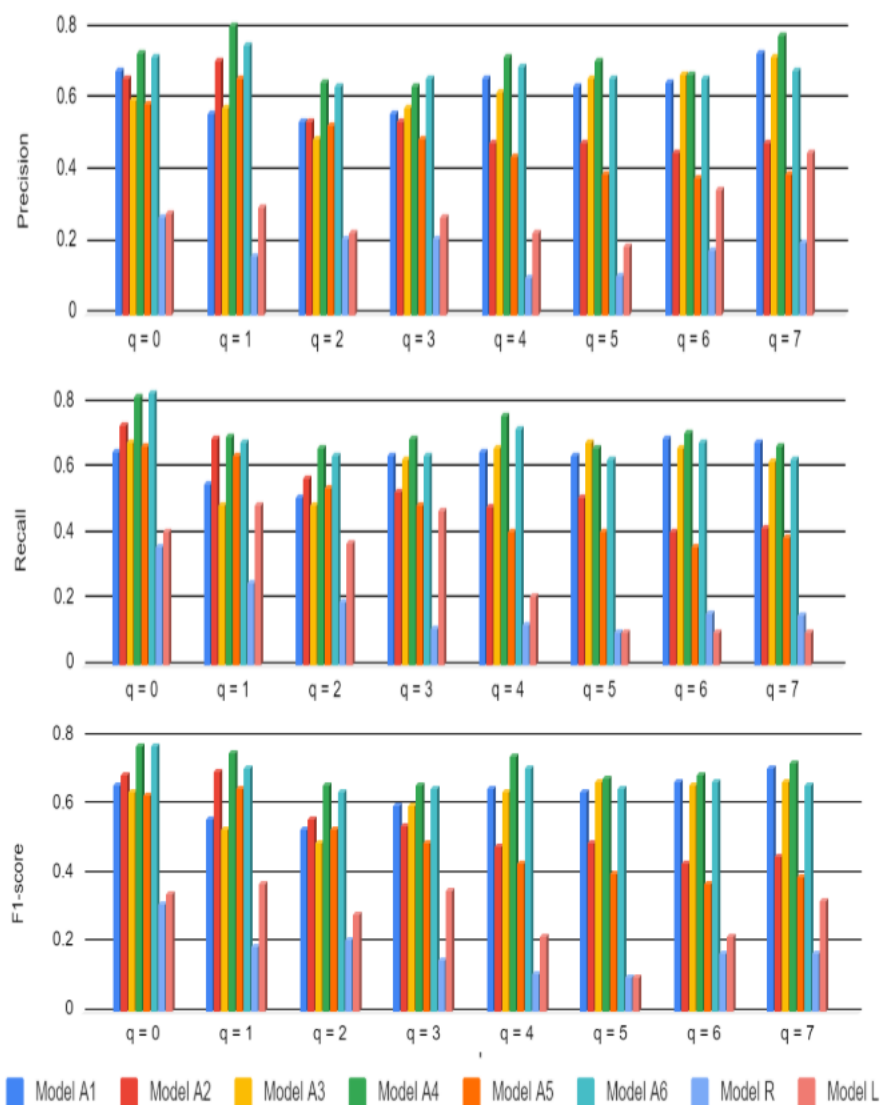


Figure 4.20: The precisions, the recalls and the f1-scores of the models identifying q

Results of identifying the d order of the APEA model

Figure 4.21 shows the scores of the precision, the recall and the f1-score between A1, A2, A3 and L. A1 and A2 using ACF and PACF as inputs respectively provide the good performance and are quite similar. When using ACF and PACF as inputs altogether in A3, the performance of the model is improved in terms of the precision and the recall. L gives the lowest performance.

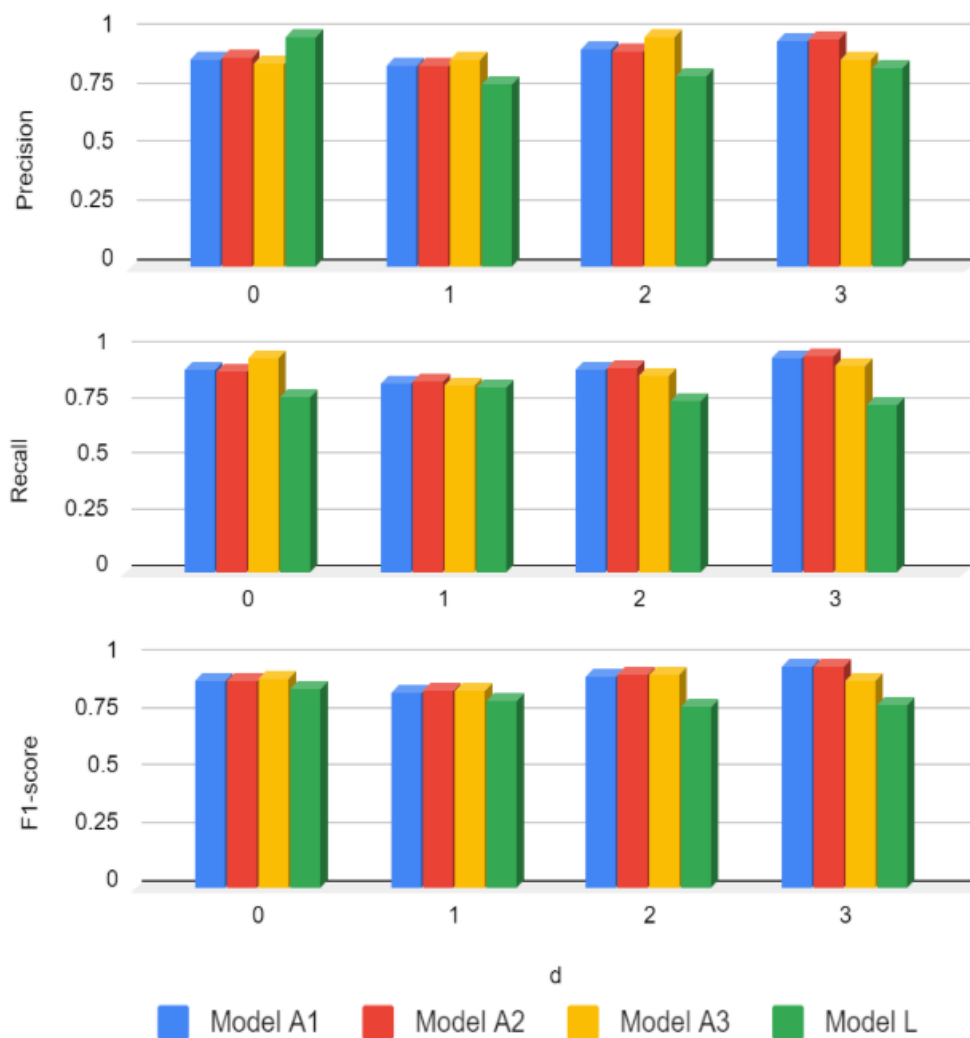


Figure 4.21: The precisions, the recalls and the f1-scores of the models identifying d

4.3.3 The conclusion of the APEA model

The APEA model is constructed based on the basic CNN model learning from the ARMA models having the ARMA orders from 0 to 7 and the differencing order from 0 to 3 via the ESACF images, the PACF images and the ACF images as the inputs. The APEA

model is separated into the pq -APEA model to identify the ARMA order and the d -APEA model to identify the differencing order. In the experimental results of the APEA model, the APEA model exhibits the best performance compared with the auto-ARIMA model and the previous deep learning model using ResNet50 via the precisions, the recalls and the f1-scores. The results are suggested that the APEA model is able to extract features from the ESACF images, the ACF images, the PACF images and the series images at the same time which it is very difficult for analysts. Next, the APEA model is extended to identify the SARIMA order for the time series data with the seasonal component. The details are shown in the next section.

4.4 Visualization of the convolutional neural network filter

This section demonstrates filter matrices from some time series data of the APEA model. All filter matrices are visualized via the color scale which is called “Viridis”. It can be used via the “matplotlib” package in python. The color scales are shown in Figure 4.22. The light yellow on the right side represents the maximum pixel value in the images and the dark purple on the left side represents the minimum pixel value in the image. The Viridis color scales is popularly used to visualize the images because it can clearly show the color difference in the image. These features of the deep learning model can be visualized in Figure 4.23 to Figure 4.26 which show the filter matrices of identifying the AR orders from 0 to 3 of the APEA model taking 4 channels as inputs consisting of the ESACF images, the PACF images, the ACF images and the differencing time series images.

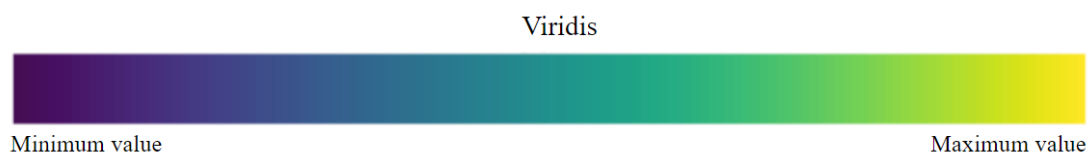
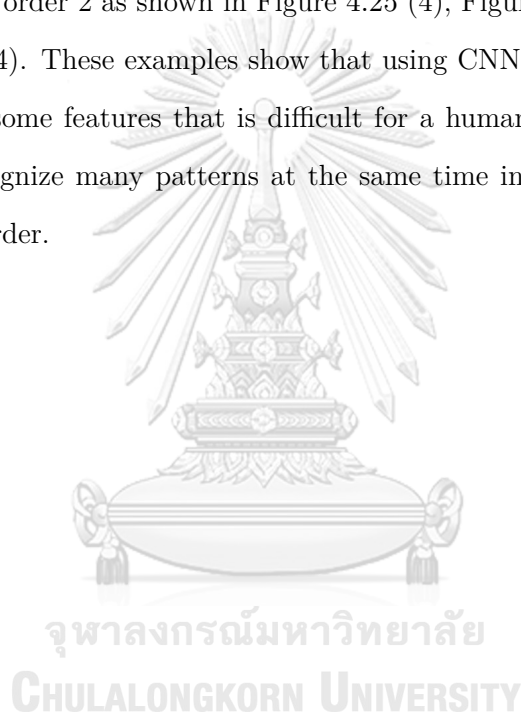


Figure 4.22: The color scales of “Viridis”.

From these figures, the APEA model attempts to combine all features of the ESACF

images, the PACF images, the ACF images and the differencing time series images together and extracts features of each channel by considering the core of the image as lags of PACF and ACF and the pattern of ESACF and the differencing time series. In Figure 4.23, the filter matrices try to capture the characteristic of the time series having the AR order equals to 0. They show the light yellow area around the axis as appeared in Figure 4.23 (4), Figure 4.23 (5), Figure 4.23 (14) and Figure 4.23 (16) whereas the other filter matrices in Figure 4.24 to Figure 4.26 try to capture the lags occurring with the corresponding AR order. For example, the filter matrices in Figure 4.25 capture at lag 2 to indicate the AR order 2 as shown in Figure 4.25 (4), Figure 4.25 (5), Figure 4.25 (13) and Figure 4.25 (14). These examples show that using CNN for identifying the ARIMA order can extract some features that is difficult for a human. Moreover, the process of CNN can also recognize many patterns at the same time in order to identify the most suitable ARIMA order.



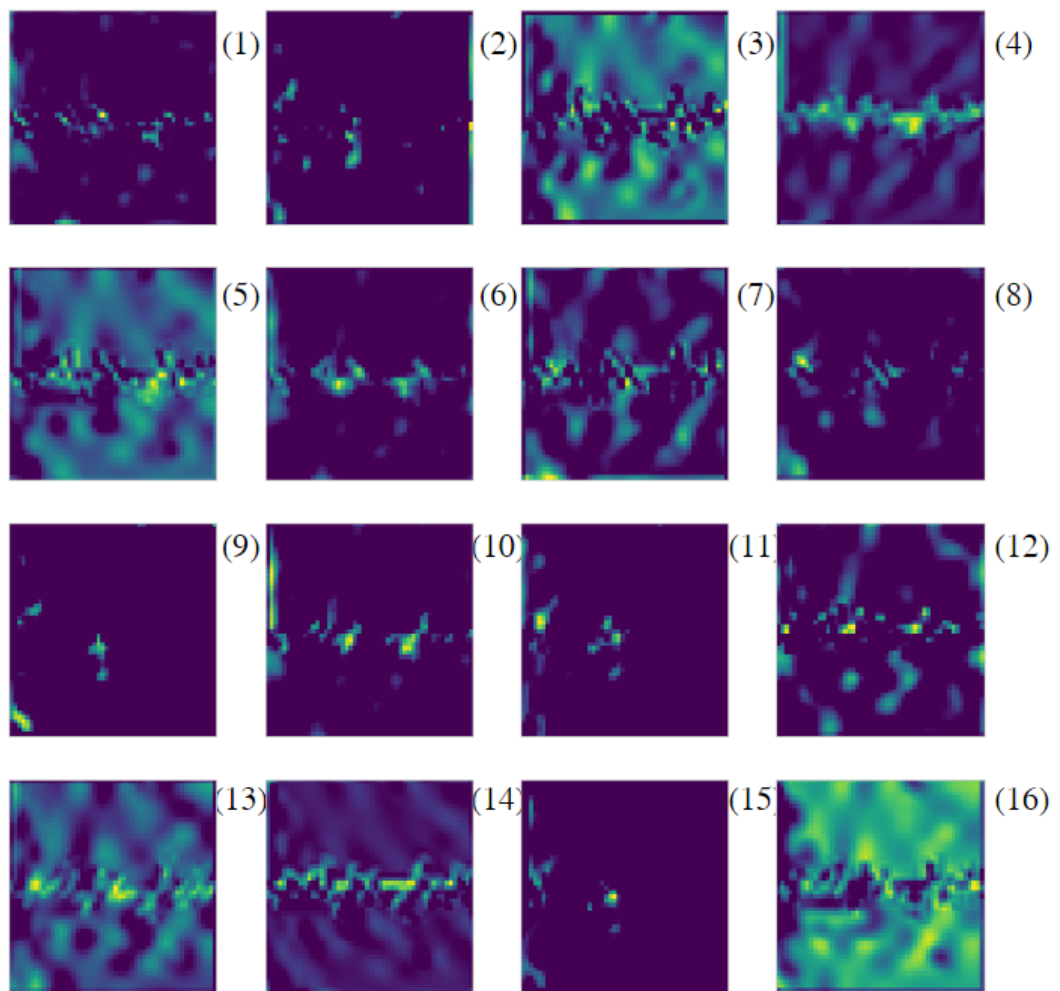


Figure 4.23: The filter matrices of the CNN layer which the AR order is 0

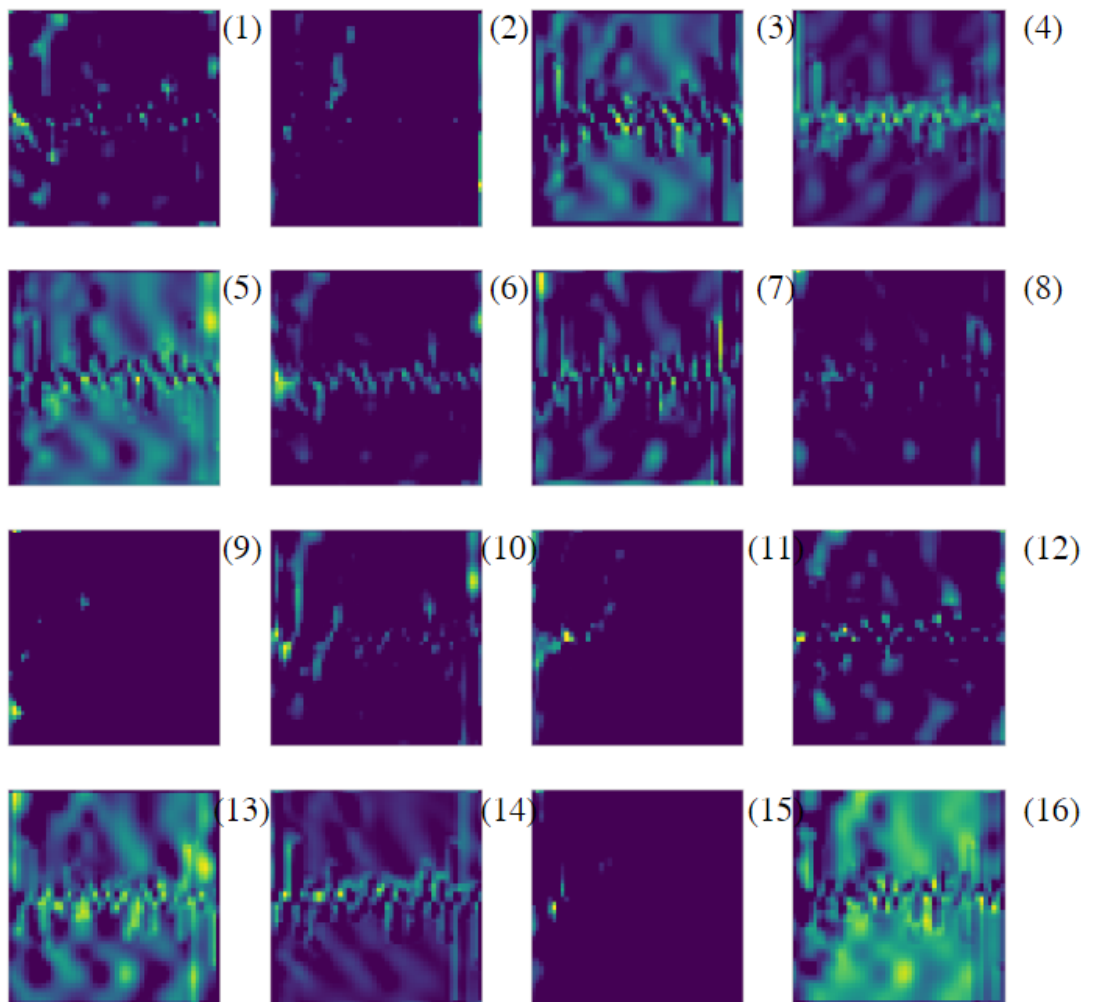


Figure 4.24: The filter matrices of the CNN layer which the AR order is 1

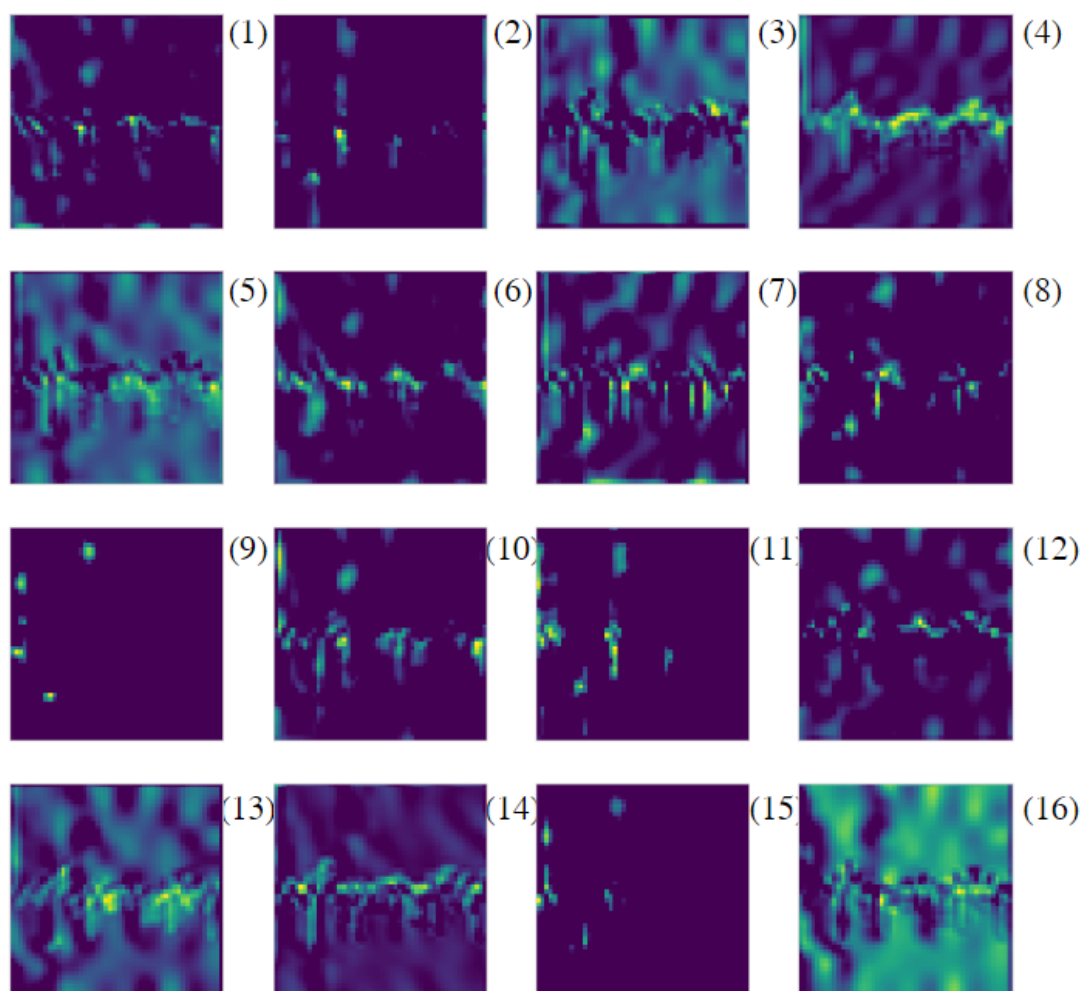


Figure 4.25: The filter matrices of the CNN layer which the AR order is 2

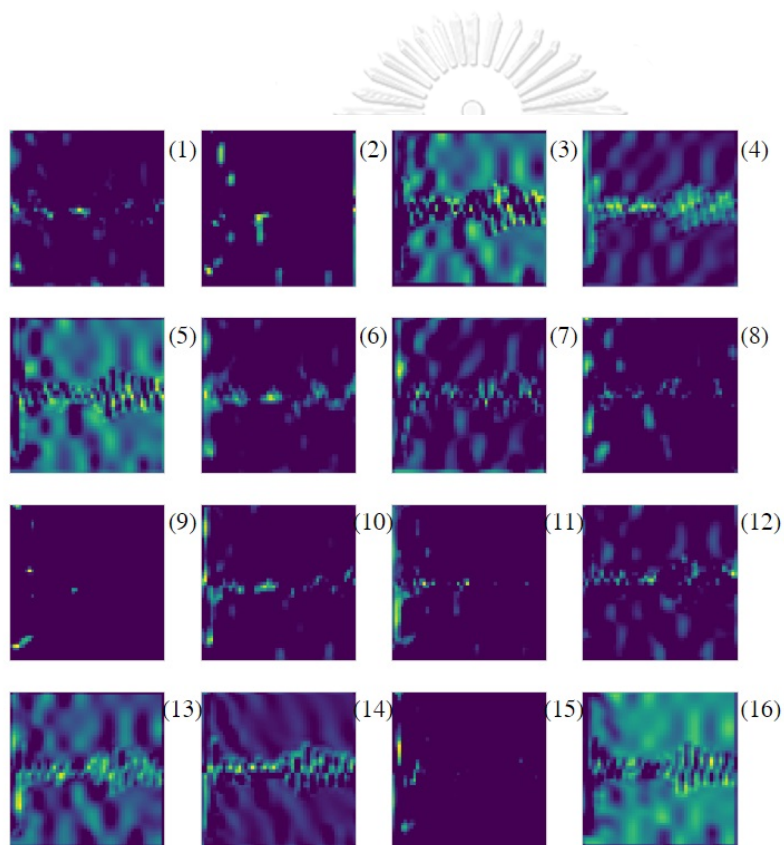


Figure 4.26: The filter matrices of the CNN layer which the AR order is 3

CHAPTER V

METHODOLOGY TO IDENTIFY THE SARIMA ORDER AND BUILD THE LINEAR TIME SERIES MODEL

The success of using a deep learning model for identifying the ARIMA order is now extended to deal with the seasonal component of the time series data via the time series analysis of the seasonal component. The process consists of two parts. The first part is to construct the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model to identify the SARIMA order and the second part is to construct the enhancing SARIMA forecasting model to forecast future values.

5.1 The ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model

This section demonstrates the rationale and the process of constructing the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model and its experimental results.

5.1.1 Constructing the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model

In the time series analysis, the autoregressive process and the moving average process are not sufficient to apply to the time series data having the seasonal component. There is a need for another concept called the spectral analysis. Before obtaining the AR order, the MA order and the differencing order, the seasonal length needs to be identified using the spectral analysis of the time series [41]. In the concept of the spectral analysis, a trend is removed from the time series first, then the spectral density function is ap-

proximated from the best fitting autoregressive model at frequency f . The autoregressive model with order p is shown as follows.

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \dots + \phi_p x_{t-p} + \epsilon_t$$

where $\phi_1, \phi_2, \dots, \phi_p$ are the coefficients of the autoregressive model and $\epsilon_t \underset{iid}{\sim} N(0, \sigma^2)$ from the Box-jenkins method.

Then, the spectral density function from the best fitting autoregressive model at frequency f can be approximated by

$$S(f, \phi_1, \phi_2, \dots, \phi_p, \sigma^2) = \frac{\sigma^2 \Delta t}{\left(\left| 1 - \sum_{k=1}^p \phi_k e^{-2i\pi f k \Delta t} \right| \right)^2}$$

where $0 < f < 0.5$, $i = \sqrt{-1}$ and Δt is the sample time interval which is 1.

The maximum in the spectral density function at f can be written as

$$f = \arg \max_{0 < f < 0.5} S(f, \phi_1, \phi_2, \dots, \phi_p, \sigma^2)$$

Then, the seasonal length will be $1/f$.

After obtaining the seasonal length, the time series data will be aggregated corresponding to the seasonal length. These aggregate time series data will be used to identify these seasonal orders of P , D and Q by the APEA model. The ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model is shown in Figure 5.1 and Algorithm 3 by defining s as the seasonal length, $\{x_t\}$ is the original time series data and $\{y_t\}$ is the aggregate time series data.

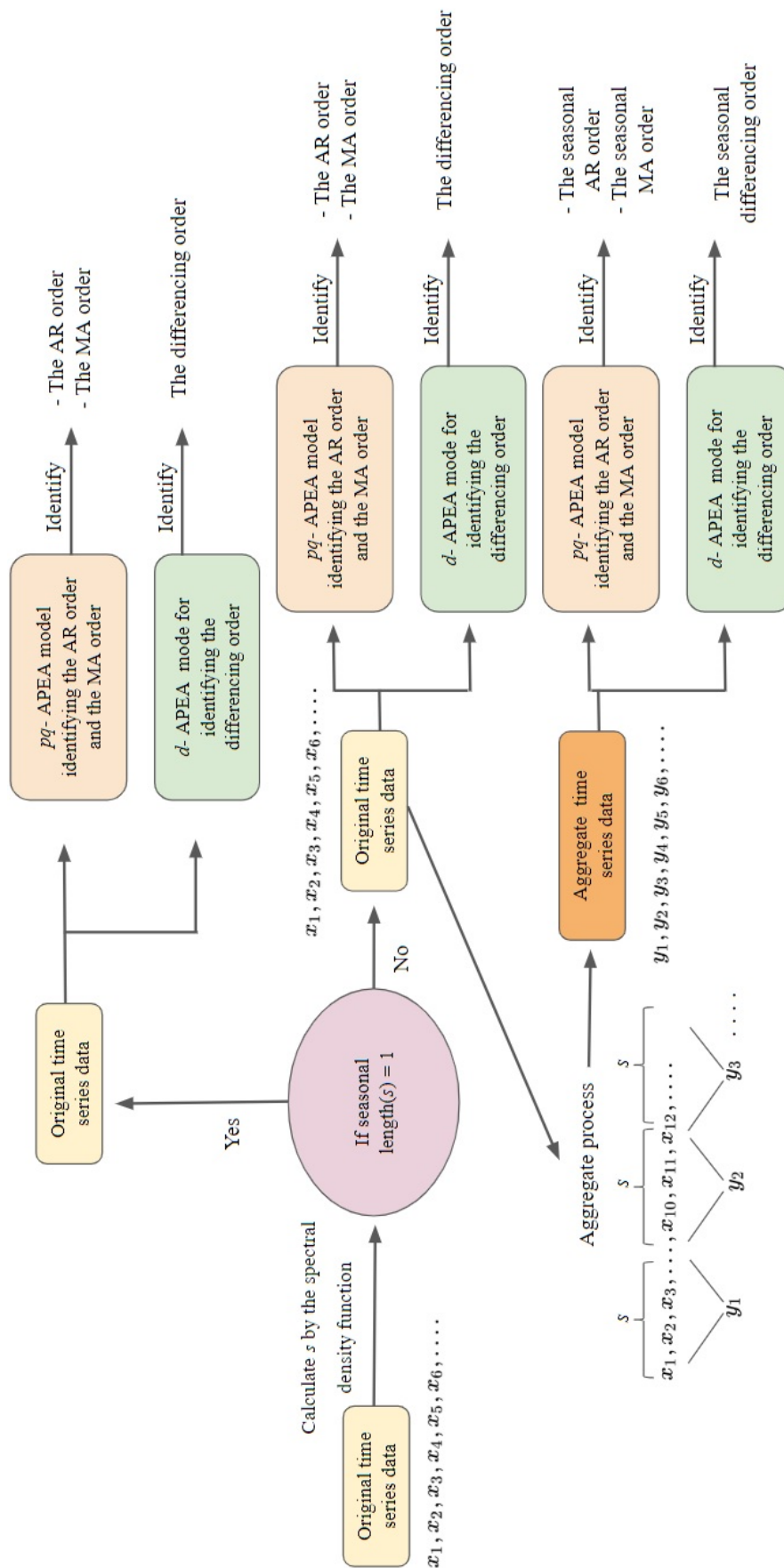


Figure 5.1: The ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model

Algorithm 3 : Pseudo code for the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model

Input: time series data

- 1: Original time series data = time series data
- 2: Calculate the seasonal period by the spectral density function
- 3: The seasonal period = s
- 4: **if** $s = 1$ **then**
- 5: - Submit the original time series to the pq -APEA model
- 6: - Submit the original time series to the d -APEA model
- 7: **Return:** the ARIMA order (p, d, q)
- 8: **else**
- 9: - Submit the original time series to the pq -APEA model
- 10: - Submit the original time series to the d -APEA model
- 11: **Return:** the ARIMA order (p, d, q)
- 12: - Aggregate the original time series data according to s
- 13: - Submit the aggregate time series to the pq -APEA model
- 14: - Submit the aggregate time series to the d -APEA model
- 15: **Return:** the Seasonal ARIMA order (P, D, Q)

5.1.2 Experimental results of the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model

To test the performance of the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model, the experimental results are set to include the case of identifying the seasonal AR orders from 0-7, the case of identifying the seasonal MA orders from 0-7 and the case of identifying the seasonal differencing orders from 0-3. The setting of each model having different channels and type of inputs are listed in Table 5.1. The APEA model is chosen among other models from Chapter 4 since it obtains the best performance. Moreover, it uses all inputs from the time series analysis including ACF, PACF and ESACF. Note that the ResNet50 model is not used since it is not designed to deal with the seasonal component. Hence, only the auto-ARIMA model can be used for comparison.

	Model	Case: identifying order P, Q and D Channel(s)
Model A	Proposed model	3 channels: ESACF, PACF and ACF images
Model L	auto-ARIMA (AIC criteria)	Series

Table 5.1: Description of each channel in the models for identifying the SARIMA order

From Table 5.1, the proposed model is called A having the inputs of the ESACF images, the PACF images and the ACF images and L is represented to the auto-ARIMA model. The seasonal length of the time series data for both methods is derived using the spectral analysis.

Results of identifying the P order

From Figure 5.2, A gives the best results of the precisions and the f1-scores except the cases of identifying $P = 1$ and 2 whereas L provides the good performance in the recalls except $P = 0$ and 3. Although the recalls of A are slightly lower than L, A still has much higher precisions than L. Consequently, the f1-scores of A are higher than L. It is suggested that A can identify the seasonal order better than L.

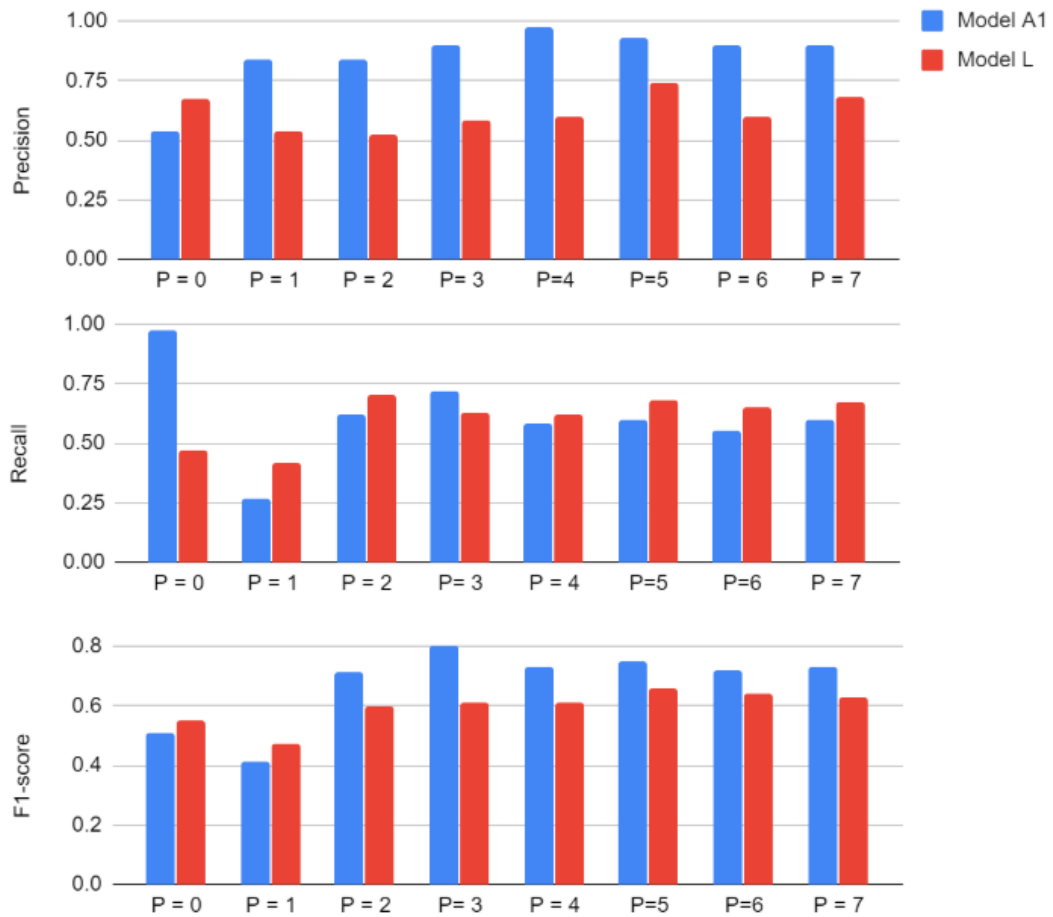


Figure 5.2: The precisions, the recalls and the f1-scores of the models identifying P

Results of identifying the Q order

From Figure 5.3, A outperforms L via the precisions and the f1-scores except the case of identifying $Q = 0$ via the precisions like the case of identifying P . Nevertheless, L provides slightly higher recalls except the case of identifying $Q = 0, 1$ and 3 . Although A gives smaller recalls than L, A shows the high performance in the precisions. Consequently, the f1-scores of A are higher than L. From these results, it can be concluded that A can identify the seasonal order Q better than L.

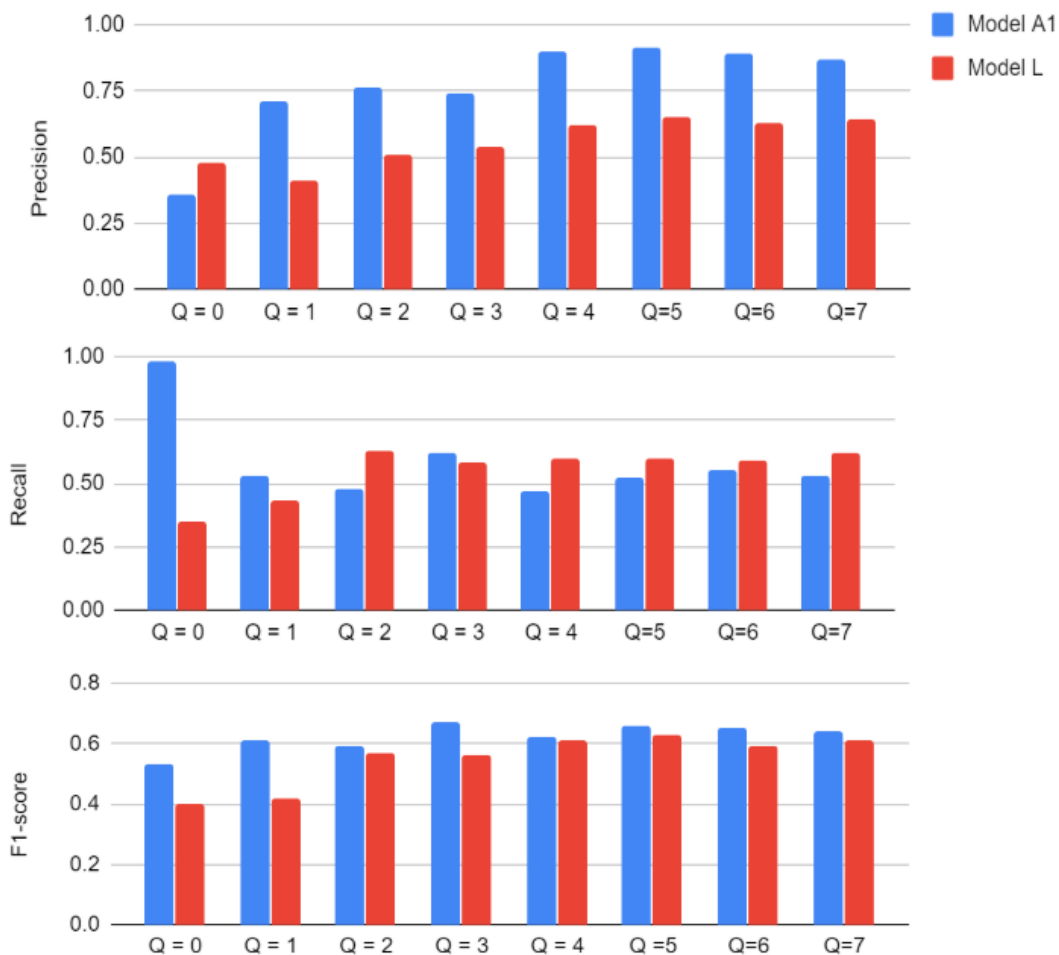


Figure 5.3: The precisions, the recalls and the f1-scores of the models identifying Q

Results of identifying the D order

Figure 5.4 shows the precisions, the recalls and the f1-scores between A and L. It is clear that A provides the better performance of the precisions, the recalls and the f1-scores than L.

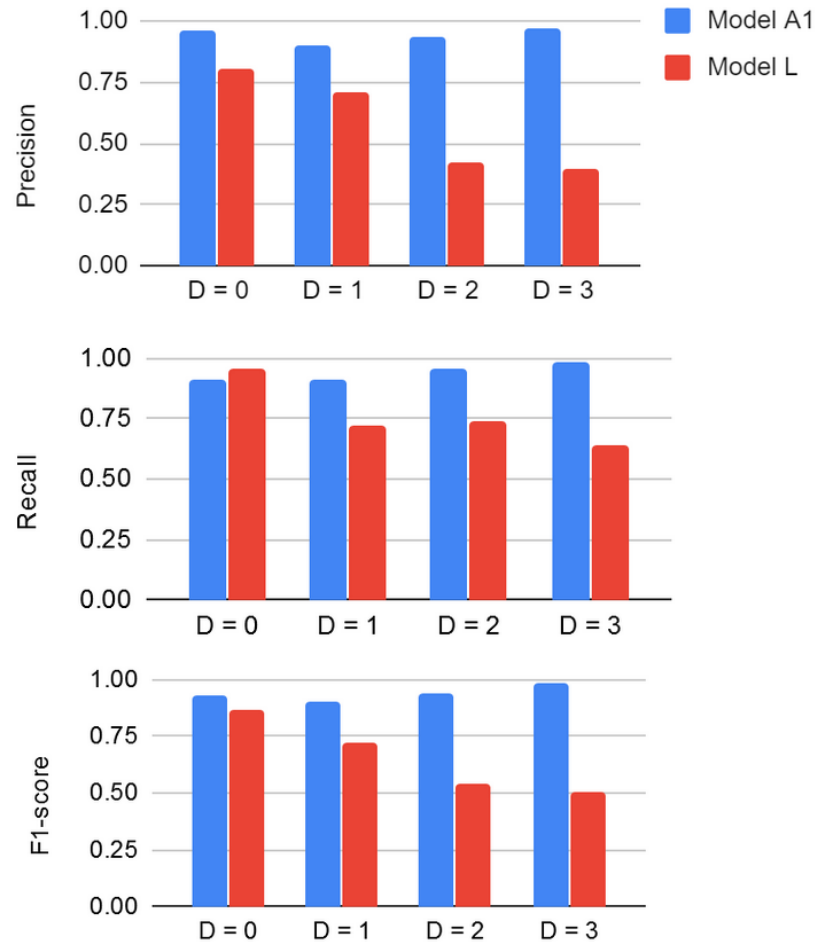


Figure 5.4: The precisions, the recalls and the f1-scores of the models identifying D

5.1.3 The conclusion of the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model

The ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model is extended from the APEA model to identify the seasonal order of the SARIMA model. The proposed model uses the spectral density function to recognize the seasonal length of the time series data. Then the concept of the aggregation before submitting to the APEA model is used to identify the seasonal order. The results show that the proposed model gives the best precisions, the recalls and the f1-scores. It is suggested that the APEA model applying with the spectral density function and the aggregate concept can recognize the seasonal order within the time series data. Next, this proposed model is used to build the SARIMA model to forecast the future values.

5.2 The enhancing SARIMA forecasting model via the deep learning algorithm

This section demonstrates the process of constructing the enhancing SARIMA forecasting model via the deep learning algorithm to build the SARIMA model and explains its experimental results.

5.2.1 Constructing the enhancing SARIMA forecasting model via the deep learning algorithm

The process starts by converting the original time series data to the images of ACF, PACF, ESACF and differencing time series to submit to the ACF-PACF-ESACF convolutional neural network ARIMA order identification model for identifying the ARIMA order. Moreover, the original time series data are sent to the aggregate process to get the aggregate time series data submitted to the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model for identifying the SARIMA order. Finally, all orders are used to fit all coefficients of the SARIMA model by the Box-Jenkins method. The process of the enhancing SARIMA forecasting model via the deep learning algorithm is shown in Algorithm 4 and Figure 5.5.

Algorithm 4 : Pseudo code for the enhancing SARIMA forecasting model via the deep learning algorithm

Input: time series data

- 1: Original time series data = time series data
 - 2: Submit the original time series to the APEA model
 - 3: **Return:** the ARIMA order (p, d, q)
 - 4: Calculate the seasonal period by the spectral density function
 - 5: The seasonal period = s
 - 6: Aggregate the original time series data according to s
 - 7: Submit the aggregate time series to the seasonal APEA model
 - 8: **Return:** the seasonal ARIMA order (P, D, Q)
 - 9: Submit the SARIMA order $(p, d, q) \times (P, D, Q)$ to the Box-Jenkins method
 - 10: **Return:** the seasonal ARIMA model
-

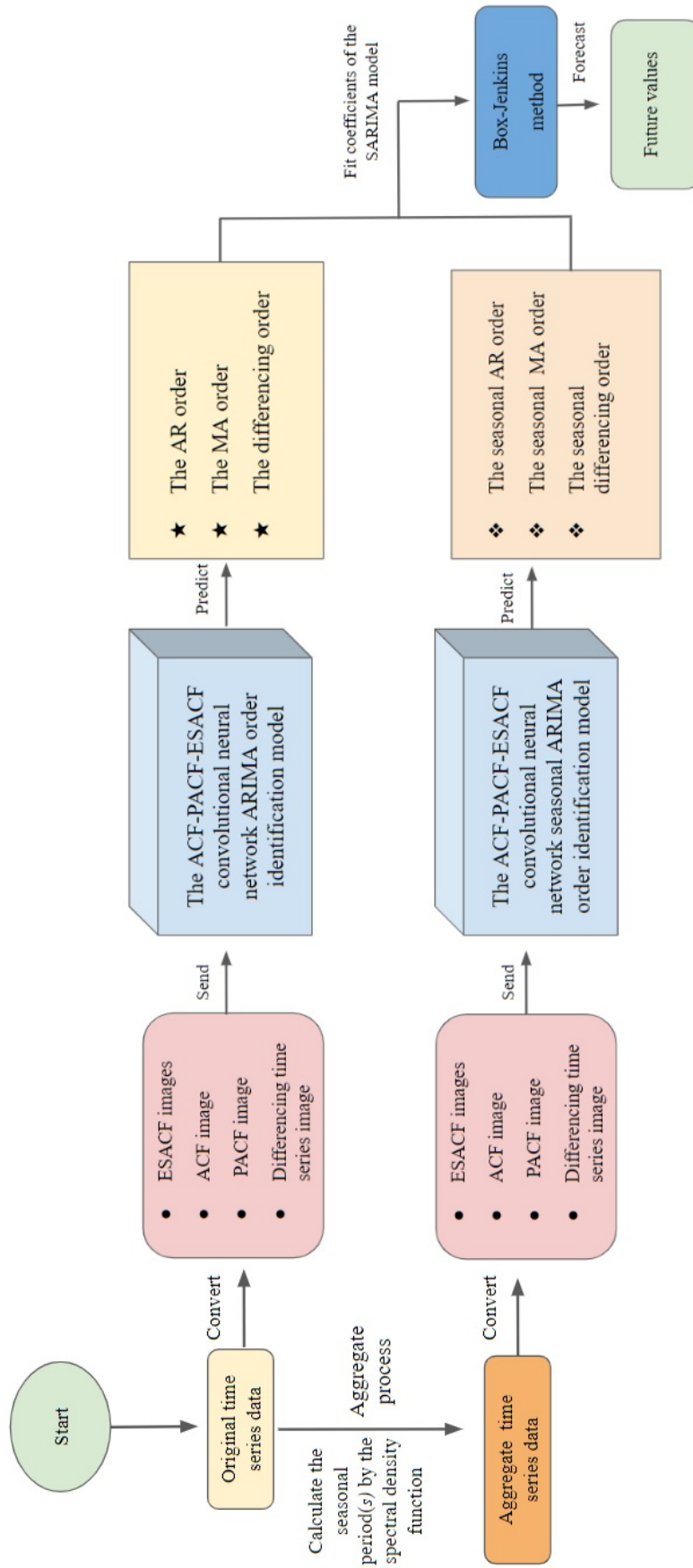


Figure 5.5: The enhancing SARIMA forecasting model via the deep learning algorithm

5.2.2 Experimental results of the enhancing SARIMA forecasting model via the deep learning algorithm

To test the performance of the SARIMA forecasting model from the enhancing SARIMA forecasting model via the deep learning algorithm, it is applied to two collections of the time series data. The first collection uses the synthetic time series data of fixed orders and the second collection uses the real world time series data.

Results of forecasting the synthetic time series data

The enhancing SARIMA forecasting model via the deep learning algorithm is applied to 10 random synthetic time series data that have the stationary and the invertibility properties. Figure 5.6 to Figure 5.10 show the graphs of forecasting values from the out-sample data and p -values from the Ljung-Box test. The residuals from the appropriate forecasting model should exhibit no correlated errors that is its values should be above the dot lines representing p -value = 0.01 and p -value = 0.05. The results show that the proposed algorithm outperforms the auto-ARIMA model. All p -values of the residuals are above 0.01 and 0.05 indicating that the residuals are independent and correspond to the white noise process whereas some residuals from the auto-ARIMA model fail the test.

Moreover, the graphs of the forecasting data between two models suggest that the proposed algorithm can forecast values closer to the out-sample data than the auto-ARIMA model. In addition, Table 5.2 shows that the proposed algorithm provides better MAPE in 7 out of 10 synthetic time series and better SMAPE, RMSE and MAE in 9 out of 10 synthetic time series.

To show all parameters in the models, the orders and the coefficients of the ARIMA models are shown in Table 5.3 and Table 5.4. These tables show that the ARIMA orders from the proposed algorithm are closer to the orders from the synthetic time series data than the ARIMA orders from the auto-ARIMA model in all cases of synthetic time series data. Especially, in the case of the time series number 1, 5 and 6, the ARIMA orders from the proposed algorithm are similar to the orders from the synthetic time series data.

From this reason, the coefficients from the orders that come from the proposed algorithm can be used to build the ARIMA model and predict the future values more efficiently than the auto-ARIMA model.



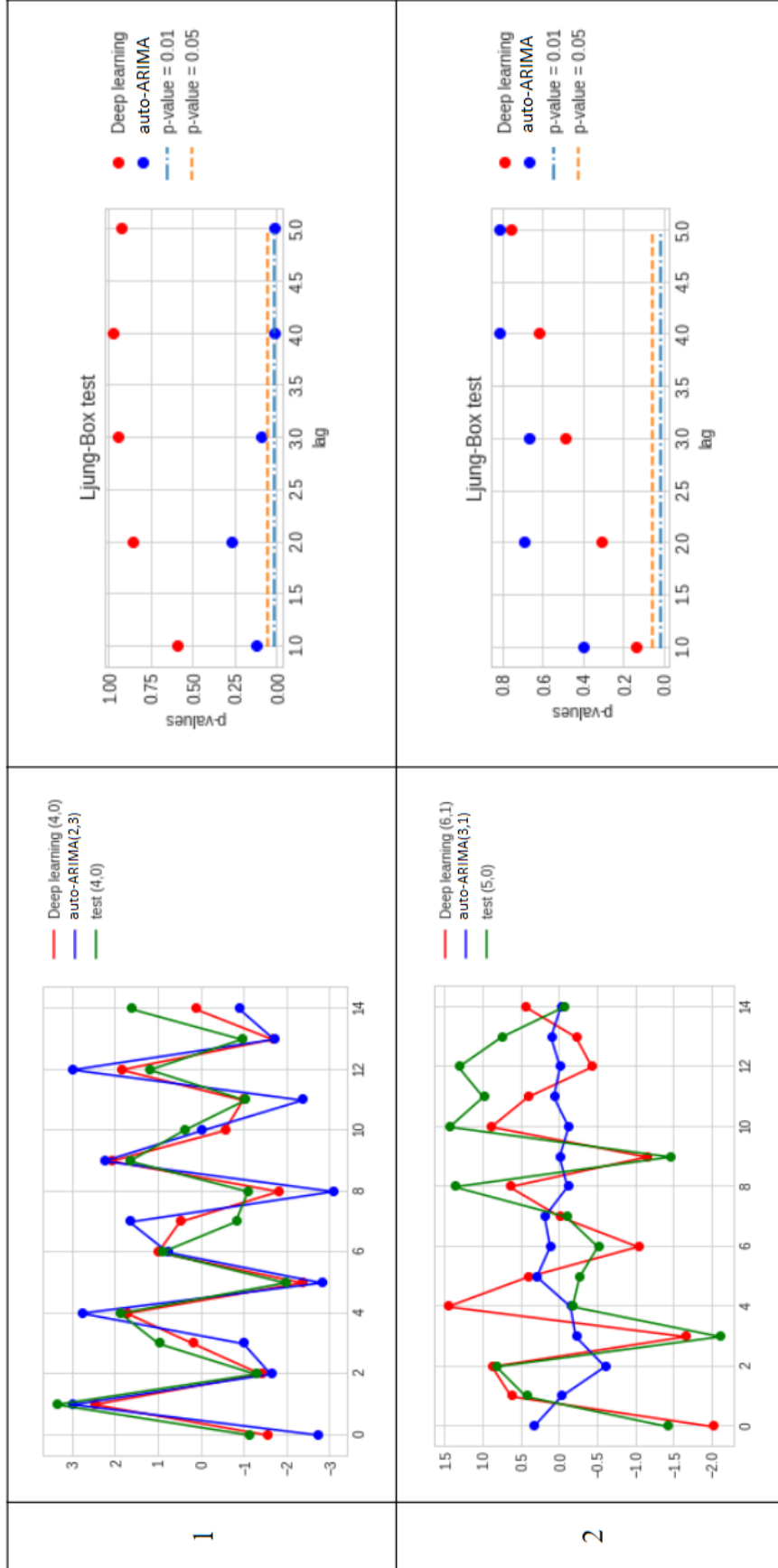


Figure 5.6: Graphs of out-sample data and forecasting learning and Ljung-Box test from synthetic time series 1 and 2.

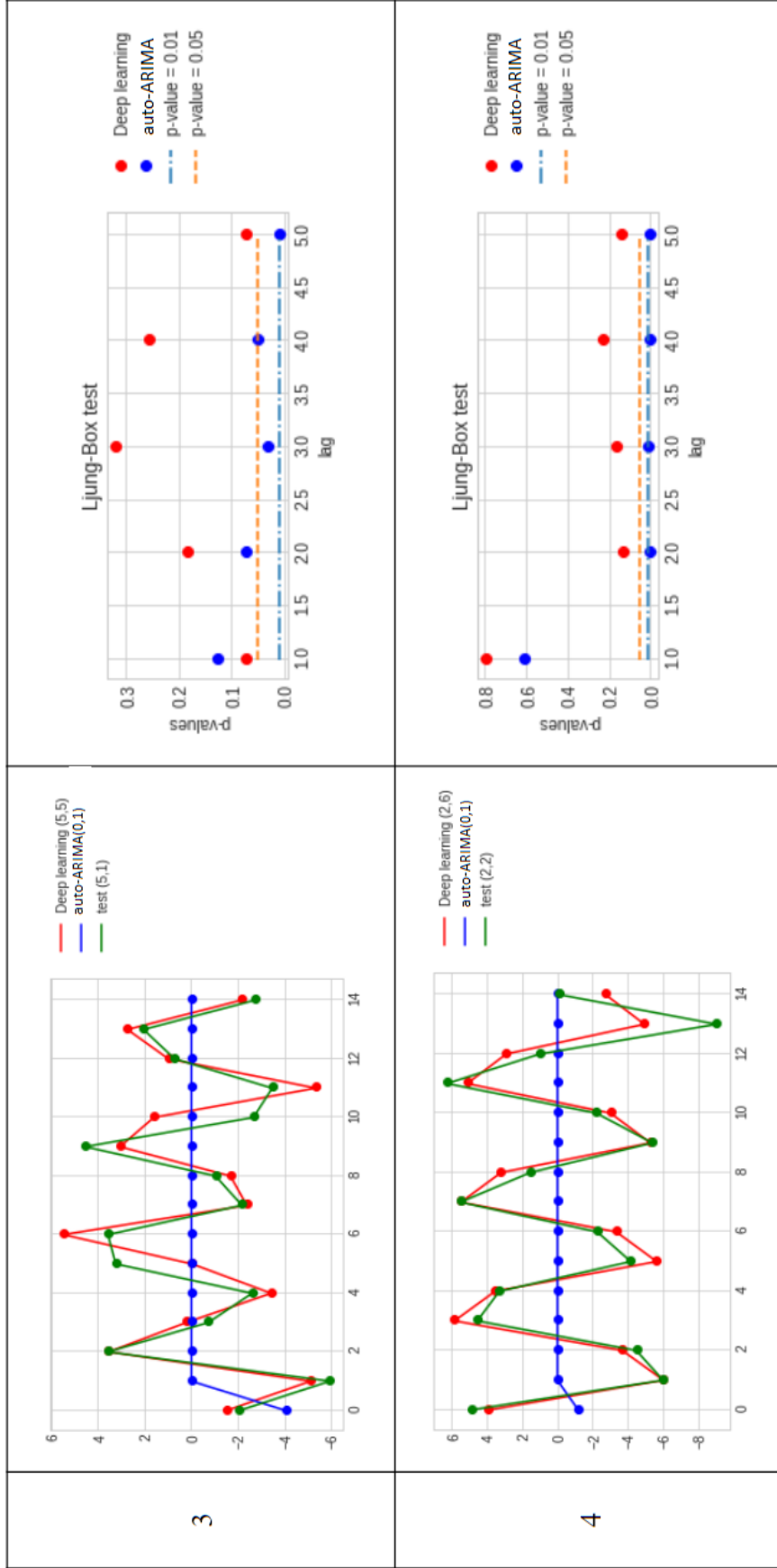


Figure 5.7: Graphs of out-sample data and forecasting data and Ljung-Box test from synthetic time series 3 and 4.

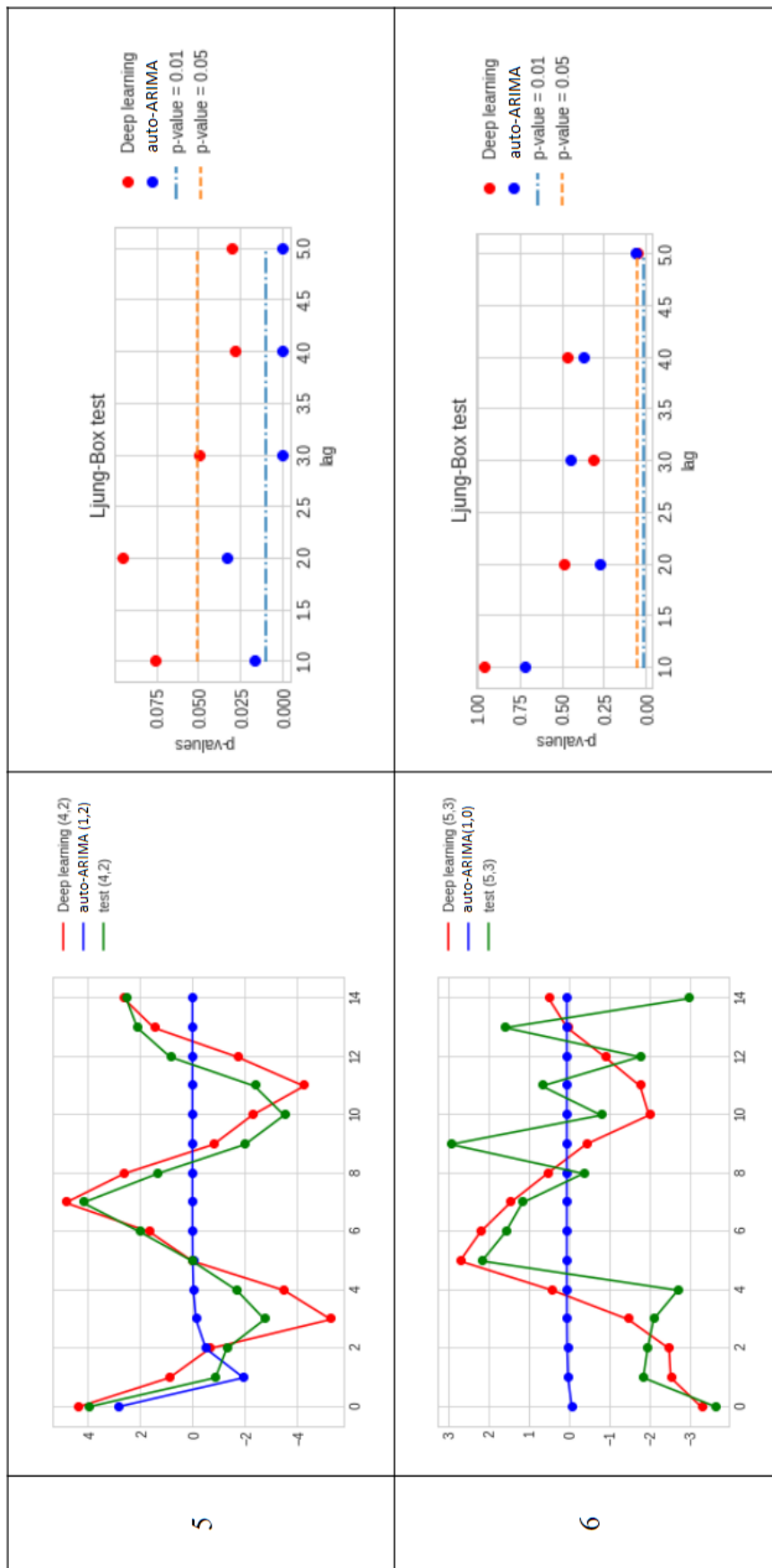


Figure 5.8: Graphs of out-sample data and forecasting data and Ljung-Box test from synthetic time series 5 and 6.

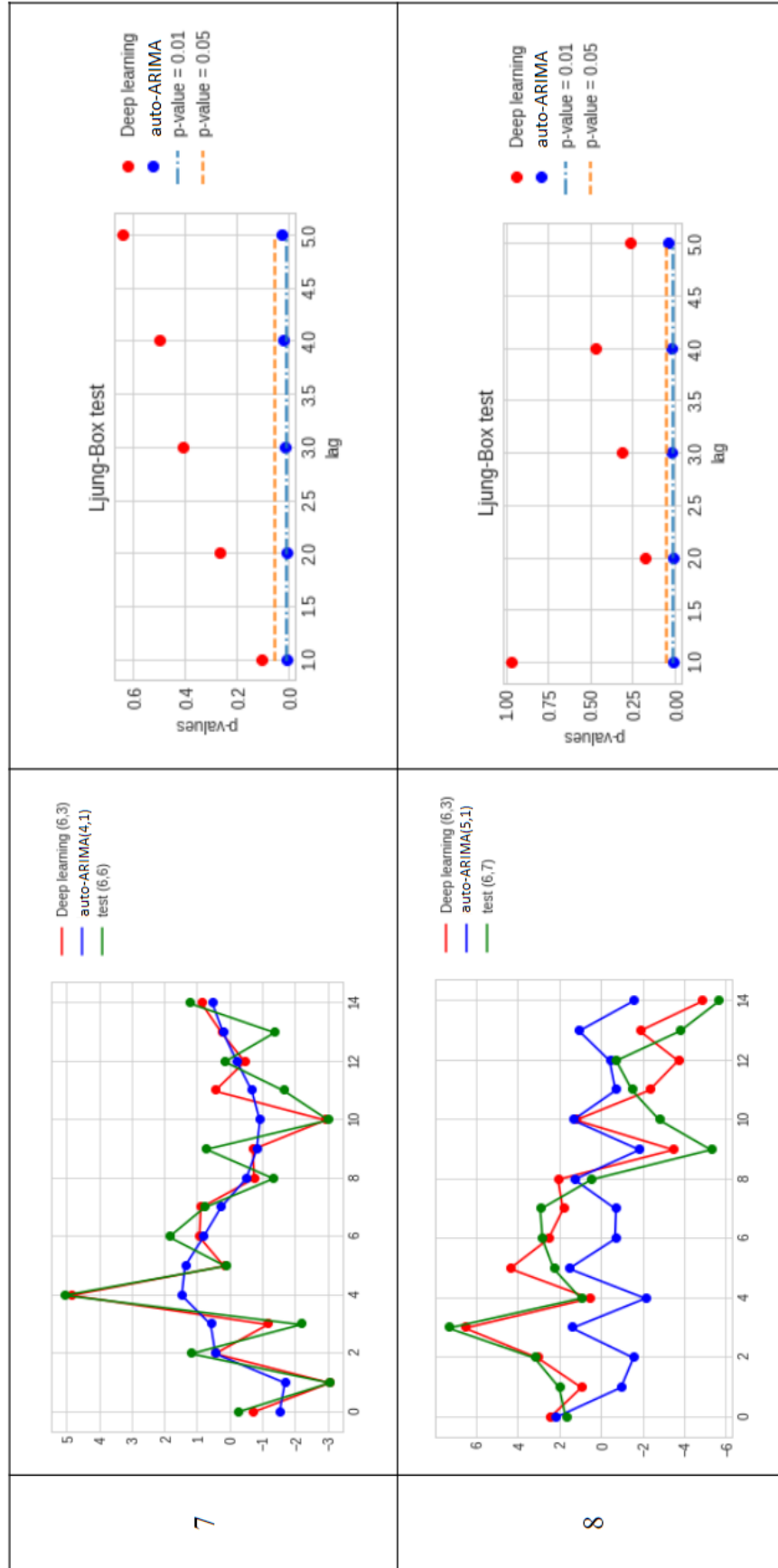


Figure 5.9: Graphs of out-sample data and forecasting data and Ljung-Box test from synthetic time series 7 and 8.

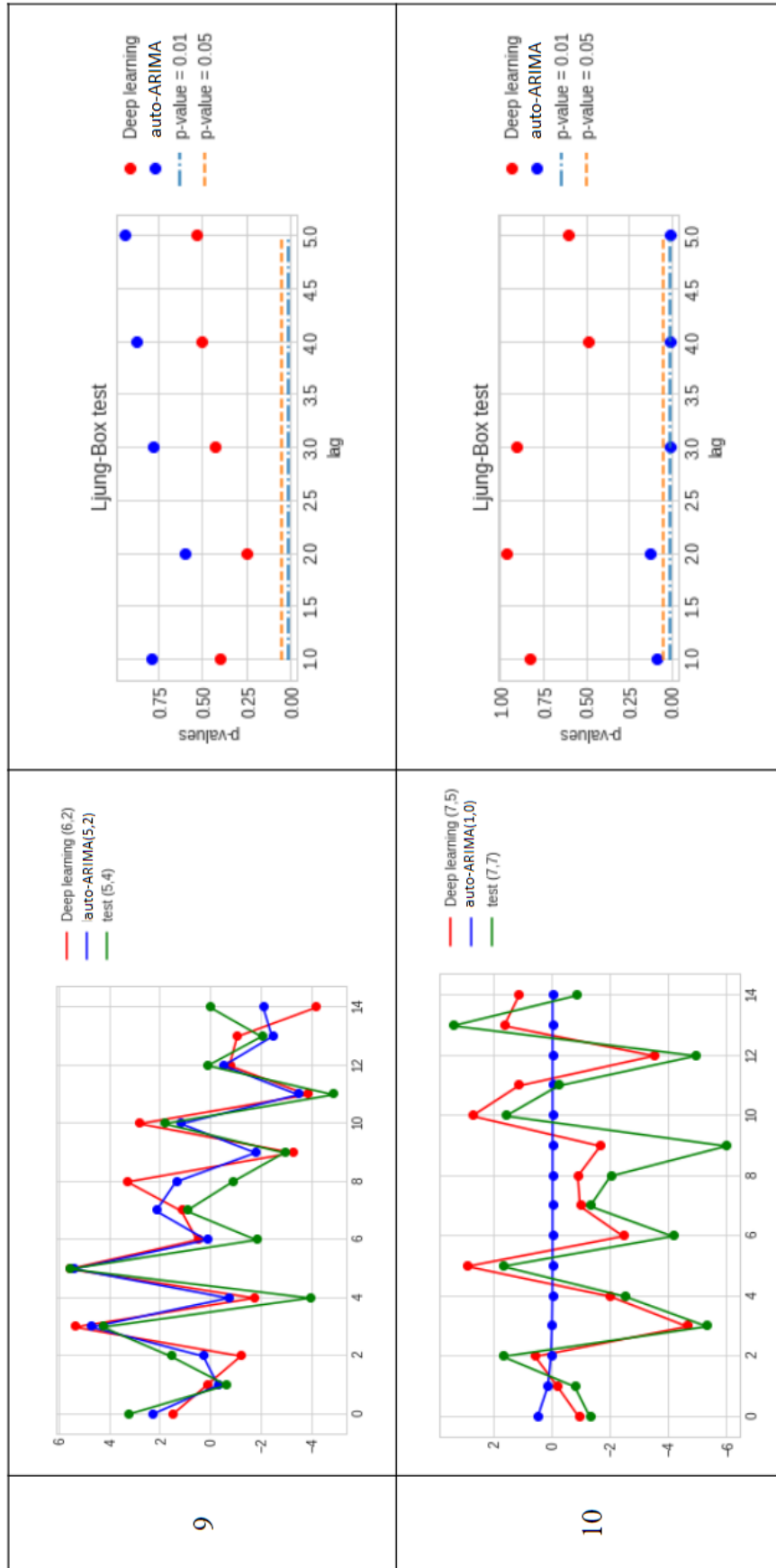
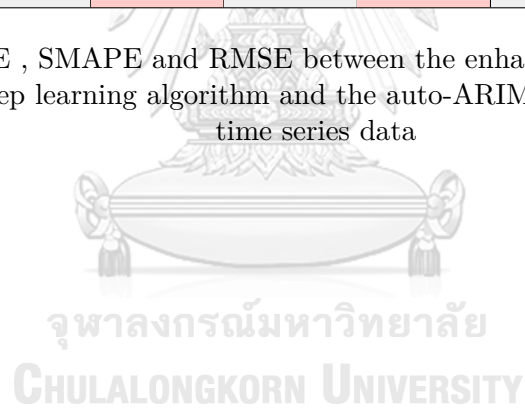


Figure 5.10: Graphs of out-sample data and forecasting data and Ljung-Box test from synthetic time series 7 and 8.

	MAPE		SMAPE		RMSE		MAE	
	Deep learning	auto-ARIMA	Deep learning	auto-ARIMA	Deep learning	auto-ARIMA	Deep learning	auto-ARIMA
1	0.61	1.09	0.66	0.90	0.74	1.43	0.61	1.21
2	1.76	1.18	1.02	1.72	0.8	1.14	0.64	0.96
3	0.51	0.99	0.63	1.89	1.68	3.06	1.23	2.74
4	2.18	1.01	0.40	1.98	1.61	4.74	1.22	4.13
5	0.77	1.01	0.67	1.71	1.39	2.15	1.13	1.88
6	0.97	1.00	1.01	1.94	1.76	2.08	1.37	1.88
7	0.89	1.84	0.84	1.27	0.92	1.61	0.69	1.38
8	2.76	1.42	1.02	0.95	2.03	1.47	1.58	1.21
9	0.90	1.13	0.62	1.36	1.72	3.41	1.38	2.91
10	0.96	1.01	0.78	1.91	1.61	3.07	1.31	2.54

Table 5.2: MAPE , SMAPE and RMSE between the enhancing SARIMA forecasting model via the deep learning algorithm and the auto-ARIMA model of the synthetic time series data



	Actual	Deep learning		auto-ARIMA	
#	order (p, q)	order (p, q)	AR and MA coefficients	order (p, q)	AR and MA coefficients
1	(4,0)	(4,0)	AR: {-0.1682, -0.3184, 0.0181, -0.8056} MA: -	(2,5)	AR: {-1.3137, -0.9992} MA: {1.6266, 1.4659, 0.3939}
2	(5,0)	(6,1)	AR: {-1.2601, -0.9587, -0.9853, -1.0495, -1.1457, -0.4189} MA: {1.1024}	(3,1)	AR: {0.0595, 0.2787, -0.5769} MA: {-0.4249}
3	(5,1)	(5,5)	AR: {0.1180, -0.1075, 0.0025, 0.1857, -0.8698} MA: {-1.5006, -0.1764, 0.1357, 0.2419, -0.1123}	(0,1)	AR: - MA: {-0.9999}
4	(2,2)	(2,6)	AR: {-0.0074, -0.9633} MA: {-0.1026, 0.2314, 0.1572, -0.1429, -0.1113, 0.1622}	(0,1)	AR: - MA: {-0.9998}
5	(4,2)	(4,2)	AR: {-0.0291, -0.1037, -0.1367, -0.8363} MA: {0.0651, -1.0396}	(1,2)	AR: {0.2624} MA: {-0.0001, -0.9999}

Table 5.3: The coefficients and the order from the synthetic time series number 0-5

	Actual	Deep learning		auto-ARIMA	
#	order (p, q)	order (p, q)	AR and MA coefficients	order (p, q)	AR and MA coefficient
6	(5,3)	(5,3)	AR: {0.2259,-0.1751, -0.1790,0.1409,-0.6864} MA: {0.2046,-0.0670, -0.9357}	(1,0)	AR: {0.2397} MA: -
7	(6,6)	(6,3)	AR: {-0.2500,-0.0934, -0.1446,-0.0996, -0.2818,-0.5994} MA: {-0.1381,0.4091, -0.0606}	(4,1)	AR: {0.3525,0.4452, -0.2562,-0.4160} MA: {-0.8114}
8	(6,7)	(6,3)	AR: {0.2448,0.1459, 0.4807,0.0456, 0.2804,-0.8902} MA: {0.3571,0.1030, -0.7982}	(5,1)	AR: {-0.1741,0.3894, 0.2286,-0.2907, 0.3368} MA: {0.8174}
9	(5,4)	(6,2)	AR: {-0.0284,0.3543, 0.4418,0.0009,-0.0403, -0.7253} MA: {-0.0029,0.5606}	(5,2)	AR: {-0.0798,1.1634, 0.3301,-0.8168, -0.2197} MA: {-0.0020,-0.9980}
10	(7,7)	(7,5)	AR: {0.0190,0.0744, 0.2288,0.3234, 0.2180,0.2330, -0.7894} MA: {0.1571,0.4137, 0.3674,-0.1343, 0.7548}	(1,0)	AR: {0.3197} MA: -

Table 5.4: The coefficients and the order from the synthetic time series number 6-10

Results of forecasting the real world time series data

To confirm the performance of the enhancing SARIMA forecasting model via the deep learning algorithm, ten real world time series data are used. They are from the fpp2 packages in R which are “ausair”, “ausbeer”, “austourists”, “elecequip”, “goog200”, “h02”, “hyndsight”, “qauselec”, “qgas” and “sunspotarea”. The details of each time series data are described in Table 5.5 and Table 5.6

Data	Quote Description from “fpp2” package in R	Source
ausair	“Total annual air passengers (in millions) including domestic and international aircraft passengers of air carriers registered in Australia. 1970-2016.”	World Bank. https://data.is/x5KiEO
ausbeer	“Total quarterly beer production in Australia (in megalitres) from 1956:Q1 to 2010:Q2”	Australian Bureau of Statistics. Cat. 8301.0.55.001
austourists	“Quarterly visitor nights (in millions) spent by international tourists to Australia. 1999-2015.”	Tourism Research Australia.
elecequip	“Monthly manufacture of electrical equipment: computer, electronic and optical products. January 1996 - March 2012. Data adjusted by working days; Euro area (17 countries). Industry new orders index. 2005=100.”	Eurostat. https://data.is/y6dO8i
goog200	“Closing stock prices of GOOG from the NASDAQ exchange, for 1000 consecutive trading days between 25 February 2013 and 13 February 2017. Adjusted for splits. goog200 contains the first 200 observations from goog.”	https://goo.gl/5KBjL5

Table 5.5: Description I of the real world datasets

Data	Quote Description from “fpp2” package in R	Source
h02	“Total monthly scripts for pharmaceutical products falling under ATC code H02, as recorded by the Australian Health Insurance Commission. Measured in millions of scripts.”	Medicare Australia
hyndsight	“Hyndsight is Rob Hyndman’s personal blog at https://robjhyndman.com/hyndsight . This series contains the daily pageviews for one year, beginning 30 April 2014. The frequency is set to 7, to allow the weekly pattern to be modelled.”	Rob Hyndman
qauselec	“Total quarterly electricity production in Australia (in billion kWh) from 1956:Q1 to 2010:Q2.”	Australian Bureau of Statistics. Cat. 8301.0.55.001.
qgas	“Total quarterly gas production in Australia (in petajoules) from 1956:Q1 to 2010:Q2.”	Australian Bureau of Statistics. Cat. 8301.0.55.001.
sunspotera	“Annual averages of the daily sunspot areas (in units of millionths of a hemisphere) for the full sun. The Royal Greenwich Observatory compiled daily sunspot observations from May 1874 to 1976.”	NASA

Table 5.6: Description II of the real world datasets

The results of this experiment appear in Figure 5.11, Figure 5.15 and Table 5.7 which demonstrate that both methods can be applied to the real world time series data. Table 5.7 shows that both methods have similar MAPE, SMAPE and RMSE. However, there are some cases in which the proposed algorithm passes the Ljung-Box test while the auto-ARIMA model fails, such as “h02”. Although most forecasting results from both methods pass the test, it is found that the proposed algorithm can forecast the future values closer than the auto-ARIMA model for “ausbeer”, “austourists”, “h02”, “qauselec”

and “sunspotarea”. Nevertheless, there is one dataset, “goog200”, which both methods fail Ljung-Box test and cannot predict near the actual values. This “goog200” series has multiple changes so it may not be captured by any ARIMA model.

The parameters of each model are shown in Table 5.8 and Table 5.9. From these tables, both methods can identify the SARIMA order differently which the SARIMA models from both method can be used to forecast the future values effectively. Nevertheless, there is the SARIMA order from the both method that cannot fit with the real world time series number 5 which is “goog200”. The SARIMA orders from the proposed algorithm and the auto-ARIMA model are $(0, 2, 1) \times (0, 0, 0)$ and $(0, 1, 0) \times (0, 0, 0)$, respectively which these orders are only the relationship of random error terms than cannot be fitted to the time series effectively.

	Name	MAPE		SMAPE		RMSE	
		Deep learning	auto-ARIMA	Deep learning	auto-ARIMA	Deep learning	auto-ARIMA
1	ausair	0.3109	0.4672	0.3109	0.4689	0.0235	0.0353
2	ausbeer	4.08731	2.9372	4.1996	2.9499	0.0622	0.0506
3	austourists	3.2012	2.9268	3.2616	2.9095	0.1161	0.1056
4	elecequip	0.814	1.1249	0.8093	1.1162	0.0497	0.0622
5	goog200	0.1356	0.1196	0.1355	0.1196	0.0105	0.0093
6	h02	0.863	1.242	0.723	1.479	0.1424	0.2551
7	hyndsight	1.1066	0.8601	1.1131	0.8598	0.099	0.0813
8	qauselec	1.0464	0.9343	1.0393	0.931	0.0508	0.0422
9	qgas	0.5955	0.61	0.598	0.6124	0.0369	0.0372
10	sunspotarea	0.199	0.3495	0.1548	0.252	1.2674	2.049

Table 5.7: MAPE , SMAPE and RMSE between the enhancing SARIMA forecasting model via the deep learning algorithm and the auto-ARIMA model of the real world datasets

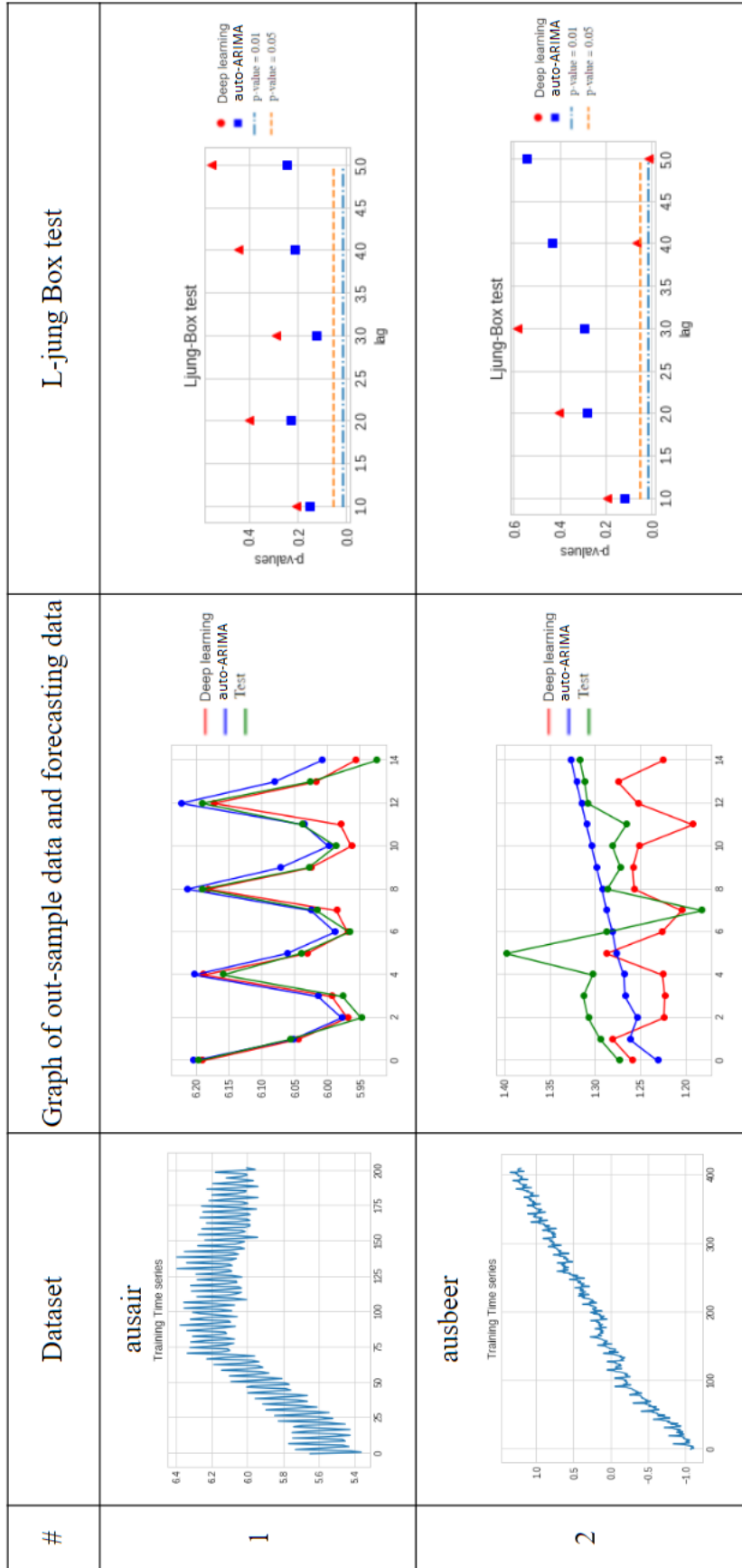


Figure 5.11: Graphs of out-sample data and forecasting data and Ljung Box test from the real world time series 1 and 2.

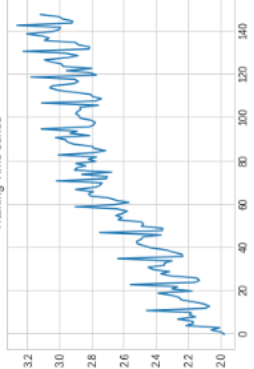
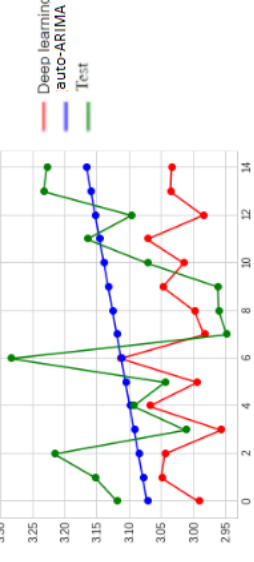
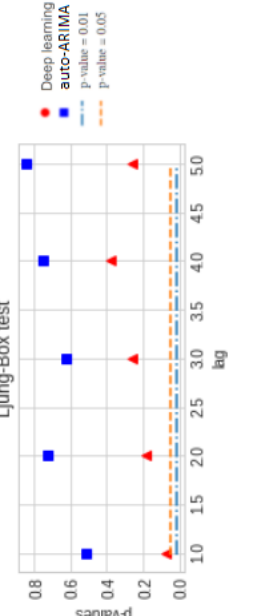
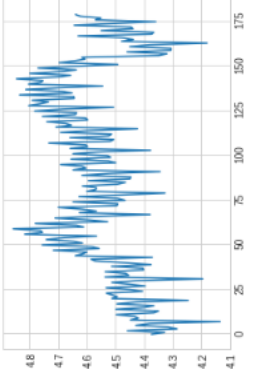
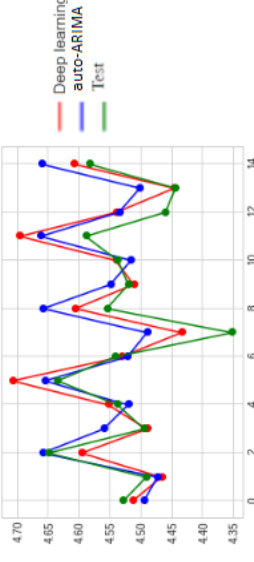
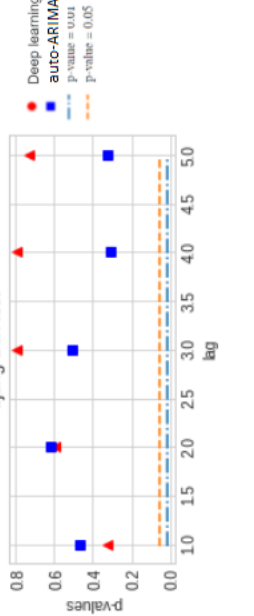
#	Dataset	Graph of out-sample data and forecasting data	Ljung-Box test
3	<p data-bbox="518 1675 544 1800">autourists</p> 		
4	<p data-bbox="869 1675 895 1800">elecequip</p> 		

Figure 5.12: Graphs of out-sample data and forecasting data and Ljung-Box test from the real world time series 3 and 4.

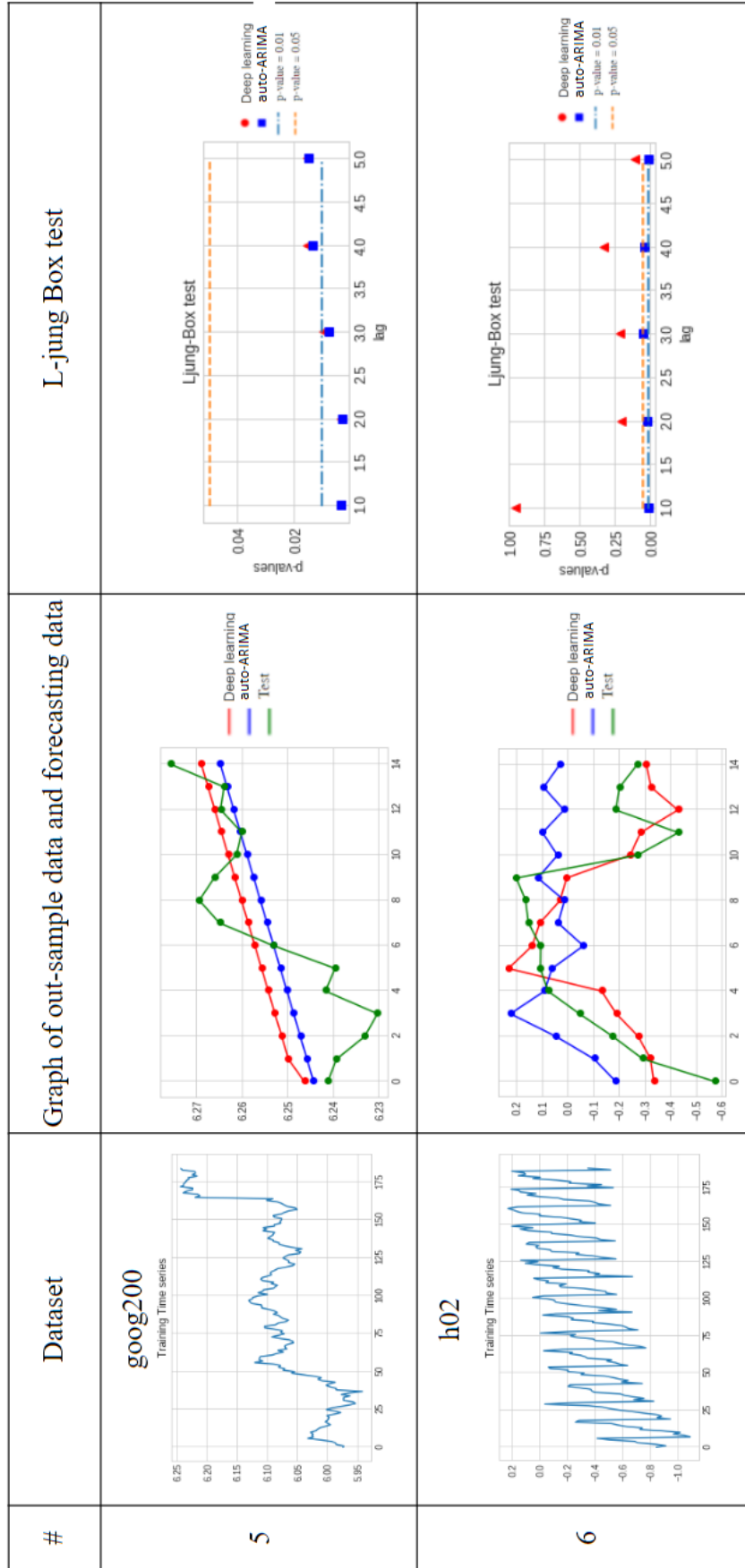


Figure 5.13: Graphs of out-sample data and forecasting data and Ljung-Box test from the real world time series 5 and 6.

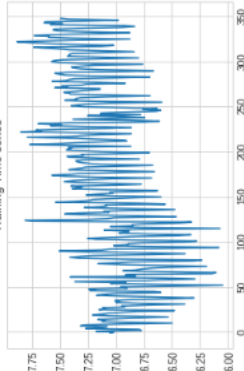
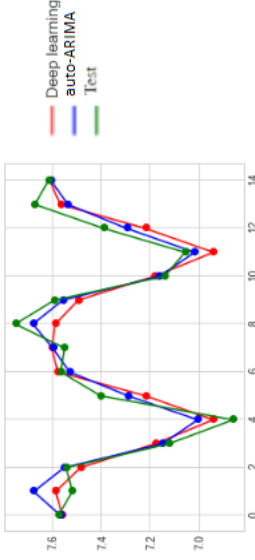
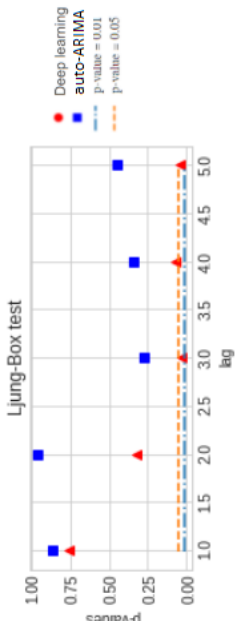
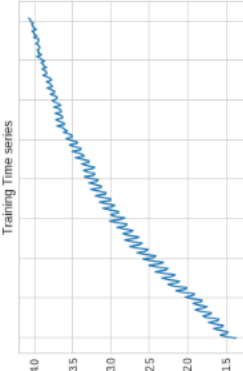
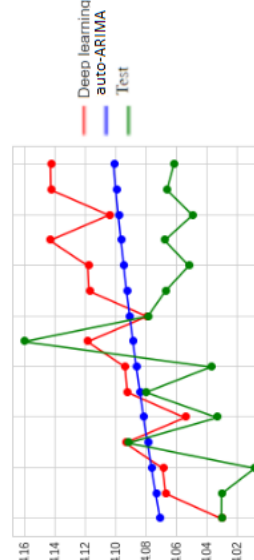
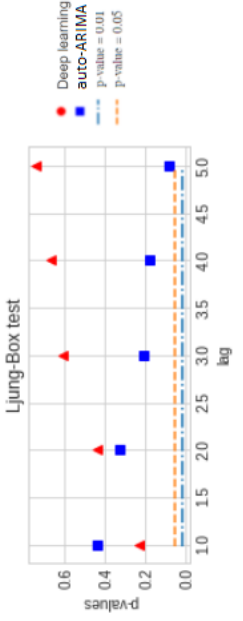
#	Dataset	Graph of out-sample data and forecasting data	Ljung-Box test
7	<p data-bbox="512 1668 544 1798">hyndsight</p> 		
8	<p data-bbox="863 1675 895 1798">gauselec</p> 		

Figure 5.14: Graphs of out-sample data and forecasting data and Ljung-Box test from the real world time series 7 and 8.

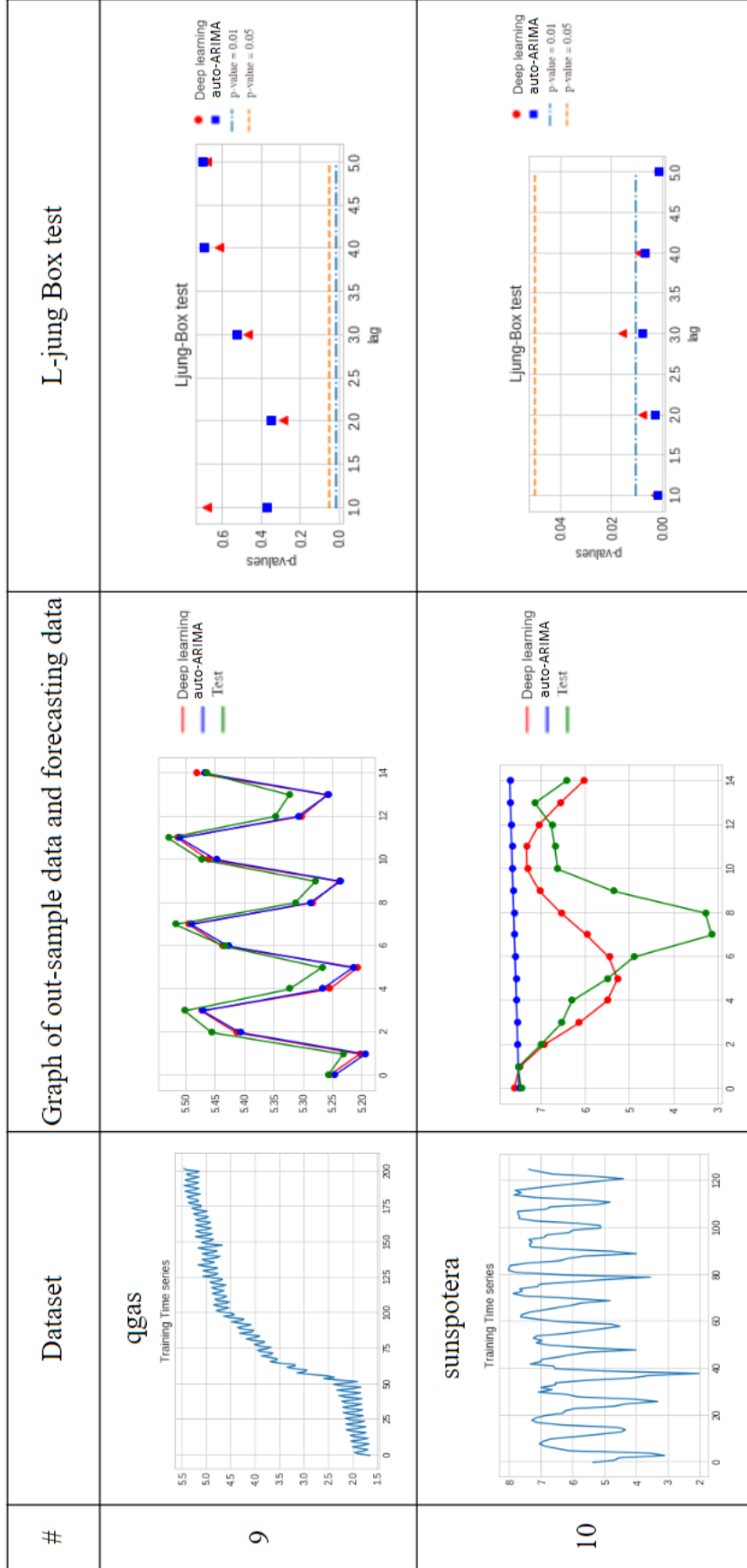


Figure 5.15: Graphs of out-sample data and forecasting data and Ljung-Box test from the real world time series 9 and 10.

	Deep learning		auto-ARIMA	
#	(p, d, q) × (P, D, Q)	SARIMA coefficients	(p, d, q) × (P, D, Q)	SARIMA coefficients
1	$(4,2,1)$ × $(0,2,1)$ $s = 4$	AR: {-0.8817,-0.4341,0.0887 ,-0.0574}, MA: {-1.0000} SAR: - , SMA: {-0.9998}	$(2,1,0)$ × $(1,0,1)$ $s = 4$	AR: {-0.9173,-0.4589,-0.4589} MA: - SAR: {0.9950}, SMA: {-0.7068}
2	$(5,2,1)$ × $(0,0,0)$ $s = 1$	AR: {-0.4488,-0.3995,-0.1161, -0.0581,0.1890} MA: {-1.0000} SAR: - , SMA: -	$(1,1,1)$ × $(0,0,0)$ $s = 1$	AR: {0.3086} MA: {-0.8448} SAR: - , SMA: -
3	$(1,2,1)$ × $(4,1,3)$ $s = 6$	AR: {-0.4747} MA: {-0.9977} SAR: {-1.0863,-0.0157,0.0752 ,-0.00006} SMA: {0.1815,-0.9461,-0.0939}	$(1,1,2)$ × $(2,0,2)$ $s = 6$	AR: {-0.7789} MA: {0.1647,-0.5927} SAR: {-0.0033,0.9965} SMA: {0.1668,-0.8018}
4	$(5,2,1)$ × $(4,1,0)$ $s = 3$	AR: {-0.5809,-0.4194,-0.2285, -0.1560,-0.0595} MA: {-0.5489} SAR: {-0.8618,-0.8059,-0.8104, 0.1052}, SMA: -	$(0,1,1)$ × $(2,0,2)$ $s = 3$	AR: - MA: {-0.4771} SAR: {0.0065,0.9931} SMA: {-0.0214,-0.9366}
5	$(0,2,1)$ × $(0,0,0)$ $s = 1$	AR: - , MA: {-1.0000} SAR: - , SMA: -	$(0,1,0)$ × $(0,0,0)$ $s = 1$	AR: - , MA: - SAR: - , SMA: -

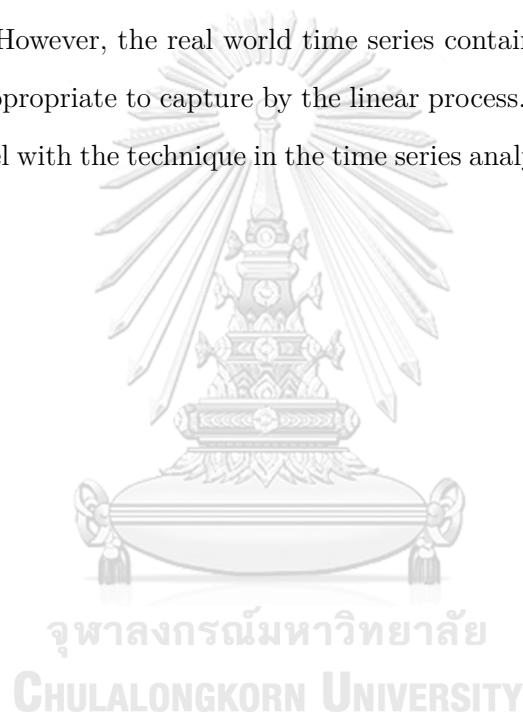
Table 5.8: The coefficients and the order from the real world time series number 0-5

	Deep learning		auto-ARIMA	
#	(p, d, q) × (P, D, Q)	SARIMA coefficients	(p, d, q) × (P, D, Q)	SARIMA coefficients
6	$(5,1,4)$ × $(0,0,0)$ $s = 1$	AR: $\{-0.4110,-0.0211,0.0213,$ $-0.1126,0.1141\}$ MA: $\{-0.4507,-0.3005,-0.1867,$ $-0.1919\}$, SAR: - , SMA: -	$(5,1,1)$ × $(0,0,0)$ $s = 1$	AR: $\{0.5010,-0.0218,-0.011,$ $-0.1489,-0.2158\}$ MA: $\{-0.9425\}$ SAR: - , SMA: -
7	$(5,1,2)$ × $(1,2,1)$ $s = 7$	AR: $\{-0.0905,-0.5799,-0.1475,$ $-0.2829,0.0994\}$ MA: $\{-0.6197,2.3119\}$ SAR: $\{-0.4614\}$, SMA: $\{-1.0111\}$	$(2,1,1)$ × $(2,0,1)$ $s = 7$	AR: $\{0.5746,0.0301\}$ MA: $\{-0.9979\}$ SAR: $\{0.9605,0.0362\}$ SMA: $\{-0.8332\}$
8	$(4,2,1)$ × $(0,2,1)$ $s = 4$	AR: $\{-0.2217,-0.0910,0.0002,$ $-0.4115\}$ MA: $\{-1.0000\}$ SAR: - , SMA: $\{-1.0000\}$	$(3,2,0)$ × $(4,0,1)$ $s = 4$	AR: $\{-0.9793-0.9937-0.9490\}$ MA: - SAR: $\{0.3632,0.0755,0.3519,$ $0.1696\}$, SMA: $\{-0.8986\}$
9	$(2,2,1)$ × $(0,2,1)$ $s = 4$	AR: $\{-1.3556,-0.7218\}$ MA: $\{0.7810\}$ SAR: - , SMA: $\{-1.0000\}$	$(3,2,0)$ × $(3,0,1)$ $s = 4$	AR: $\{-0.8397,-0.9932,-0.8266\}$ MA: - SAR: $\{0.4610,0.4207,0.0931\}$ SMA: $\{-0.9785\}$
10	$(7,1,7)$ × $(2,1,1)$ $s = 11$	AR: $\{0.2137,-0.1218,-0.7223,$ $0.4212,0.2142,0.0312$ $, -0.0023\}$ MA: $\{-0.3271,-0.2291,-0.6113$ $, -0.1232,0.0142,0.1181,-0.5221\}$ SAR: $\{0.1237,0.5711\}$ SMA: $\{0.1341\}$	$(0,1,1)$ × $(3,1,1)$ $s = 11$	AR: - , MA: $\{-0.9871\}$ SAR: $\{0.3720,-0.1256,-0.0239\}$ SMA: $\{-0.8789\}$

Table 5.9: The coefficients and the order from the real world time series number 6-10

5.2.3 The conclusion of the enhancing SARIMA forecasting model via the deep learning algorithm

The experimental results are divided to the synthetic time series data from to the ARIMA process and the real world time series data from the “fpp2” package in R. The proposed algorithm can apply to the synthetic time series data effectively via MAE, RMSE, MAPE and SMAPE and the Ljung-Box test. In addition, it can forecast the future values closer to the actual values than the auto-ARIMA model on the real world datasets. The appropriate SARIMA order helps to improve the performance of forecasting the future values. However, the real world time series contains non-linear characteristics that may not be appropriate to capture by the linear process. The next idea is to use the deep learning model with the technique in the time series analysis to capture non-constant variance.



CHAPTER VI

METHODOLOGY FOR APPLYING TO THE NONLINEAR TIME SERIES MODEL

To enhance the ability to capture non-linear characteristics of the time series data, the APEA model and the enhancing SARIMA forecasting model via the deep learning algorithm are adapted to the ARCH model to apply to the financial time series data. This algorithm is called the automatic ARCH forecasting via deep learning algorithm.

6.1 Constructing the automatic ARCH forecasting via deep learning algorithm

After obtaining the algorithms for forecasting the time series data based on the linear process, these algorithms are now adapted to build the forecasting time series model for the financial time series. The algorithm will build the ARCH model from the time series data.

The ARCH model is the basic model for applying with the time series data having the non-constant variance. The details of the ARCH model are described in Section 2.2 of Chapter 2. From the ARCH process, error term ϵ_t is the residual from the fitted SARIMA model. The new algorithm will use the enhancing SARIMA forecasting model via the deep learning algorithm to fit the SARIMA model and use the APEA model for the square residuals of the previous fitted SARIMA model. The process of the algorithm is shown in Figure 6.1 and Algorithm 5.

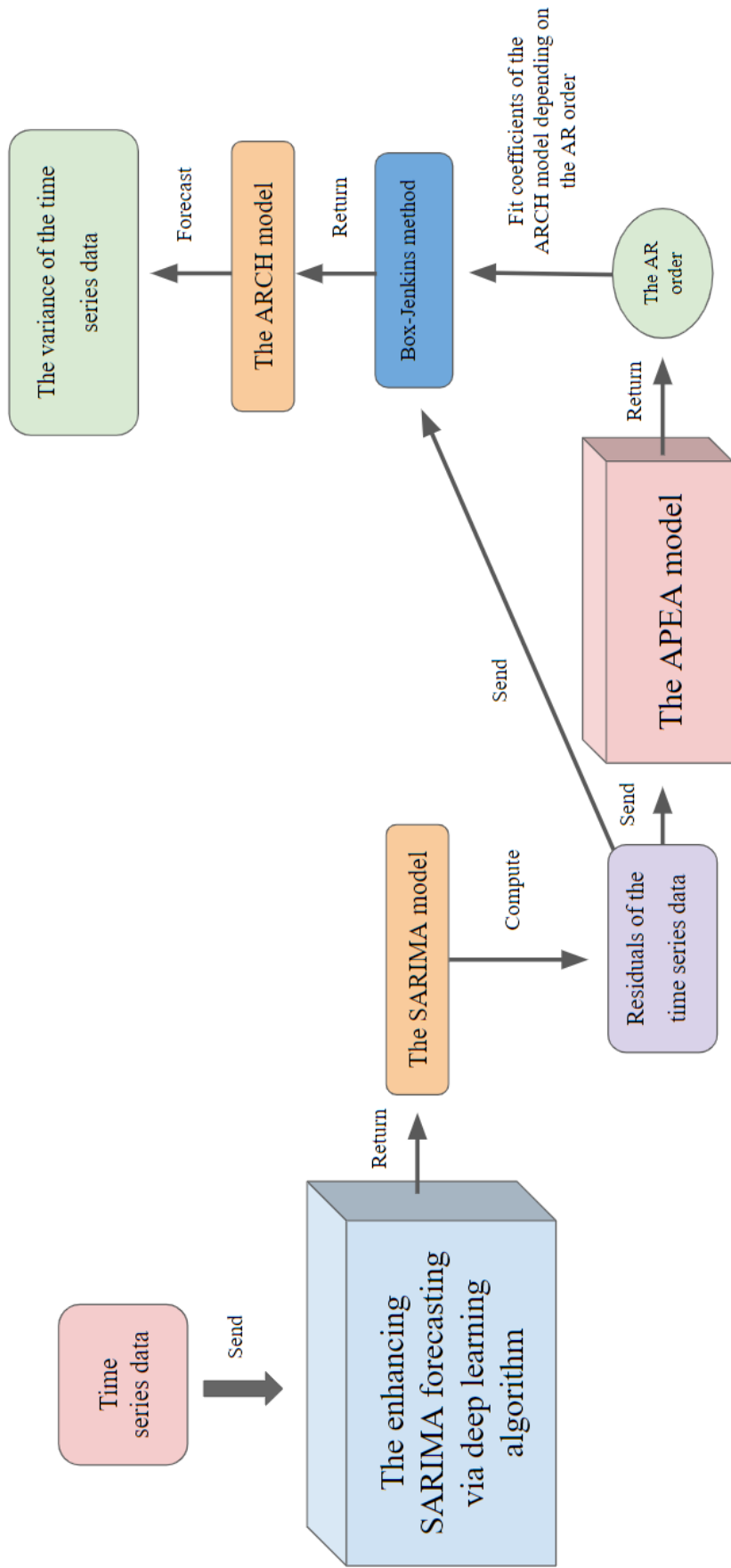


Figure 6.1: The automatic ARCH forecasting via deep learning algorithm

Algorithm 5 : Pseudo code for the automatic ARCH forecasting via deep learning algorithm

Input: time series data

- 1: Submit the time series data to the enhancing SARIMA forecasting model via the deep learning algorithm
 - 2: **Return:** the SARIMA model
 - 3: Generate the residuals from the SARIMA model
 - 4: Submit the residuals to the APEA model
 - 5: **Return:** the ARCH order
 - 6: Submit the ARCH order to the Box-Jenkins method
 - 7: **Return:** the ARCH model
-

The automatic ARCH forecasting via deep learning algorithm starts by sending the time series data to the enhancing SARIMA forecasting model via the deep learning algorithm to build the SARIMA model. Then, it is used for calculating the residuals from the original time series data. Next, the square residuals are sent to the APEA model to predict the AR order of the ARCH order. Finally, the ARCH model is built by the Box-Jenkins method. The complete ARCH model is used to forecast the original time series data.

6.2 Experimental results of the automatic ARCH forecasting via deep learning algorithm by applying to the financial time series data

Three financial time series data are used to compare with the auto-ARCH model based on AIC. The process of auto-ARCH model based on AIC is similar to the auto-ARIMA model but auto-ARCH model uses the ARCH model to fit the coefficients and compute the AIC instead of using the ARIMA model. The S&P500 index, the NASDAQ index and the Dow Jones index are recorded daily from 2 May 2019 to 2 May 2020. The financial time series plots of their closing prices are shown in Figure 6.2, Figure 6.3 and Figure 6.4.



Figure 6.2: The daily closing price of the S&P500 index

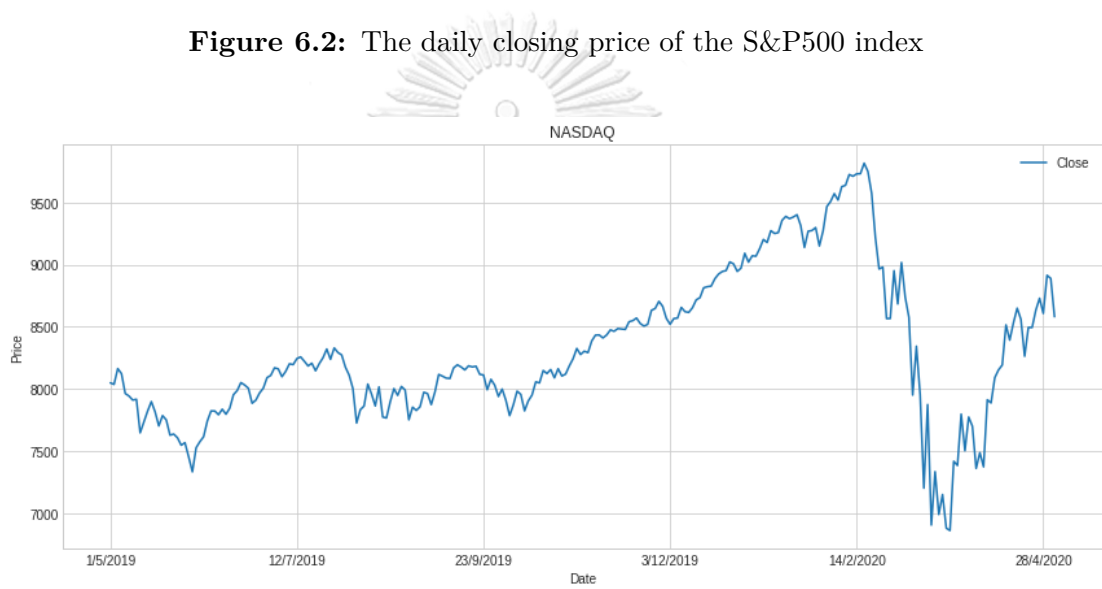


Figure 6.3: The daily closing price of the NASDAQ index

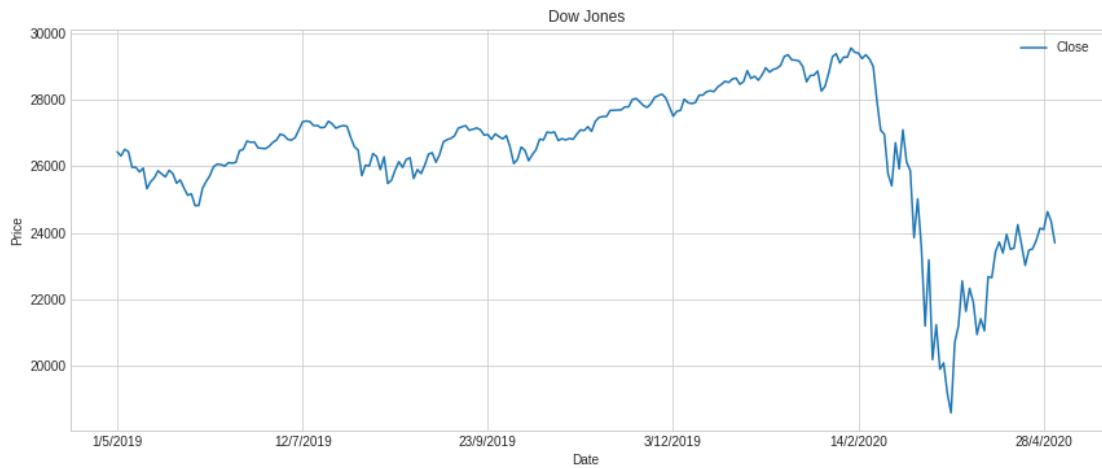


Figure 6.4: The daily closing price of the Dow Jones index

The automatic ARCH forecasting via deep learning algorithm starts by sending the financial time series data as inputs to the enhancing SARIMA forecasting model via the deep learning algorithm. The fitted SARIMA model is used to derive the residuals. The residual plot and its squared residual plot are shown in Figure 6.5, Figure 6.6 and Figure 6.7. These figures confirm that the residuals of all financial time series data are not satisfied the white noise property. They exhibit high variance in the last periods of the time series data which should be fitted with the ARCH model.

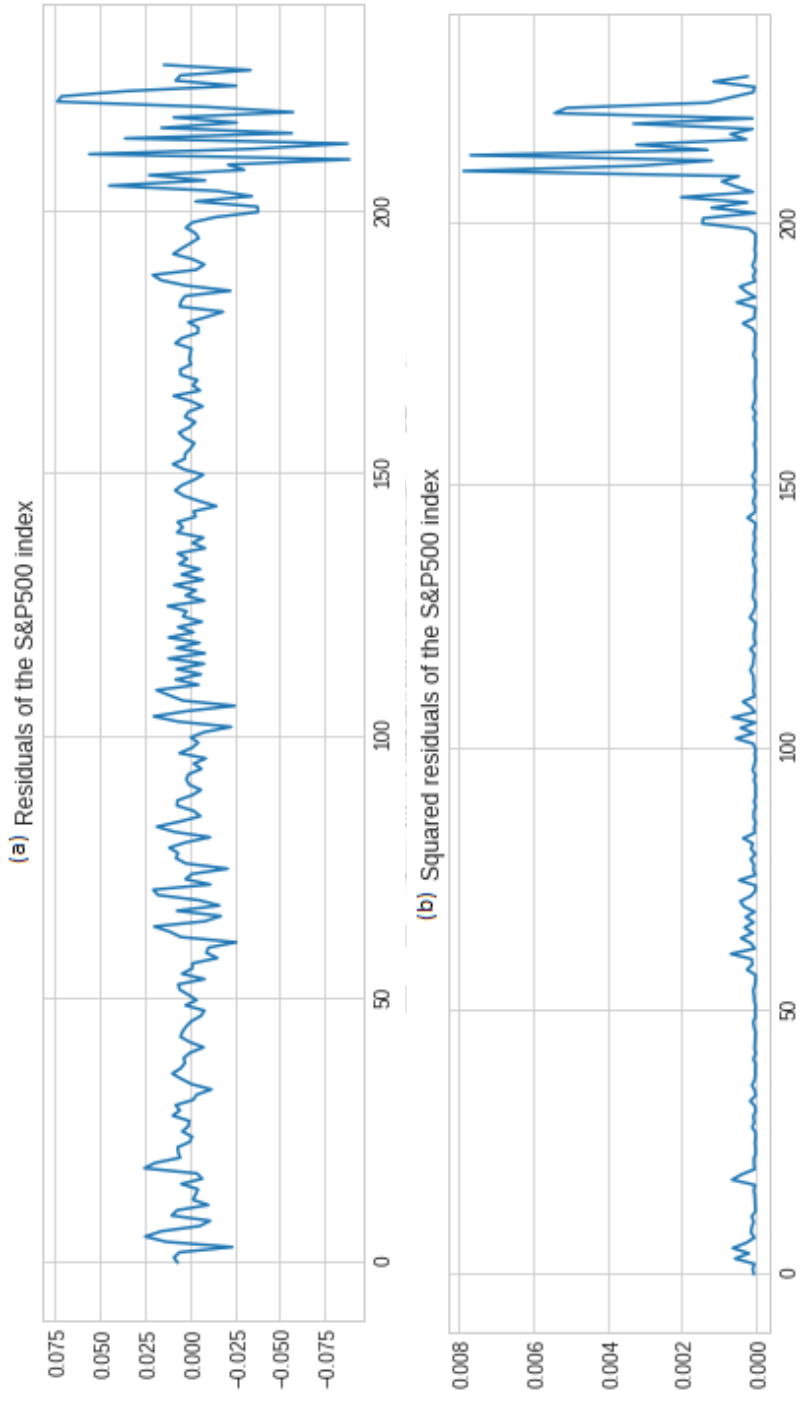


Figure 6.5: (a)Residuals and (b)Squared residuals of the S&P500 index

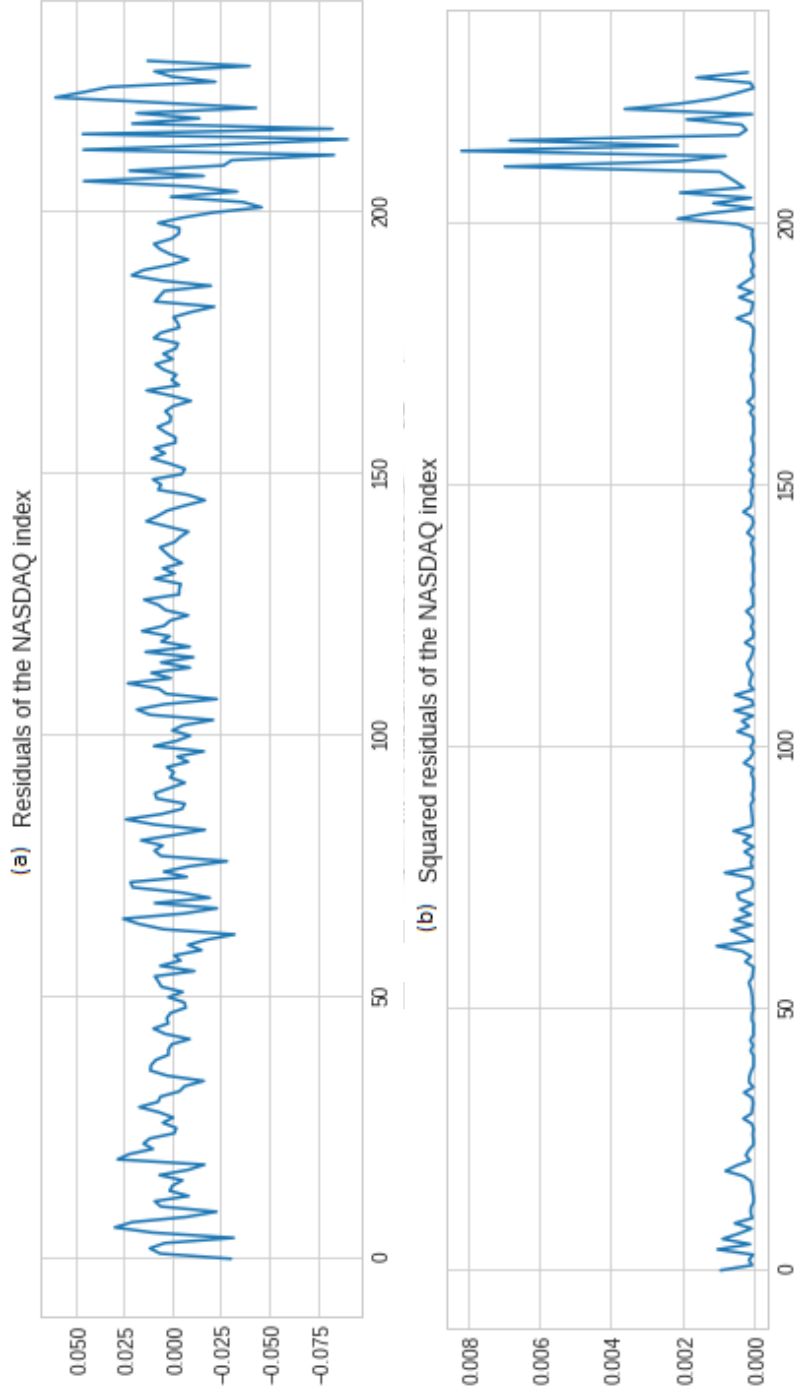


Figure 6.6: (a)Residuals and (b)Squared residuals of the NASDAQ index



Figure 6.7: (a)Residuals and (b)Squared residuals of the Dow Jones index

To fit the ARCH model with the residuals of the financial time series data, the APEA is used to identify the AR order of the residuals. From the ARCH model describing in Section 2.2.4 on Chapter 2, the residuals from the fitted SARIMA model can be written as Equation (6.1).

$$\epsilon_t = \sigma_t z_t \quad (6.1)$$

where ϵ_t is the residuals from the fitted SARIMA model, σ_t is the standard deviation of the residuals and $z_t \underset{iid}{\sim} N(0, 1)$ is a Gaussian white noise process.

Hence, the standardized residuals or the noise z_t of the fitted ARCH model can be computed by the quotient of the residuals of the SARIMA model and the standard deviation of the residuals as shown in Equation (6.2)

$$z_t = \frac{\epsilon_t}{\sigma_t} \quad (6.2)$$

which σ_t can be computed by $\sigma_t^2 = \alpha_0 + \alpha_1 \epsilon_{t-1}^2 + \dots + \alpha_q \epsilon_{t-q}^2$.

To measure the performance of the ARCH model, the standardized residuals z_t of the ARCH model must satisfy the white noise property. The ACF plot and the Ljung-Box test are used to test the performance.

The standardized residuals of all financial time series data are shown in Figure 6.8, Figure 6.9 and Figure 6.10. The experimental results show that the standardized residuals of both methods of three financial time series data is around 0 which the standardized residuals from the automatic ARCH forecasting via deep learning algorithm is quite smaller than the auto-ARCH model based on AIC in the case of S&P500 and NASDAQ. Nevertheless, in the last period of the standardized residuals exhibit high variance in both methods. For the Dow Jones index, the result of both methods is quite similar.

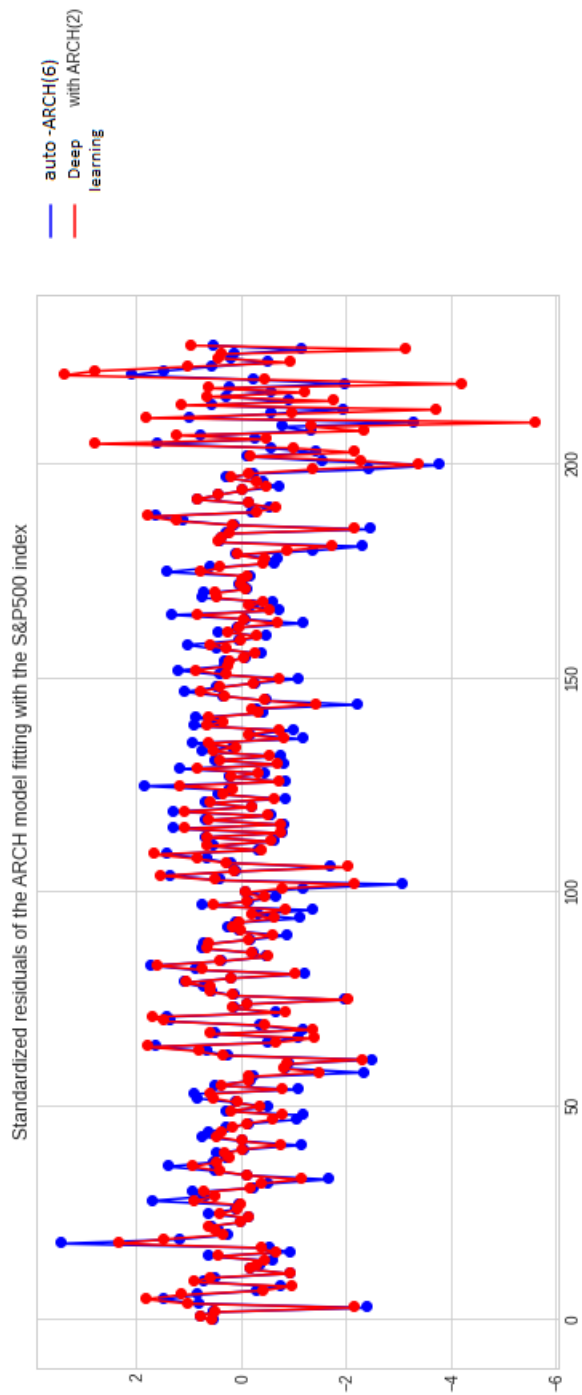


Figure 6.8: The Standardized residuals of the ARCH model fitting with the S&P500 index

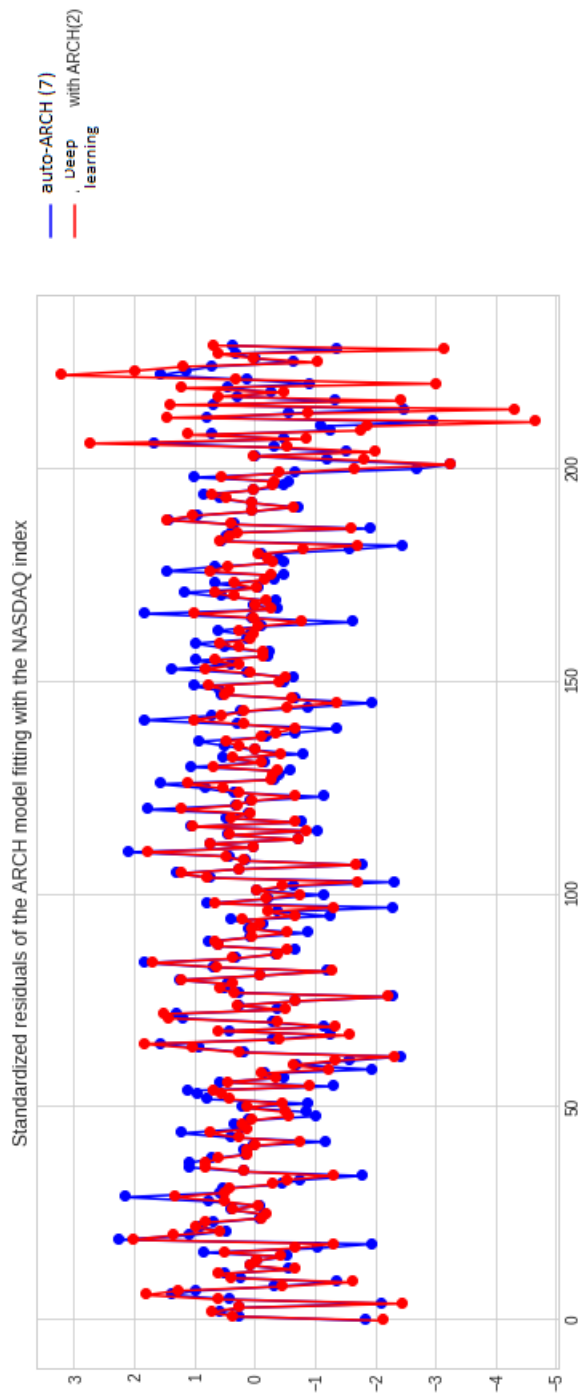


Figure 6.9: The Standardized residuals of the ARCH model fitting with the NASDAQ index

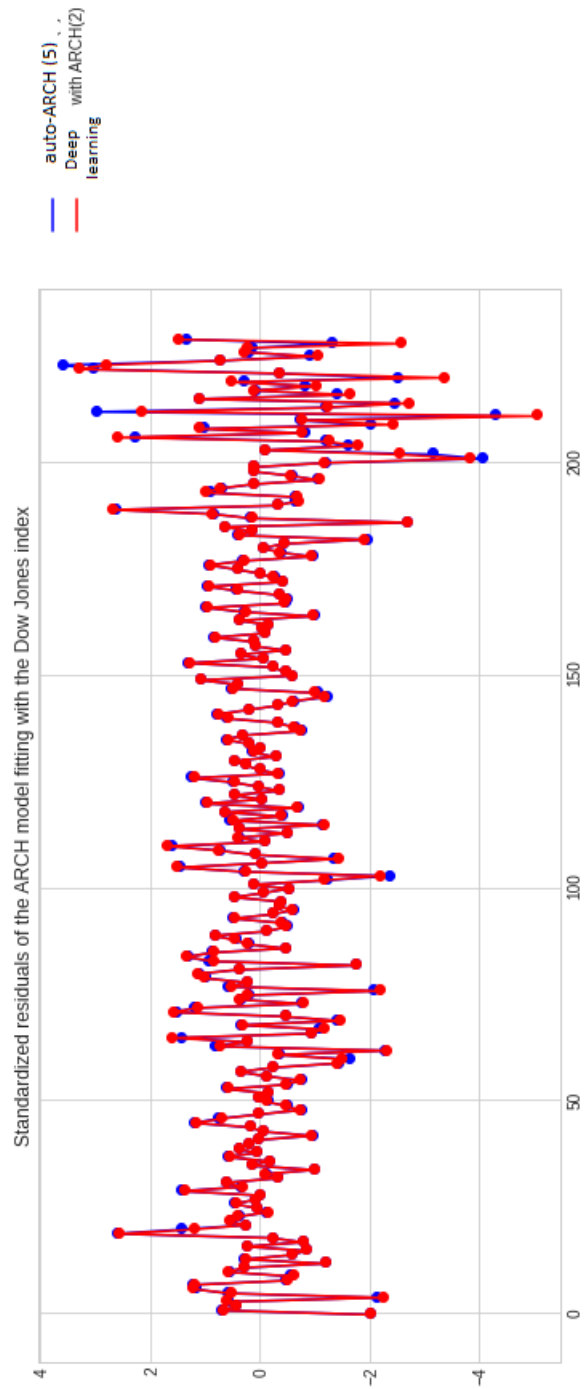
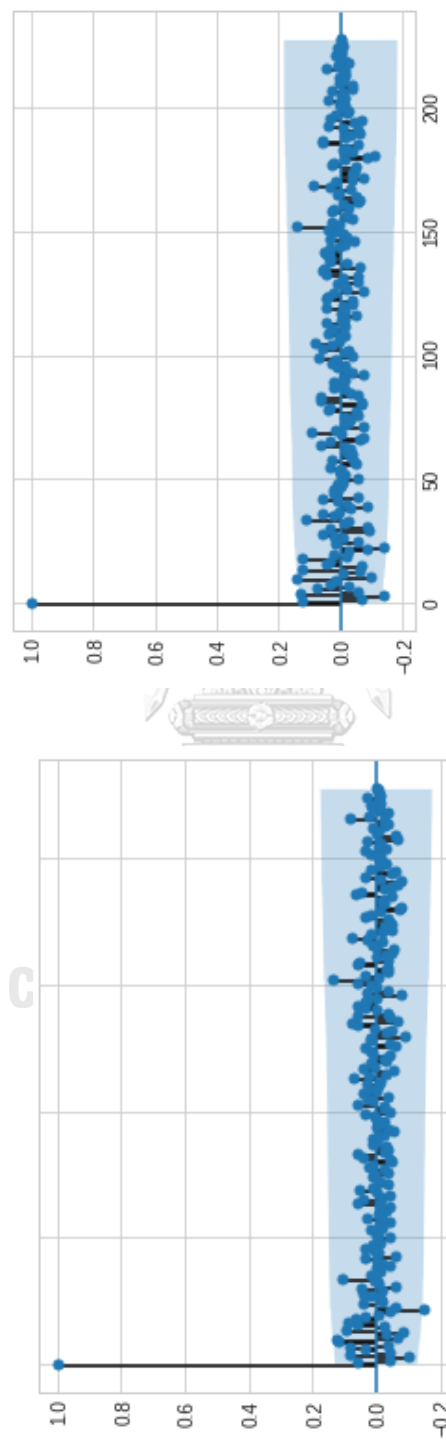


Figure 6.10: The Standardized residuals of the ARCH model fitting with the Dow Jones index

To confirm the performance of forecasting the variance by the ARCH model, the ACF plots and the Ljung-Box plots of both methods are shown in Figure 6.11 - Figure 6.16. The ACF plots of both methods show that the residuals of the ARCH models from both methods are independent because the ACF values of both methods are quite small and is in the confidence band. Nevertheless, from the Ljung-Box plots demonstrating in Figure 6.14, Figure 6.15 and Figure 6.16, the automatic ARCH forecasting via deep learning algorithm passes the Ljung-Box test in the case of S&P500 and NASDAQ whereas the auto-ARCH model based on AIC fails it. For the case of Dow Jones, both methods give the same performance.

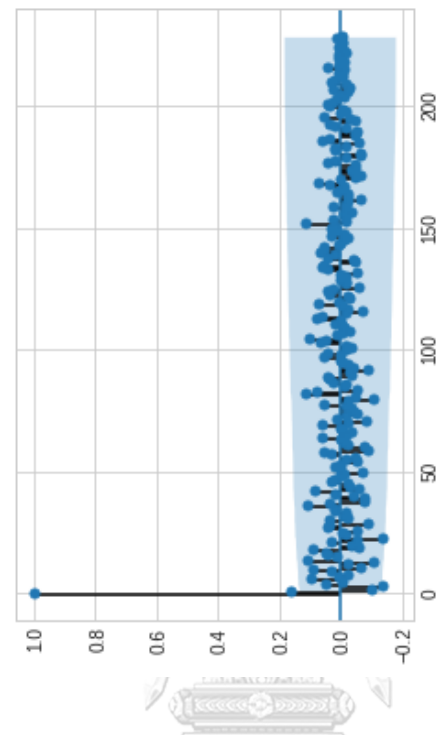




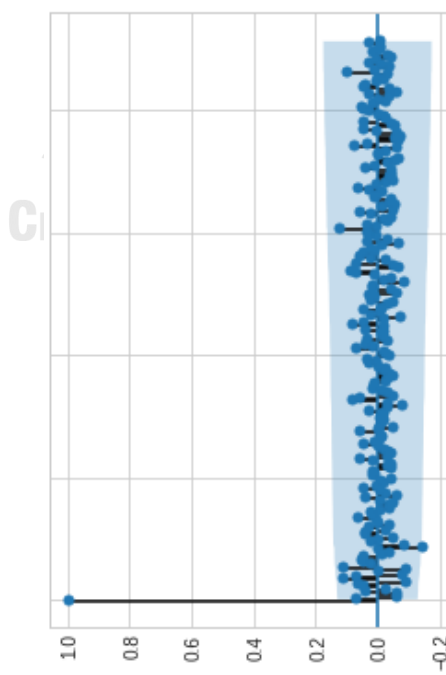
(a) The ACF plot of the deep learning algorithm

(b) The ACF plot of the auto-ARCH model based on AIC

Figure 6.11: The ACF plot in the case of S&P500

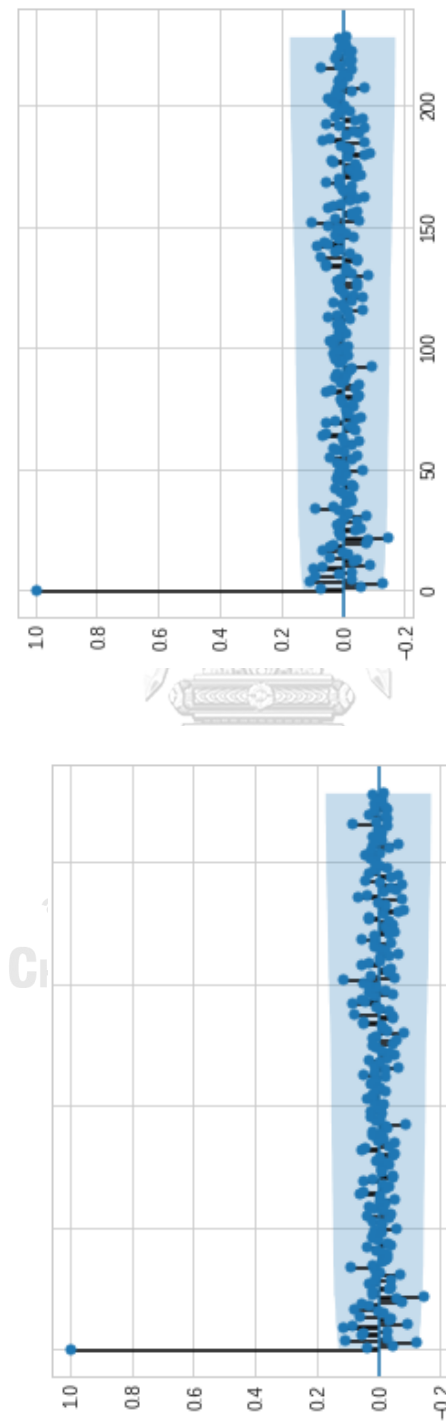


(b) The ACF plot of the auto-ARCH model based on AIC



(a) The ACF plot of the deep learning algorithm

Figure 6.12: The ACF plot in the case of NASDAQ



(a) The ACF plot of the deep learning algorithm

(b) The ACF plot of the auto-ARCH model based on AIC

Figure 6.13: The ACF plot in the case of Dow Jones

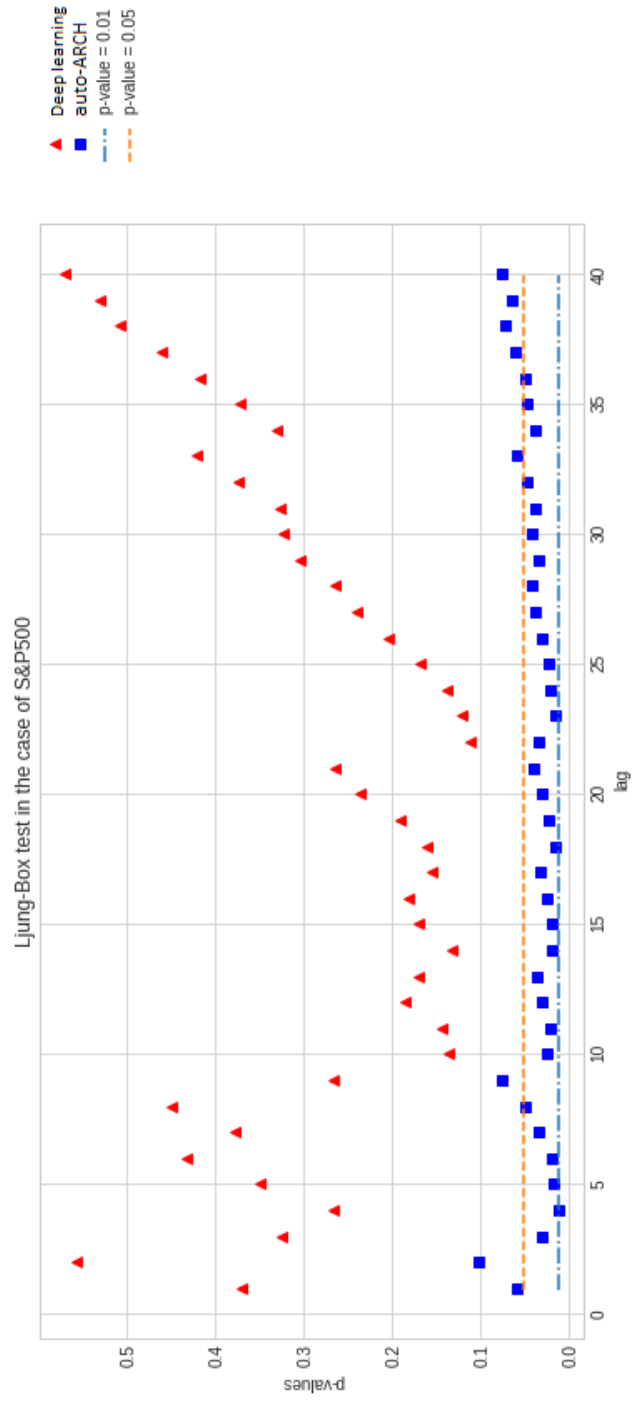


Figure 6.14: Ljung-Box test in the case of S&P500

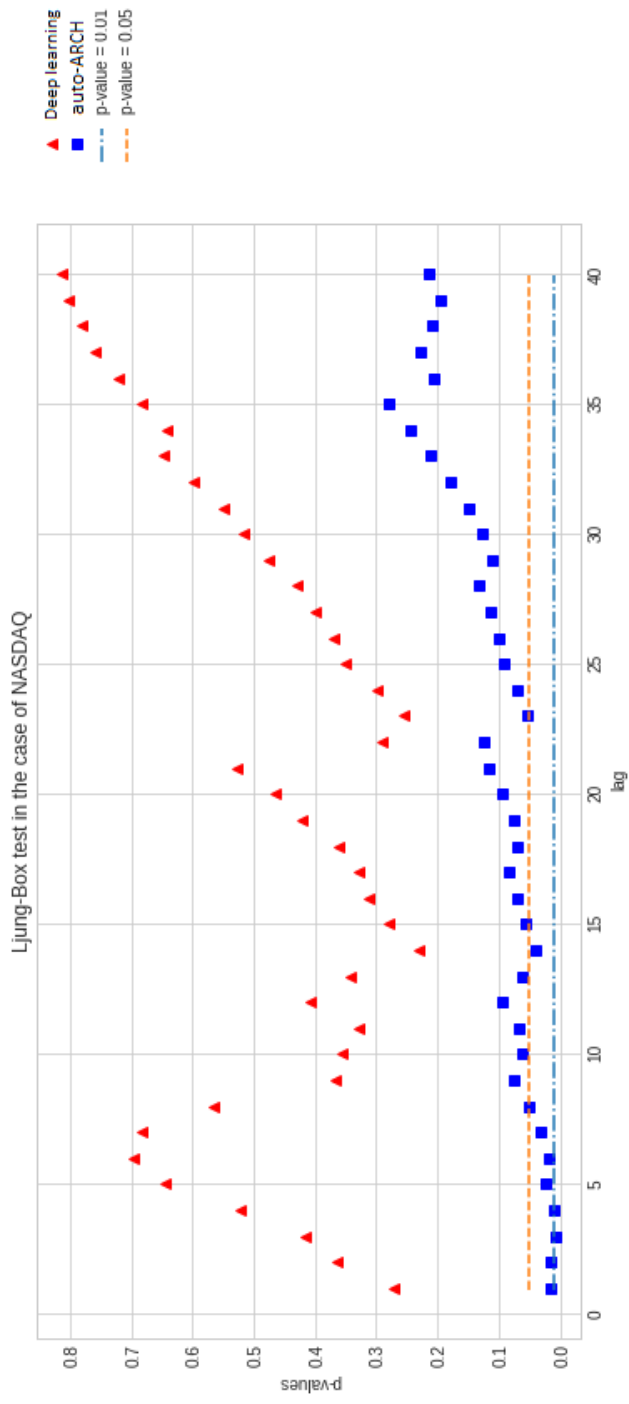


Figure 6.15: Ljung-Box test in the case of NASDAQ

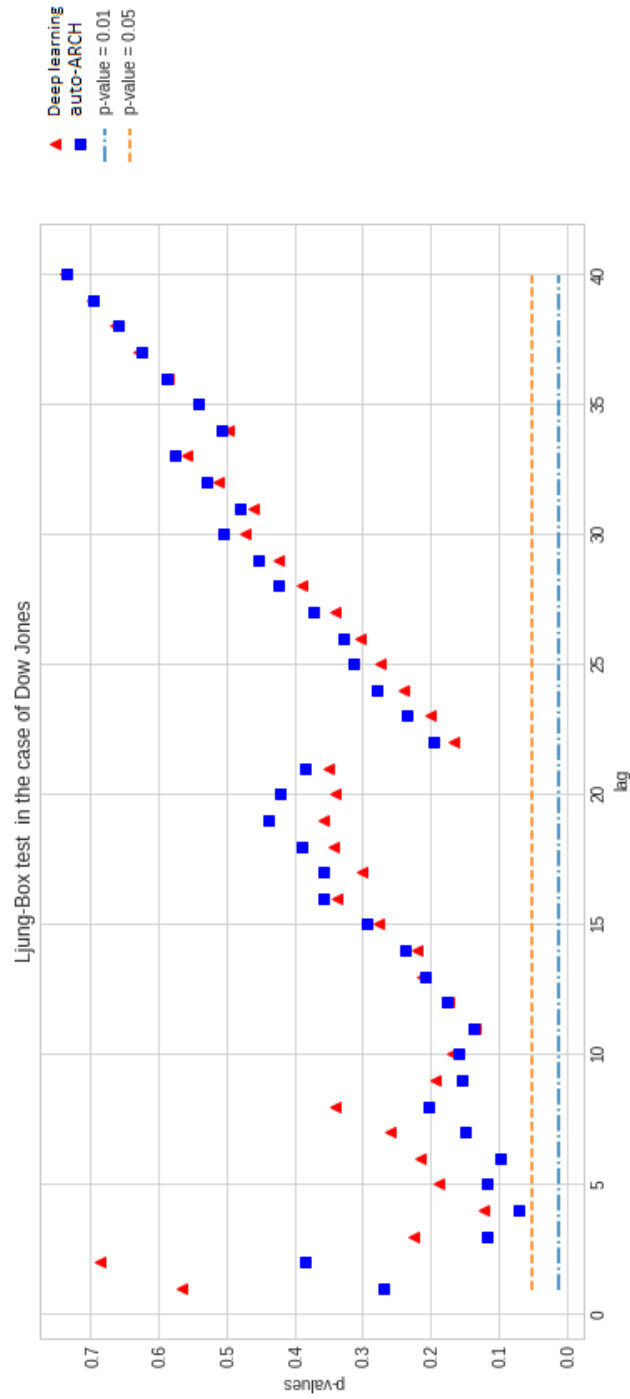


Figure 6.16: Ljung-Box test in the case of Dow Jones

6.3 The conclusion of the automatic ARCH forecasting via deep learning algorithm

The new algorithm is build based on applying the enhancing SARIMA forecasting model via the deep learning algorithm to build the SARIMA model and the APEA model is used to identify the ARCH order from the residuals occurring from the fitted SARIMA model. The proposed algorithm is applied to the three financial time series data including the S&P index, the NASDAQ index and the Dow Jones index. The experimental results show that the APEA model can capture the residuals by ARCH better than the auto-ARCH model based on AIC. The performance of the proposed algorithm is confirmed via the Ljung-Box test which the proposed model can pass the Ljung-Box test for all three financial time series data whereas the auto-ARCH model based on AIC succeeds only the Dow Jones index.



CHAPTER VII

CONCLUSIONS AND DISCUSSIONS

This dissertation proposes five new deep learning models. The first model is called the SID model which composes of the pq -SID model to identify the ARMA order from 0-5 and the d -SID model to identify the differencing order from 0-2. The SID model is constructed by training ACF, PACF and series images. It is improved by the ResNet architecture utilizing skip connections of convolutional neural networks layers. The second model is called the SIRO model. Moreover, ESACF is used as the input of the third model which the APEA model is introduced. It is spitting into the pq -APEA model and the d -APEA model. The pq -APEA model is constructed using ACF, PACF, ESACF and the differencing time series images to identify the ARMA order from 0-7 while the d -APEA model is constructed using ACF and PACF taking differencing from 0 to 3 to identify the differencing order from 0-3. Then, the fourth model is extended from the APEA model to identify the SARIMA order, called the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model, using the technique of time series aggregation and spectral analysis to find the seasonal length of the time series data. To build the SARIMA model, the outputs of the model are used for constructing the enhancing SARIMA forecasting model via the deep learning algorithm by applying the Box-jenkins method to fit the SARIMA coefficients and forecast future values. The fifth model is constructed for the financial time series data having the non-constant variance by combining the enhancing SARIMA forecasting model via the deep learning algorithm to build the SARIMA model and the pq -APEA to identify the ARCH order of the residuals.

From the visualization of the convolutional neural network filters in Section 4.4, the APEA model can extract the features of various inputs effectively. PACF and ACF are recognized in the middle of the images and try to capture the lags which is hard for humans to recognize visually. For the background of the images, ESACF and differencing time

series images are used to fulfill in the background images for improving the performance of identifying the ARIMA order by CNN. The visualization of the CNN filters shows that the features of ESACF and differencing time series images can be extracted.

In the experiments in Section 4.1 and Section 4.2, the SID model and the SIRO model outperforms the auto-ARIMA model and the previous deep learning model, ResNet50, in terms of the precision, the recall and the f1-score. It is suggested that the proposed models can extract the features of PACF, ACF or series images which are laborious for analysts for identifying the ARIMA order. Moreover, the results indicate that changing the time series to PACF images, ACF images or differencing time series images can predict orders better than using time series as direct input like ResNet50. In addition, these results suggest that the identifying ARIMA order using PACF, ACF or differencing is still more efficient than using the auto-ARIMA model because these tools can represent the correlation of time series data whereas the auto-ARIMA model does not use it.

In the experiment in Section 4.3, the APEA model for identifying the ARIMA order from 0-7 gives the better performance than the other models consisting of ResNet50 and the auto-ARIMA model. Especially, the APEA model using 3 and 4 channels of ESACF, PACF, ACF, and differencing time series images can identify the ARIMA order accurately and give the best scores of the precisions, the recalls and the f1-scores. It is suggested that identifying the ARIMA order by ESACF, PACF, ACF and differencing time series images is more effective than the auto-ARIMA model and the ResNet50 using the pure time series as inputs. ESACF helps to identify the ARIMA order better than using only ACF, PACF and series images. Moreover, the APEA model can extract the features in ACF, PACF, ESACF or differencing series images which are difficult for analysts. In addition, the experiment suggests that the use of time series without converting to ACF, PACF, ESACF or taking differencing cannot identify the ARIMA order effectively which may be caused by the time series being disturbed by some noises. For the experimental results of the ACF-PACF-ESACF convolutional neural network seasonal ARIMA order identification model to identify the SARIMA order as demonstrates in Section 5.1, the model can identify the SARIMA order better than the auto-ARIMA model when considering the

precisions, the recalls and the f1-scores.

Then, the enhancing SARIMA forecasting model via the deep learning algorithm is applied with the various time series data including the synthetic time series data and the real world time series data. The first experiment for the proposed algorithm is to apply with the synthetic time series data according to stationarity and invertibility. The results show that the error from forecasting of the proposed algorithm passes the Ljung-Box test and provides the forecasted values close to the actual values whereas the auto-arma model cannot fit the model in some case of the synthetic time series data. To ensure the performance of the enhancing SARIMA forecasting model via the deep learning algorithm, the real world time series data is applied. The results show that the proposed algorithm can apply to the real world datasets like the auto-arma model, though there are some real world time series data which fail the Ljung-Box test. This may be caused by the real world time series being inconsistent with the ARIMA process.

Finally, the automatic ARCH forecasting via deep learning algorithm is applied to the financial time series data including the S&P500 index, the NASDAQ index and the Dow Jones index. The results shows that the automatic ARCH forecasting via deep learning algorithm can better forecast the variance of the financial time series than the auto-ARCH model based on AIC. Moreover, the performance of the the automatic ARCH forecasting via deep learning algorithm is also confirmed by the success of the Ljung-Box test whereas the auto-ARCH model based on AIC fails the test.

For the future work, forecasting the financial time series data using only the ARCH model may not be able to get the best prediction of financial time series. Therefore, the algorithm in this research may be able to extend the ARCH model to the GARCH model in order to more accurately and efficiently predict the financial time series data.

REFERENCES

- [1] C. W. Granger and S.-H. Poon, "Forecasting financial market volatility: A review," *Available at SSRN 268866*, 2001.
- [2] M. P. Clements, P. H. Franses, and N. R. Swanson, "Forecasting economic and financial time-series with non-linear models," *International Journal of Forecasting*, vol. 20, no. 2, pp. 169–183, 2004.
- [3] S. J. Taylor, *Modelling financial time series*. world scientific, 2008.
- [4] C. Cheng, A. Sa-Ngasongsong, O. Beyca, T. Le, H. Yang, Z. Kong, and S. T. Bukkapatnam, "Time series forecasting for nonlinear and non-stationary processes: A review and comparative study," *Iie Transactions*, vol. 47, no. 10, pp. 1053–1071, 2015.
- [5] D. Tjøstheim, "Non-linear time series: a selective review," *Scandinavian Journal of Statistics*, pp. 97–130, 1994.
- [6] G. Kirchgässner and J. Wolters, *Introduction to modern time series analysis*. Springer Science & Business Media, 2007.
- [7] W. Härdle, H. Lütkepohl, and R. Chen, "A review of nonparametric time series analysis," *International Statistical Review*, vol. 65, no. 1, pp. 49–72, 1997.
- [8] M. S. Bartlett, "Periodogram analysis and continuous spectra," *Biometrika*, vol. 37, no. 1/2, pp. 1–16, 1950.
- [9] T. T. Tchraikian, B. Basu, and M. O'Mahony, "Real-time traffic flow forecasting using spectral analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 519–526, 2011.
- [10] K. Afshar and N. Bigdeli, "Data analysis and short term load forecasting in iran electricity market using singular spectral analysis (ssa)," *Energy*, vol. 36, no. 5, pp. 2620–2627, 2011.

- [11] K. Du, Y. Zhao, and J. Lei, “The incorrect usage of singular spectral analysis and discrete wavelet transform in hybrid models to predict hydrological time series,” *Journal of Hydrology*, vol. 552, pp. 44–51, 2017.
- [12] L. Grafakos, *Classical fourier analysis*, vol. 2. Springer, 2008.
- [13] G. E. Box and G. M. Jenkins, “Time series analysis: Forecasting and control holden-day,” *San Francisco*, p. 498, 1970.
- [14] R. F. Engle, “Estimates of the variance of us inflation based upon the arch model,” *Journal of Money, Credit and Banking*, vol. 15, no. 3, pp. 286–301, 1983.
- [15] T. Bollerslev, “Modelling the coherence in short-run nominal exchange rates: a multivariate generalized arch model,” *The review of economics and statistics*, pp. 498–505, 1990.
- [16] T. D. Little, *The Oxford Handbook of Quantitative Methods in Psychology: Vol. 2: Statistical Analysis*, vol. 2. Oxford University Press, 2013.
- [17] R. S. Tsay and G. C. Tiao, “Consistent estimates of autoregressive parameters and extended sample autocorrelation function for stationary and nonstationary arma models,” *Journal of the American Statistical Association*, vol. 79, no. 385, pp. 84–96, 1984.
- [18] T. Chenoweth, R. Hubata, and R. D. S. Louis, “Automatic arma identification using neural networks and the extended sample autocorrelation function: a reevaluation,” *Decision Support Systems*, vol. 29, no. 1, pp. 21–30, 2000.
- [19] H. Akaike, “Factor analysis and aic,” in *Selected papers of hirotugu akaike*, pp. 371–386, Springer, 1987.
- [20] A. A. Neath and J. E. Cavanaugh, “The bayesian information criterion: background, derivation, and applications,” *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 4, no. 2, pp. 199–203, 2012.

- [21] P. S. Karlsson, L. Behrenz, and G. Shukur, “Performances of model selection criteria when variables are ill conditioned,” *Computational Economics*, vol. 54, no. 1, pp. 77–98, 2019.
- [22] P. Chen, T. Pedersen, B. Bak-Jensen, and Z. Chen, “Arima-based time series model of stochastic wind power generation,” *IEEE transactions on power systems*, vol. 25, no. 2, pp. 667–676, 2009.
- [23] D. Ö. Faruk, “A hybrid neural network and arima model for water quality time series prediction,” *Engineering applications of artificial intelligence*, vol. 23, no. 4, pp. 586–594, 2010.
- [24] M. H. Alsharif, M. K. Younes, and J. Kim, “Time series arima model for prediction of daily and monthly average global solar radiation: the case study of seoul, south korea,” *Symmetry*, vol. 11, no. 2, p. 240, 2019.
- [25] K. C. Lee and S. B. Oh, “An intelligent approach to time series identification by a neural network-driven decision tree classifier,” *Decision Support Systems*, vol. 17, no. 3, pp. 183–197, 1996.
- [26] K. E. Al-Qawasmi, A. M. Al-Smadi, and A. Al-Hamami, “Artificial neural network-based algorithm for arma model order estimation,” in *International Conference on Networked Digital Technologies*, pp. 184–192, Springer, 2010.
- [27] M. H. Amini, A. Kargarian, and O. Karabasoglu, “Arima-based decoupled time series forecasting of electric vehicle charging demand for stochastic power system operation,” *Electric Power Systems Research*, vol. 140, pp. 378–390, 2016.
- [28] C. Guarnaccia, J. Quartieri, and C. Tepedino, “Deterministic decomposition and seasonal arima time series models applied to airport noise forecasting,” in *AIP Conference Proceedings*, vol. 1836, p. 020079, AIP Publishing LLC, 2017.
- [29] R. Fukuoka, H. Suzuki, T. Kitajima, A. Kuwahara, and T. Yasuno, “Wind speed prediction model using lstm and 1d-cnn,” *Journal of Signal Processing*, vol. 22, no. 4, pp. 207–210, 2018.

- [30] T. Kim and H. Y. Kim, "Forecasting stock prices with a feature fusion lstm-cnn model using different representations of the same data," *PloS one*, vol. 14, no. 2, 2019.
- [31] W. H. Tang and A. Röllin, "Model identification for arma time series through convolutional neural networks," *arXiv preprint arXiv:1804.04299*, 2018.
- [32] R. J. Hyndman, Y. Khandakar, *et al.*, *Automatic time series for forecasting: the forecast package for R*, 2007.
- [33] L. Deng, D. Yu, *et al.*, "Deep learning: methods and applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, 2014.
- [34] N. Aloysius and M. Geetha, "A review on deep convolutional neural networks," in *2017 International Conference on Communication and Signal Processing (ICCSP)*, pp. 0588–0592, IEEE, 2017.
- [35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.
- [36] A. S. Modi, "Review article on deep learning approaches," in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 1635–1639, IEEE, 2018.
- [37] H.-J. Yoo, "Deep convolution neural networks in computer vision: a review," *IEIE Transactions on Smart Processing & Computing*, vol. 4, no. 1, pp. 35–43, 2015.
- [38] C.-L. Liu, W.-H. Hsaio, and Y.-C. Tu, "Time series classification with multivariate convolutional neural network," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 6, pp. 4788–4797, 2018.
- [39] I. Koprinska, D. Wu, and Z. Wang, "Convolutional neural networks for energy time series forecasting," in *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2018.

- [40] B. Wang, Y. Lei, T. Yan, N. Li, and L. Guo, “Recurrent convolutional neural network: A new framework for remaining useful life prediction of machinery,” *Neurocomputing*, vol. 379, pp. 117–129, 2020.
- [41] R. J. Hyndman, *CRAN task view: Time series analysis*, 2019.



BIOGRAPHY

Name	Mr. Paisit Khanarsa
Date of Birth	2 December 1992
Place of Birth	Bangkok, Thailand
Education	B.Sc. (Mathematics) (First-class honour), Kasertsart University, 2014 M.Sc. (Applied Mathematics and Computational Science), Chulalongkorn University, 2016
Scholarships	Science Achievement Scholarship of Thailand (SAST)

Publications

- Paisit Khanarsa, Arthorn Luangsodsai, and Krung Sinapiromsaran, “Self-Identification Deep Learning ARIMA”, Journal of Physics: Conference Series, Vol.1564, No.1, IOP Publishing, 2020.
- Paisit Khanarsa, Arthorn Luangsodsai, and Krung Sinapiromsaran, “Self-Identification ResNet-ARIMA Forecasting Model”, WSEAS Transactions on Systems and Control, Vol. 15, No.21, pp.196-211, 2020.
- Paisit Khanarsa and Krung Sinapiromsaran, “Automatic SARIMA Order Identification Convolutional Neural Network”, International Journal of Machine Learning and Computing, Vol.10, No.5, pp.685-691, 2020.
- Paisit Khanarsa, and Krung Sinapiromsaran, “Multiple ARIMA subsequences aggregate time series model to forecast cash in ATM”, 2017 9th International Conference on Knowledge and Smart Technology (KST), IEEE, 2017.