

**COMBINING DEEP LEARNING AND  
TECHNICAL ANALYSIS FOR  
MULTIRESOLUTION FORECASTING OF  
FINANCIAL STOCK MARKETS**

BY

**KHALED AHMED LUTF AL-THELAYA**

A Thesis Presented to the  
DEANSHIP OF GRADUATE STUDIES

**KING FAHD UNIVERSITY OF PETROLEUM & MINERALS**

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the  
Requirements for the Degree of

**MASTER OF SCIENCE**

In

**COMPUTER SCIENCE**

**DECEMBER 2018**





KING FAHD UNIVERSITY OF PETROLEUM & MINERALS  
DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

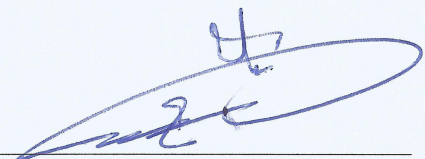
This thesis, written by **KHALED AHMED LUTF AL-THELAYA** under the direction of his thesis adviser and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN COMPUTER SCIENCE**.

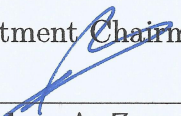
Thesis Committee

  
Dr. Salahadin Adem Mohammed (Adviser)

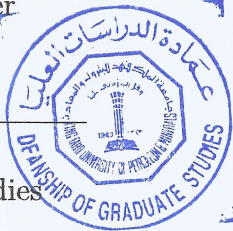
 25/12/18  
Prof. El-Sayed Mohamed El-Alfy (Member)

  
Dr. Wasfi Ghassan Al-Khatib (Member)

  
Dr. Khalid A. Al-Jasser  
Department Chairman

  
Dr. Salam A. Zummo  
Dean of Graduate Studies

31/12/18  
Date



©Khaled Ahmed Lutf Al-Thelaya  
2018



*This thesis could have never been completed without the help, care, and support of my dearest mother, beloved father, darling wife, lovely sisters, and kind brothers. I dedicate this work to them and to my two lovely boys. I also dedicate my thesis work to my respectful kind teachers and friends whose help, guidance, and support made obstacles and difficulties vanish.*

# ACKNOWLEDGMENTS

All praises and thanks are due to the almighty Allah for bestowing me with health, knowledge and urge to complete this work. I would like to express my sincere gratitude to my advisor Dr. Salahadin A. Mohammed for the continuous support and guidance throughout my thesis work and research, and for his patience, motivation, and encouragement. I would also like to express my gratitude towards Prof. El-Sayed M. El-Alfy who has given me a lot of his time and knowledge and never hesitate to help me with his knowledge and experience. Without his guidance and instructions this work would have never been. I would like to thank Dr. Wasfi G. Al-Khatib for his valuable cooperation and constructive comments throughout my thesis. He has dedicated valuable time to provide useful feedback to improve the quality of this work. Great thanks to King Fahd University of Petroleum and Minerals, especially the department of information and computer science for providing facilities and support to make this work come true. I would also like to thank my parents and friends for their precious advice and encouragement. Last but not least, I am grateful to my beloved wife who had been a constant source of inspiration and support during hardship times.

# TABLE OF CONTENTS

ACKNOWLEDGMENTS	vi
LIST OF TABLES	xi
LIST OF FIGURES	xiv
LIST OF ABBREVIATIONS	xv
ABSTRACT (ENGLISH)	xv
ABSTRACT (ARABIC)	xvi
<b>CHAPTER 1 INTRODUCTION</b>	<b>1</b>
1.1 Motivation . . . . .	6
1.2 Objectives . . . . .	7
1.3 Contributions . . . . .	8
1.4 Thesis Outline . . . . .	9
<b>CHAPTER 2 RELATED WORK</b>	<b>10</b>
2.1 Linear Forecasting Models . . . . .	11
2.2 Nonlinear Forecasting Models . . . . .	12
2.3 Deep Learning Forecasting Models . . . . .	15
2.4 Multiresolution Analysis . . . . .	18
2.5 Technical Analysis and Technical Indicators . . . . .	20



<b>CHAPTER 3</b>	<b>PROPOSED METHODOLOGY</b>	<b>23</b>
3.1	Methodology Overview . . . . .	23
3.2	Technical Analysis . . . . .	25
3.3	Multiresolution Analysis . . . . .	26
3.3.1	Wavelet Transform Concept . . . . .	27
3.3.2	Empirical Wavelet Transform . . . . .	31
3.4	Deep Learning Based Models . . . . .	35
3.4.1	Deep Recurrent Neural Networks (RNNs) . . . . .	37
3.4.2	Long-Short Term Memory (LSTM) . . . . .	39
3.4.3	Gated Recurrent Unit (GRU) . . . . .	40
3.4.4	Stacked and Bidirectional RNNs . . . . .	41
<b>CHAPTER 4</b>	<b>DATA EXPLORATION AND PREPROCESSING</b>	<b>43</b>
4.1	Benchmark Datasets . . . . .	43
4.2	Data Transformation . . . . .	48
4.3	Autocorrelation Analysis . . . . .	51
4.3.1	Autocorrelation Plot (Correlogram) . . . . .	51
4.3.2	Partial Autocorrelation Plot . . . . .	52
4.3.3	Autocorrelation Analysis Results . . . . .	53
4.4	Randomness Test . . . . .	61
4.4.1	Run Test . . . . .	61
4.4.2	Results of the Run Test . . . . .	63
4.5	White Noise Tests . . . . .	64
4.5.1	Box-Pierce Test . . . . .	64
4.5.2	Ljung-Box Test . . . . .	65
4.5.3	McLeod-Li Test . . . . .	66
4.5.4	Results of White Noise Tests . . . . .	67
<b>CHAPTER 5</b>	<b>EXPERIMENTS AND RESULTS</b>	<b>68</b>
5.1	Experimental Settings . . . . .	69
5.2	Comparison Between Bidirectional and Stacked (GRU, LSTM) . . . . .	73

5.3	Empirical Wavelet Transform and BGRU . . . . .	84
5.4	Multi-Step Forecast Using Hybrid Direct Recursive Method . . . . .	93
5.5	Multiresolution Analysis and Technical Indicators . . . . .	100
5.6	Benchmarking of Proposed Forecasting Models . . . . .	106
<b>CHAPTER 6 CONCLUSIONS &amp; FUTURE WORK</b>		<b>113</b>
6.1	Conclusions . . . . .	114
6.2	Limitations . . . . .	116
6.3	Future Work . . . . .	118
<b>APPENDIX A EFFICIENT MARKET HYPOTHESIS (EMH)</b>		<b>119</b>
<b>APPENDIX B FINANCIAL TIME SERIES</b>		<b>121</b>
B.1	Financial Stock Market . . . . .	122
B.2	Technical and Fundamental Analysis . . . . .	123
B.3	Stock Market Data . . . . .	124
<b>APPENDIX C TECHNICAL INDICATORS</b>		<b>126</b>
C.1	Simple Moving Average (SMA) . . . . .	127
C.2	Exponential Moving Average (EMA) . . . . .	128
C.3	Weighted Moving Average (WMA) . . . . .	128
C.4	Momentum (MOM) . . . . .	129
C.5	Rate of Change (ROC) . . . . .	129
C.6	Relative Strength Index (RSI) . . . . .	129
C.7	Moving Average Convergence / Divergence (MACD) . . . . .	131
C.8	Stochastic Oscillator . . . . .	132
C.9	Williams R% (WR) . . . . .	132
<b>APPENDIX D ADDITIONAL EXPERIMENTS</b>		<b>133</b>
D.1	Evaluation of Bidirectional and Stacked LSTM . . . . .	134
D.2	Forecast Using Multivariate Analysis . . . . .	139
D.3	Evaluation of Empirical and Stationary Wavelet Transforms . . . . .	143

D.4 Deep Autoencoder and Technical Indicators . . . . .	150
<b>APPENDIX E THESIS PUBLICATIONS</b>	<b>158</b>
<b>REFERENCES</b>	<b>160</b>
<b>VITAE</b>	<b>180</b>



# LIST OF TABLES

2.1	List of some forecasting techniques im the literature . . . . .	15
2.2	List of some related work using deep learning techniques for financial time-series forecasting . . . . .	17
2.3	Some technical indicators adopted in other studies to perform stock market forecasting . . . . .	21
2.4	Continuous of Table 2.3 . . . . .	22
4.1	Statistical summary for benchmark datasets . . . . .	47
4.2	The results of the run test using the median as reference value . . . . .	64
4.3	Results of white noise statistics for S&P dataset . . . . .	67
5.1	Short-term forecast results of S&P test data using raw closing prices and log returns of closing prices . . . . .	88
5.2	Short-term forecast results of DJIA test data using raw closing prices and log returns of closing prices . . . . .	89
5.3	Long-term forecast results of S&P test data using raw closing prices and log returns of closing prices . . . . .	90
5.4	Long-term forecast results of DJIA test data using raw closing prices and log returns of closing prices . . . . .	91
5.5	Best trained EWT-BGRU models based on test data forecast results of S&P and DJIA for short and long-term . . . . .	93
5.6	Forecast results of the RW model using S&P and DJIA for short and long-term . . . . .	93

5.7	Results of S&P data for both BGRU and EWT-BGRU models using hybrid direct recursive multi-step forecasting for up to one month (22 working days) . . . . .	96
5.8	Results of DJIA data for both BGRU and EWT-BGRU models using hybrid direct recursive multi-step forecasting for up to one month (22 working days) . . . . .	97
5.9	Selected technical indicators and the corresponding number of studies	101
5.10	Best trained models for short-term forecasting based on performance of S&P test data with corresponding employed technical indicators . .	103
5.11	Best trained models for short-term forecasting based on performance of DJIA test data with corresponding employed technical indicators . .	103
5.12	Best trained models for long-term forecasting based on performance of S&P test data with corresponding employed technical indicators . . . .	104
5.13	Best trained models for long-term forecasting based on performance of DJIA test data with corresponding employed technical indicators . . .	104
5.14	Number of times each technical indicator is used in the top best 100 models for both S&P and DJIA datasets for short-term forecast . . . .	105
5.15	Number of times each technical indicator is used in the top best 100 models for both S&P and DJIA datasets for long-term forecast . . . . .	106
5.16	Performance of forecasting models trained on S&P dataset for each proposed approach . . . . .	108
5.17	Performance of forecasting models trained on DJIA dataset for each proposed approach . . . . .	109
5.18	Performance of forecasting models trained on TASI dataset for each proposed approach . . . . .	109
5.19	Comparison between our proposed approach and some related work using S&P dataset . . . . .	112
5.20	Comparison between proposed approach and some related work using DJIA dataset . . . . .	112
D.1	MAE, RMSE, and $R^2$ of BLSTM network results for short-term forecast	135

D.2	MAE, RMSE, and $R^2$ of SLSTM network results for short-term forecast	136
D.3	MAE, RMSE, and $R^2$ of BLSTM network results for long-term forecast	136
D.4	MAE, RMSE, and $R^2$ of SLSTM network results for long-term forecast	136
D.5	Comparison between best selected models for short-term forecast . . .	138
D.6	Comparison between best selected models for long-term forecast . . . .	139
D.7	Results of random walk model forecast for short and long-term . . . . .	139
D.8	Comparison between forecasting performance averages of several network structures using normalized data . . . . .	141
D.9	Comparison between best forecasting performance models based on testing results . . . . .	141
D.10	Results of forecasting using EWT-SLSTM and SWT-SLSTM models for Mackey-Glass time series normalized and original test data . . . . .	146
D.11	Results of short-term forecasting using EWT-SLSTM and SWT-SLSTM for S&P normalized and original test data . . . . .	148
D.12	Results of long-term forecasting using EWT-SLSTM and SWT-SLSTM for S&P normalized and original test data . . . . .	148
D.13	Performance averages for both short- and long-term forecast of S&P and DJIA datasets . . . . .	157
D.14	Best performance based on the RMSE results using test data for both short- and long-term forecast of S&P and DJIA datasets . . . . .	157



# LIST OF FIGURES

2.1	Classification of some related works according to type of forecasting models . . . . .	11
3.1	Overview of the proposed methodology . . . . .	24
3.2	Wavelet and scaling functions . . . . .	30
3.3	Decomposition block diagram of stationary wavelet transform . . . . .	30
3.4	Plot of part of the actual S&P data before performing wavelet analysis	31
3.5	Stationary wavelet decomposition for part of the S&P data . . . . .	32
3.6	Adaptive wavelet decomposition for part of the S&P data. . . . .	36
3.7	Framework of the proposed methodology . . . . .	38
3.8	Sliding window and input features reshaping to perform forecast . . .	38
3.9	Illustration of the internal architecture of the LSTM and GRU hidden units . . . . .	41
3.10	Stacked architecture used to train deep RNN . . . . .	42
3.11	Bidirectional architecture used to train deep RNN . . . . .	42
4.1	Plot of the closing price of S&P index . . . . .	46
4.2	Plot of the closing price of DJIA index . . . . .	46
4.3	Plot of the closing price of TASI index . . . . .	47
4.4	Plots of the log returns of the three datasets . . . . .	50
4.5	Autocorrelation plot of the S&P closing price . . . . .	54
4.6	Partial autocorrelation plot of the S&P closing price . . . . .	54
4.7	Autocorrelation plot of the DJIA closing price . . . . .	55
4.8	Partial autocorrelation plot of the DJIA closing price . . . . .	55

4.9	Autocorrelation plot of the TASI closing price . . . . .	56
4.10	Partial autocorrelation plot of the TASI closing price . . . . .	56
4.11	Autocorrelation plot of the log returns of S&P closing price . . . . .	57
4.12	Partial autocorrelation plot of the log returns of S&P closing price . . .	58
4.13	Autocorrelation plot of the log returns of DJIA closing price . . . . .	58
4.14	Partial autocorrelation plot of the log returns of DJIA closing price . .	59
4.15	Autocorrelation plot of the log returns of TASI closing price. . . . .	59
4.16	Partial autocorrelation plot of the log returns of TASI closing price . .	60
5.1	Illustration of short-term and long-term forecast . . . . .	69
5.2	Illustration of data split into training, validation and testing . . . . .	70
5.3	Comparison between RMSE averages of S&P test data based on different time lags for multiple time scales . . . . .	76
5.4	Comparison between RMSE averages of DJIA test data based on different time lags for multiple time scales . . . . .	76
5.5	Comparison between RMSE averages of S&P test data based on different number of neurons for multiple time scales . . . . .	77
5.6	Comparison between RMSE averages of DJIA test data based on different number of neurons for multiple time scales . . . . .	77
5.7	Comparison between RMSE averages of S&P test data based on additional different time lags for multiple time scales . . . . .	79
5.8	Comparison between RMSE averages of DJIA test data based on additional different time lags for multiple time scales . . . . .	80
5.9	Comparison between RMSE averages of S&P test data based on additional different number of neurons for multiple time scales . . . . .	80
5.10	Comparison between RMSE averages of DJIA test data based on additional different number of neurons for multiple time scales . . . . .	81
5.11	Comparison between best RMSE results of S&P test data for multiple hori- zon scales . . . . .	82
5.12	Comparison between best RMSE results of DJIA test data for multiple hori- zon scales . . . . .	83

5.13	The architecture employed to develop the forecasting approach using BGRU learning technique and EWT method (EWT-BGRU) . . . . .	84
5.14	Comparison between test data RMSE of trained models to forecast S&P short-term closing prices using raw data and log returns based on different time lags and number of neurons . . . . .	86
5.15	Comparison between test data RMSE of trained models to forecast DJIA short-term closing prices using raw data and log returns based on different time lags and number of neurons . . . . .	87
5.16	Comparison between test data RMSE of trained models to forecast S&P long-term closing prices using raw data and log returns based on different time lags and number of neurons . . . . .	87
5.17	Comparison between test data RMSE of trained models to forecast DJIA long-term closing prices using raw data and log returns based on different time lags and number of neurons . . . . .	92
5.18	Input structure of multi-step forecasting models using hybrid direct recursive method . . . . .	96
5.19	Comparison between EWT-BGRU, BGRU, and RW models based on Curves of MAE, MAPE, RMSE for multi-step forecasting of S&P data	97
5.20	Comparison between EWT-BGRU, BGRU, and RW models based on Curves of MAE, MAPE, RMSE for multi-step forecasting of DJIA data	98
5.21	Multi-step forecast curves for BGRU and EWT-BGRU models with actual prices of first sample of S&P test data . . . . .	98
5.22	Multi-step forecast curves for BGRU and EWT-BGRU models with actual prices of first sample of DJIA test data . . . . .	98
5.23	Design of decomposition and learning methodology combined with technical indicators . . . . .	101
C.1	Illustration of some technical indicators applied on S&P CNX Nifty index	127
D.1	The short-term forecast versus target curve for BLSTM, SLSTM, LSTM, and MLP Models . . . . .	137

D.2	The long-term forecast versus target curve for BLSTM, SLSTM, LSTM, and MLP Models . . . . .	138
D.3	The short-term forecasting curve for actual data and forecasts using BLSTM, SLSTM, SGRU, BGRU, and MLP Models . . . . .	142
D.4	Learning curve of BLSTM, SLSTM, SGRU, and BGRU Models for short-term forecasting . . . . .	143
D.5	Methodology proposed to design the MRA-SLSTM architecture . . . .	144
D.6	Plot of data generated by the Mackey-Glass time series ( $a = 0.2$ , $b = 0.1$ , $\tau = 17$ , and $x_0 = 1.2$ ) . . . . .	145
D.7	Mackey-Glass data curves of actual values versus forecasts using best selected models of EWT-SLSTM and SWT-SLSTM architectures . . . .	147
D.8	S&P data curves of actual prices versus short-term forecasts using best selected models of EWT-SLSTM and SWT-SLSTM approaches . . . .	149
D.9	Methodology used to train the GRU autoencoder . . . . .	153
D.10	Defining simple moving and exponential moving averages . . . . .	153
D.11	Input data composed of denoised features and moving averages . . . .	154
D.12	Methodology used to train the model . . . . .	154
D.13	The curves of the original and denoised data for part of the data . . .	155
D.14	Curves of part of the original data with corresponding generated technical features . . . . .	156

# LIST OF ABBREVIATIONS

2DPCA	Two-Dimensional Principal Component Analysis
ANNs	Artificial Neural Networks
AR	Autoregressive
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
BGRU	Bidirectional Gated Recurrent Unit
BLSTM	Bidirectional Long Short Term Memory
BPTT	Backpropagation Through Time
DNNs	Deep Learning Neural Networks
DWT	Discrete Wavelet Transform
EMD	Empirical Mode Decomposition
EMH	Efficient Market Hypothesis
EWT	Empirical Wavelet Transform
FRPCA	Fuzzy Robust Principal Component Analysis
GRU	Gated Recurrent Unit
HMM	Hidden Markov Model

KPCA	Kernel-based Principal Component Analysis
LSTM	Long-Short Term Memory
MA	Moving Average
MAPE	Mean Absolute Percentage Error
MLP	Multilayer Perceptron Neural Network
MMSE	Minimum Mean Square Error
MOPSO	Multi-Objective Particle Swarm Optimization
MRA	Multiresolution Analysis
PCA	Principal Component Analysis
RBFNN	Radial Basis Functional Neural Network
RNN	Recurrent Neural Network
SGRU	Stacked Gated Recurrent Unit
SLSTM	Stacked Long-Short Term Memory
SVM	Support Vector Machine
SWT	Stationary Wavelet Transform
TIs	Technical Indicators

# THESIS ABSTRACT

**NAME:** KHALED AHMED LUTF AL-THELAYA

**TITLE OF STUDY:** COMBINING DEEP LEARNING AND TECHNICAL ANALYSIS FOR MULTIREOLUTION FORECASTING OF FINANCIAL STOCK MARKETS

**MAJOR FIELD:** Computer Science

**DATE OF DEGREE:** December 2018

*Forecasting financial time series is considered one of the most challenging problems due to the noisy and complex structure exhibited by its constructs. Recently, there has been a rapidly growing interest in deep learning research and its applications to real-world complex problems. Deep learning neural networks are proven to be superior in many research fields. The aim of this work is to develop a multiresolution forecasting approach for financial stock markets using a deep learning methodology. We analyze the significance of various technical indicators in improving forecasting of financial markets and explore the impact of combining technical analysis with deep learning on the forecasting results. Many deep learning architectures designed using variants of deep recurrent neural networks are developed to forecast short- and long-*

*term prices. We investigate and compare two multiresolution analysis methods for data decomposition. The proposed approach is developed using a combination of empirical wavelet analysis method with deep gated recurrent unit network. The impact of several technical indicators on the proposed approach is investigated and analyzed. The proposed methodology is evaluated using S&P and DJIA indices which are two of the most common benchmark time series widely employed to evaluate financial time-series forecasting. Furthermore, we evaluate the proposed approach on the TASI Saudi stock market index (Tadawul). Our proposed models show forecasting accuracies superior to some recent related work in literature. The achieved accuracies are compared to that of the random-walk model and analyzed with respect to the efficient market hypothesis. Our results indicate that efficient market hypothesis could be rejected for short-term (one day) forecast and approved for long-term (one month) forecast.*



## ملخص الرسالة

الاسم: خالد أحمد لطف الثالبا

عنوان الدراسة: استخدام تقنية التعلم العميق والتحليل التقني للتنبؤ بسوق الأسهم المالية باستخدام التحليل متعدد المستويات

التخصص: علوم الحاسوب

تاريخ الدرجة العلمية: ديسمبر 2018

يعد التنبؤ بمستقبل السلاسل الزمنية للأسواق المالية من أكثر المجالات صعوبةً نتيجةً للتركيب المعقدة والتشويش الكامن في مكوناتها، وحدثاً زاد الاهتمام بالبحوث المتعلقة بالتعلم العميق وتطبيقاته على مشاكل واقعية، وقد أثبتت الشبكات العصبية الاصطناعية المعتمدة على التعلم العميق قدرةً عاليةً في معظم مجالات البحث العلمي. وتهدف هذه الدراسة إلى استخدام تقنية التحليل متعدد المستويات (Multiresolution Analysis) للتنبؤ بأسعار أسواق الأسهم العالمية بالاستفادة من منهجيات التعلم العميق المتعددة، كما تهدف أيضًا إلى دراسة مدى تأثير دمج المؤشرات التقنية (Technical Indicators) المختلفة مع التعلم العميق على النتائج. وقد تم تطوير العديد من أنواع الشبكات العصبية المتكررة العميقة (Deep Recurrent Neural Networks) للتنبؤ بالأسعار في المستقبل القريب والبعيد، وقامت الدراسة بتحليل ومقارنة طريقتين من طرق التحليل متعدد المستويات لتحليل البيانات إلى عدة أجزاء، ومن ثم اقتراح وتطوير طريقة باستخدام تركيبة مكونة من طريقة المويجات التحليلية التجريبية (Empirical Wavelet Analysis) مع أسلوب الشبكات العصبية المتكررة المبوية (Gated Recurrent Unit)، وتضمنت الدراسة أيضًا تحليل دور المؤشرات التقنية (Technical Indicators) في تحسين النتائج، واستخدمت في تقييم الأداء مؤشر إس أند بي (S&P) ومؤشر داو جونز (DJIA) وهما من أكثر سلاسل البيانات استخدامًا في تقييم مستوى أداء الطرق المختلفة للتنبؤ بسلاسل البيانات المالية، وكذلك استخدمت الدراسة مؤشر سوق الأسهم السعودية تداول (TASI) في بعض التجارب. وقد أثبتت التجارب كفاءة ودقة الطريقة المقترحة مقارنةً ببعض الدراسات السابقة الحديثة، وتمت أيضًا مقارنة النتائج مع نموذج السير العشوائي (Random Walk) وتحليلها في ضوء فرضية كفاءة السوق (Efficient Market Hypothesis).

## CHAPTER 1

# INTRODUCTION

People are increasingly engaged into financial stock markets interactions by acquiring and following information related to stock market trends, currency exchange, global market indicators. Interest in financial data is receiving a growing attention due to its significant impact on our daily lives. People might be directly or indirectly affected by price changes in financial markets and they eventually need to deal with information brought to their attention related to financial markets interactions. Daily news reports on television, radio, and newspapers keep people informed about latest stock market trends, interest rates, and currency exchange rates. Moreover, people who are involved in international trade are dealing directly with financial time series. Financial analysts, brokers, businessmen, and corporate investors need to analyze and observe price trends and behavior in order to understand the likely changes of the prices in the future. Forecasts of future prices provide early warning to avoid unnecessary risks.

Researchers have been studying and investigating financial time series for more than three decades attempting to develop efficient solutions to analyze, forecast and

predict future trends and returns of financial time series. Technical analysis is one of the evolved methodologies commonly followed to forecast and analyze financial time series to identify trading opportunities [1]. It is mainly based on performing different kinds of analysis using historical prices in order to draw conclusions about market trends in the future. Technical analysts attempt to predict the market by tracing patterns observed in the historical market data represented by various charts. They usually employ different Technical Indicators (TIs), such as moving averages, trend lines and momentum to forecast trends and future prices. Another methodology used to predict market behavior is fundamental analysis, which depends on using information related to the intrinsic values of the market and national economy, such as inflation, interest rate, trade balance, etc. Fundamental analysts attempt to study the market or the industry to have a clear picture of the firm they will choose to invest on. They perform an in-depth analysis of the market performance and growth prospect to make predictions about the future. It is mainly used for long-term investment decisions and strategies.

One of the methodologies employed to perform financial time-series forecasting is statistical analysis. It aims at approximating future values of financial time series based on a linear combination of historical data. Many solutions have been proposed based on linear regression methods such as linear Autoregressive (AR) and Autoregressive Moving Average (ARMA) [2–4]. Most of the financial time series tend to exhibit nonlinear patterns. Thus, linear statistical models to some extent are unable to adequately represent financial time-series patterns to predict its undefined behavior based on seen historical data [5, 6]. The solution to this problem is to use nonlinear

machine-learning methodologies, such as Artificial Neural Networks (ANNs), to model nonlinearity and build more efficient models which have the ability to produce better estimations. Solutions developed by ANNs show significant improvement in time-series prediction and forecasting [7]. However, a recent machine-learning methodology has been proposed based on deep learning architectures to perform deep analysis and modeling to data science. It has been developed based on performing deep data analysis and training to achieve prediction and forecasting. For instance, Deep Learning Neural Networks (DNNs) show superior performance and efficiency in many scientific fields including forecasting of financial time series [8–10]

Due to the unprecedented advances in deep learning, many scientific fields exploit its high performance to build solutions to different kinds of problems. Financial time series is considered to be a good candidate and suitable environment for deep learning techniques. Some research works attempted to study different aspects of integrating deep learning into financial time series [9–11]. Some of them approached the problem by examining and comparing different kinds of deep learning techniques [8, 9]. Others combine deep learning with data preprocessing methods to reduce and denoise the input data [12]. Data preprocessing is considered a highly important stage in data analysis and modeling. Our study investigates the application of Multiresolution Analysis (MRA), which is one of the most effective tools for data preprocessing, analysis, denoising and decomposition. It has been widely used in a variety of time-series modeling approaches [2, 13], yet few studies used it with deep learning networks to perform financial time-series forecasting [10, 14].

Integrating deep learning networks and multiresolution analysis into financial time

series is a new research area full of opportunities to build more efficient models. Many methods of wavelet analysis are proposed in the literature. In our study, we adopted Empirical Wavelet Transform (EWT) to perform data decomposition and analysis. To the best of our knowledge, no previous work has been allocated to study the impact of using multiresolution analysis using EWT with deep learning networks for financial time-series analysis and forecasting. Our work aims at filling this gap by designing an efficient and accurate forecasting model based on using multiresolution analysis for data decomposition and denoising. We compared forecast results of the EWT analysis method with the traditional Stationary Wavelet Transform (SWT). Moreover, we studied the integration and investigation of TIs with deep learning networks to build accurate forecasting models. Our study explored the impact of using TIs as input features to deep learning networks to perform forecasting. Our proposed solution combined both multiresolution analysis and deep learning techniques with TIs to build an efficient an MRA-DNN solution for financial time-series forecasting.

We investigated multiple architectures based on different parameters and multiple time steps for forecasting several time horizon scales in the future. Many models were developed based on the proposed methodology by applying changes to the types of deep learning networks and multiresolution wavelet analysis methods. Besides, several trials were conducted using different combinations of input features. Some of the developed models were trained using raw stock market data only whereas other models employed TIs as additional input features. Different deep recurrent neural networks, such as, Gated Recurrent Unit (GRU) and Long-Short Term Memory (LSTM), were designed using stacked and bidirectional architectures. We also developed different

models using raw stock data as input features without performing multiresolution analysis. All models developed by this work were evaluated using S&P and DJIA stock market indices which are two of the most common benchmark datasets used for evaluating financial time-series forecasting approaches in the literature. We further evaluated the forecasting approach proposed by this study on Saudi stock market using the Tadawul All Share Index (TASI). The best developed models were also analyzed and compared to some previous work in the literature.

Proving the predictability of the market is an arguable subject that has been much investigated by researchers and academics. A hypothesis in finance, known as the Efficient Market Hypothesis (EMH) devised in 1960s implies that forecasting efficient market is impossible. However, so far there has been no ultimate empirical evidence on the validity of EMH (more details can be found in Appendix A) [15]. Many studies [16–18] argued the validity of the EMH by developing market forecasting models and the results do not support the EMH and question the applicability of this hypothesis with respect to employed data and investigated markets. Therefore, before we perform forecasting experiments using benchmark datasets, we prove that daily returns of the benchmark datasets are not randomly distributed. Testing for non-randomness of time series provides an indication that the forecasting task is possible. To contribute to the answer of the question of stock market predictability, we compare the results of our proposed methodology to the results produced by the Random Walk (RW) model implied by the EMH. Failing to outperform RW model emphasizes that the stock market prices are unpredictable which may provide a supportive evidence for the EMH. Most of the forecasting models proposed in the literature test the EMH in

its weak or semi-strong form [19, 20]. The scope of this work will be limited to testing the weak form of the EMH by working on technical data which involves analyzing changes of prices over the past to forecast the future. Expanding the scope of this study to test the semi-strong form of the EMH is beyond the scope of this work due to the unavailability of required fundamental data about the market shares and companies.

## 1.1 Motivation

Financial time series are highly volatile noisy data streams and stock market forecasting is typically regarded as a challenging application. The research of accurately forecasting stock market prices is in progress with the goal of fulfilling best economic gain and better profitable return.

Many studies found that time-series data tends to exhibit nonlinear pattern and they thus adopted different kinds of nonlinear modeling approaches. However, new trends in machine learning showed that deep learning methodology can model highly nonlinear data efficiently. It can be observed from the literature that exploring and addressing the integration of deep learning methodology to time-series analysis and forecasting is an important area of research which is receiving increasing attention. We also observe that there is still room to investigate the application of deep learning combined with different kinds of preprocessing techniques such as multiresolution analysis to forecast financial time series.

Using multiresolution analysis for denoising, analyzing and decomposing the data,

is an important research field and a key factor to developing efficient time-series forecasting models. Besides, technical analysis and TIs help develop representative, effective new features, free of noise, and free of unanticipated patterns in the data. The impact of TIs on the analysis of time series has been addressed by some research with respect to some modeling techniques such as ANNs [21], yet it was not adequately explored regarding deep learning methodologies using multiresolution analysis. Thus, applying deep learning to financial time series with the integration of multiresolution analysis techniques and TIs is an open question that needs to be addressed. It is an interesting problem domain worthy to be studied and analyzed.

## 1.2 Objectives

The objectives of this thesis are as follows:

- Intensively review related work on multiresolution forecasting of financial markets.
- Apply and compare different deep learning architectures to perform stock market forecasting for multiple time scale horizons using different time steps input structure.
- Design and develop a multiresolution forecasting approach of financial stock markets using deep learning techniques.
- Analyze the significance of various TIs in improving the forecasting of financial markets.



- Explore the impact of combining TIs with multiresolution analysis and deep learning on the forecasting results.

### 1.3 Contributions

- Evaluation of several variants of deep recurrent neural networks such as LSTM and GRU for stock market forecasting using different deep layered architectures.
- Stock market forecast using multivariate analysis by comparing deep architectures with shallow neural networks.
- Perform stock market forecasting using combination of TIs and deep autoencoder GRU for data denoising.
- Conduct comparative experiments using Bidirectional Long Short Term Memory (BLSTM) deep network to forecast stock market using EWT and SWT for multiresolution analysis.
- Forecasting stock market using Bidirectional Gated Recurrent Unit (BGRU) and EWT using both raw stock prices and log returns of stock prices.
- Develop a deep learning architecture using BGRU with combination of TIs and multiresolution analysis using EWT.

## 1.4 Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 introduces some related works and classifies them according to the employed approach and methodology to perform the forecasting. Chapter 3 describes the details of the proposed methodology for financial time-series forecasting. Chapter 4 describes the benchmarking datasets and applies some statistical methods to explore their characteristics. It also describes various techniques for data preprocessing. Chapter 5 discusses the conducted experiments to evaluate the proposed methodology. Chapter 6 summarizes the conclusions drawn by this study in addition to limitations and suggestions for future work.

## CHAPTER 2

# RELATED WORK

The analysis of time series mainly comprises statistical techniques to characterize, describe, and model data patterns. Forecasting time series aims to employ historical data to predict future values. Weather, wind speed, water demand, electricity consumption and many other problem domains apply time-series forecasting techniques to predict future values [22–24]. Financial time series forecasting [25, 26] is one of the important fields in time series used for making best choices to reach objectives and achieve profitable goals. In the following sections, we discuss some related works which developed financial time-series forecasting models. Figure 2.1 shows classification of some related works based on type of forecasting models. Some linear forecasting models in the literature are discussed in Section 2.1. Moreover, we review some related works which used nonlinear forecasting techniques to perform the forecasting in Section 2.2. Section 2.3 describes some related works that used deep learning techniques in financial time series forecasting. Some related works which developed forecasting models based on MRA are discussed in Section 2.4. Related works which developed forecasting models using TIs are discussed in Section 2.5.

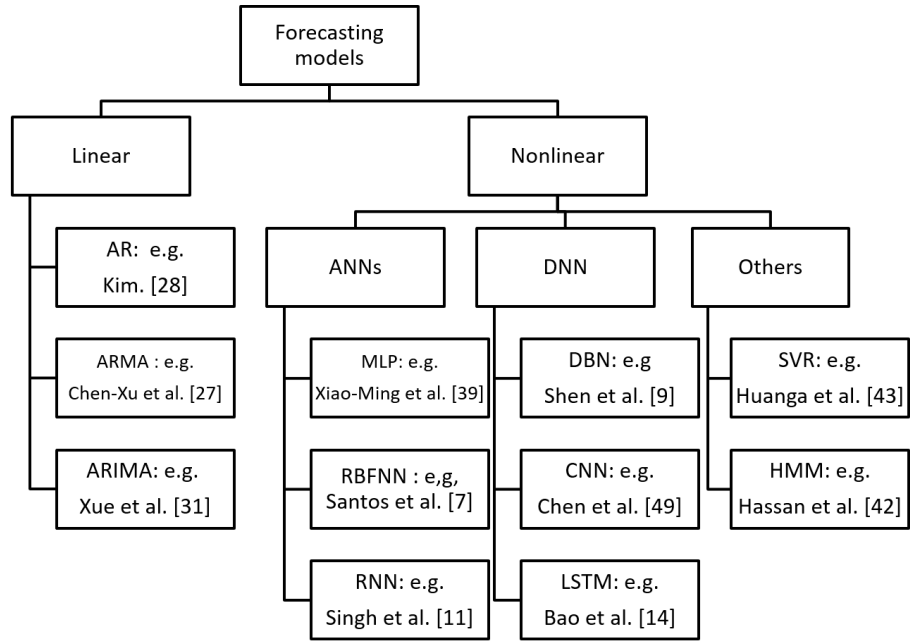


Figure 2.1: Classification of some related works according to type of forecasting models

## 2.1 Linear Forecasting Models

Time-series forecasting involves different linear modeling techniques such as AR, Moving Average (MA), ARMA, and Autoregressive Integrated Moving Average (ARIMA). Many research [27–30] adopted ARMA modeling technique for time-series forecasting and prediction. Chen-Xu et al. [27] built a model based on ARMA to predict bank cash flow. The performance was compared with the MA model only. Kim [28] discussed the symmetric maximum likelihood loss function and proposed asymmetric loss function to build ARMA model to forecast stock returns. The proposed cost-sensitive loss function outperformed ML estimator function, yet it was not compared with any other forecasting models. Chen et al. [30] applied an adaptive approach to build ARMA model by deriving the error based on the theory of Minimum Mean Square Error (MMSE). Similarly, ARIMA modeling was adopted by [31–34] for time series-

forecasting. The constructed models were evaluated on different application domains including stock market forecasting. All linear modeling techniques cited above, were not compared with nonlinear modeling techniques or other types of modeling.

## 2.2 Nonlinear Forecasting Models

Mainly, nonlinear modeling techniques when compared with linear techniques show good performance and better accuracy. Studies [7, 35–37] that compare the performance of the linear and nonlinear forecasting models support the superiority of nonlinear models. Thus, many nonlinear modeling techniques were employed to forecast time series. Santos et al. [7] investigated the advantages of integrating nonlinear Multilayer Perceptron Neural Network (MLP) with Radial Basis Functional Neural Network (RBFNN) and the Takagi-Sugeno fuzzy system to exchange-rate forecasting. The performance of the proposed model was compared with the performance of the linear ARMA and ARMA-GARCH models and showed better forecasting accuracy. Another study was conducted by Pao et al. [38] to compare the performance of linear and nonlinear models, particularly two nonlinear ANN models and three linear time-series cross sectional models. It demonstrated that ANN models exhibit higher forecasting accuracy.

ANNs are widely used for time-series analysis and forecasting due to its ability to model nonlinearity in time-series data. Many studies investigated the merits and abilities of ANNs in time-series forecasting. Xiao-Ming et al. [39] combined ten ANN models together to learn the forecasting model. They used Adaboost technique to

train the combined models by selecting 38 TIs of Shanghai Stock Exchange and international stock markets. The proposed model forecasts the weekly direction of stock index movement. The combination of ANN weak predictor models were employed to form a more efficient model. They verified the performance of the proposed model by comparing the prediction results of the single ANN models on the same test set with the results recorded by the combined model. Guo et al. [40] also used ANN to build a hybrid dimensional reduction approach. The model combined Two-Directional Two-Dimensional Principal Component Analysis (2DPCA) with RBFNN to forecast stock market daily closing price trend. The proposed model input features consisted of 36 stock market technical variables and use a sliding window to shape the input data. The dimension of the input data reduced by 2DPCA to extract its intrinsic features. The data was fed into RBFNN to forecast the next day's stock movement and price. The evaluation used the Shanghai stock market index. Zhong et al. [41] devised that the best technique to forecast time series is ANN and put more emphasis on data preprocessing phase. They proposed using three mature dimensionality reduction techniques, including Principal Component Analysis (PCA), Fuzzy Robust Principal Component Analysis (FRPCA), and Kernel-based Principal Component Analysis (KPCA). The ANN model with the PCA gives slightly higher classification accuracy than the KPCA or FRPCA.

Some other modeling techniques such as Adaboost, and Hidden Markov Model (HMM) were also applied to time-series forecasting. Hassan and Nath [42] came up with a new approach which depends on using HMM to predict next day return value. The proposed approach searches for patterns in the dataset that matches specific

query. They used daily stock data of Southwest Airlines for training and testing. They also compared the new model with other four ANNs models and found that the Mean Absolute Percentage Error (MAPE) values are quite similar. Huang et al. [43] used Support Vector Machine (SVM) technique to train a model to forecast the direction of the weekly movement of the NIKKEI 225 index. They compared the produced model with other models generated by Linear Discriminant Analysis, Quadratic Discriminant Analysis and Elman Backpropagation Neural Networks. They found that SVM produces the best results.

Moreover, Majhi et al. [44] introduced a nondominated sorting genetic algorithm version-II (NSGA-II) and Multi-Objective Particle Swarm Optimization (MOPSO) to efficiently design models for stock market prediction to adjust four performance constraints. The developed adaptive model was introduced with nonlinearity at the input end by Legendre polynomial expansion scheme. It was developed by adapting the stepwise algorithms. A decision-making strategy based on fuzzy logic suggested to get the best solutions from models. Comparison of the results showed that the performance of multi-objective optimization model is better while the single objective optimization model exhibited better performance in terms of the Theil's U.

Another study conducted by Liu et al. [45] employed automatic clustering and features categorizing of growing hierarchical self-organizing map (GHSOM) to build the model. They described patterns and criteria to determine stocks for investment and maximize profits. The proposed approach adopts Elite Particle Swarm Optimization (EPSO) to elucidate optimal trading opportunities and combines Growing Hierarchical Self-Organizing Map (GHSOM) and EPSO in its stock selection strategy.

Table 2.1 shows some related works that use different kinds of techniques for financial time-series prediction and forecasting.

Table 2.1: List of some forecasting techniques im the literature

Ref	Technique	Dataset
[42]	Hidden markov model	The daily stock data of southwest airlines
[43]	Support vector machine	NIKKEI 225 index
[39]	Adaboost and ANN	Shanghai stock exchange and international stock markets
[40]	Two-directional two-dimensional PCA and a radial basis function neural network	Shanghai stock market index
[44]	NSGA-II, MOPSO	DJIA stock index.
[46]	Bayesian regularized ANN	Microsoft corp. and Goldman Sachs group Inc.
[41]	PCA, FRPCA, KPCA, ANN	S&P index ETF (SPY)
[45]	EPSO, GHSOM	Taiwan stock market.
[47]	Adaboost (SVM), Adaboost (QGA-SVM), Adaboost (GA-SVM)	MTK and China steel, Taiwan stock market

## 2.3 Deep Learning Forecasting Models

Generally, nonlinear techniques outperformed linear models and produced better forecasting methods. The majority of the reviewed research above, suggested that ANN modeling technique is adequate to represent the nonlinearity of time-series data for forecasting and prediction. However, it did not succeed to provide an efficient models with encouraging performance. According to Singh et al. [11], ANNs don't produce quite enough performance to warrant being the best modeling technique for financial time series. DNNs, on the other hand, showed superior performance in many other areas of science such as signal processing, speech recognition, and image classification. Therefore, adopting DNNs techniques into financial time-series forecasting is a promising research area. Singh et al. [11] proposed an approach using Recurrent Neural Network (RNN) and suggested that DNNs models are better than ANNs.



Although there are limited number of DNNs techniques proposed, they tend to produce more efficient models. More investigation to examine the advantages and limitations of using deep learning methodologies is needed to improve financial time-series forecasting. Chong et al. [8] explored the potential advantages and drawbacks of using and integrating DNNs into stock market forecasting. They used it to extract features from high frequency raw data collected from intra-day stock returns. They conducted experiments to study the effects of three unsupervised feature extraction techniques on the network to predict market direction. It showed that DNNs can improve the results of the autoregressive model and enhance the prediction ability of the model.

Shen et al. [9] also designed a model using an improved Deep Belief Network (DBN). The proposed model designated to forecast exchange rates. Continuous restricted Boltzmann machines (CRBMs) used to construct and improve the DBN. Tsantekidis et al. [48] applied Convolutional Neural Network (CNN) on special kind of high frequency data collected by limit order book (LOB). Limit order data is collected from a specific set of constraints established to buy or sell a specific number of shares within a set price. The proposed model outperformed both SVM and MLP models.

Li et al. [12] attempted to improve the prediction model built using LSTM neural network by integrating naïve bayes modeling technique to include and extract investor sentiment and market factors from forum posts. Some researchers use different kinds of data to train deep learning models. Chen et al. [49] transferred time-series data into 2D images and fed them into CNN for training.

Table 2.2: List of some related work using deep learning techniques for financial time-series forecasting

Ref	Problem	Technique	Data Preprocessing	Dataset	Results
[13]	Financial time-series forecasting	Two layers of MLPs. Three stages of prediction models.	Multiresolution wavelet transform, autocorrelation shell representation. Bayesian method of ARD	Multiple datasets	MSE is between 0.0103 and 0.0155
[2]	Financial time-series analysis and forecasting	AR, ARMA, ARIMA, MWT, MODWT, neural network autoregressive.	Fourier transform, Discrete wavelet transform, multiresolution analysis, MODWT	S&P data	R-Squared = 0.636
[14]	Financial time-series forecasting	SAEs and LSTM combined with wavelet transform	Multiresolution analysis, wavelet transform	Multiple datasets	MAPE between 0.011 and 0.019
[8]	High-frequency intraday stock returns	DNN, autoregressive model	Principal component analysis, autoencoder, and the restricted Boltzmann machine	KOSPI stock market	MAE scores between 0.5852 and 0.5838
[9]	Forecasting exchange rates	DBN, conjugate gradient method	Continuous restricted Boltzmann machines (CRBMs)	Three exchange rates datasets	RMSE, MAE and MAPE are 0.016310, 0.012188 and 2.923540, respectively.
[10]	Predict stock market indices	Convolutional Neural Networks (CNN), LSTM recurrent neural networks	Wavelet transform	S&P stock time series	Wavelet-CNN scores MSE = 0.24, Accuracy = 0.55
[11]	Stock market prediction	Recurrent neural network	2-directional principal component analysis 2D-2PCA	NASDAQ and the prediction is done for individual stock	MAPE = 0.081409, RMSE = 0.010119
[48]	Predict mid-price future movements	Convolutional neural networks (CNNs)		High frequency limit order book (LOB) data	Recall = 55.58, precision = 67.12, F1=59.44
[12]	Stock market Prediction by incorporating investor sentiment	LSTM, Naive Bayes		Stock discussion forum posts, CSI300 index	Prediction accuracy of 87.86%
[47]	Financial time-series prediction and analysis	CNN	MAM, DMAM, transform the time-series data into 2D images	Intraday one-minute data of Taiwan index	Average accuracy between 53.76 and 54.86

## 2.4 Multiresolution Analysis

Characterizing, modeling, and extracting features of time series is best achieved by integrating signal processing methodologies into time-series analysis and decomposition. Time series can be represented using different kinds of signal transforms such as Fourier and wavelet transforms. Wavelet transform outperforms Fourier transform in analyzing nonstationary data thus making it a good candidate for time-series decomposition. Some research suggested that the use of wavelet transform is better than Fourier transform, hence, Fourier transform is not a perfect choice for non-stationary time-series analysis. And using Short-Time Fourier Transform (STFT) for MRA is not the best choice as devised by Kilic et al. [2]. Wavelet methodology was first introduced by Grossmann and Morlet [50]. This pioneering work was followed by presenting MRA by Mallat [51]. Combining multiresolution wavelet methodology showed a significant improvement in data analysis and decomposition in many scientific areas including time series.

Many researchers explored the advantage of representing time-series data using MRA. Ismail et al. [52] for example, attempted to understand and characterize financial time series using MRA. The study used Linear ARIMA with wavelets to address forecasting results using multiresolution approach fitting. Kilic et al. [2] conducted experiments to analyze S&P stock market data using MRA, and some other descriptive statistical modeling. They also conducted experiments to investigate and compare the integration of MRA with linear as well as nonlinear forecasting methods. They concluded that using nonlinear models with MRA produced better results.

Some other studies concluded that nonlinear modeling techniques and ANNs particularly show better performance as Bekiros et al. [53] suggested. The study applied MRA to linear and nonlinear models involving neural networks that show superior performance. Zhan et al. [13] also found that the integration of wavelet transforms with ANN produce better forecasting results. They used shift invariant scale-related wavelet transform representation. The transformation established based on Autocorrelation Shell Representation (ASR). The proposed approach transforms the financial time series and extracts wavelet coefficient by ASR and applies Bayesian method of automatic relevance determination (ARD) to select best features for the first layer that is composed of multiple MLP predictors. The output of this layer is input to the second layer that consists of one MLP predictor. The proposed model proved to be efficient when compared with another MLP model without wavelet transform.

Few research attempted to combine wavelet transforms with deep learning techniques to learn financial time-series prediction model. Persio et al. [10] explored the effectiveness and efficiency of wavelet analysis with deep learning techniques in financial time-series forecasting . The study included several experiments comparing the forecasting performance of CNN and LSTM. The study concluded that CNN when combined with MRA tend to outperform other compared DNNs. Other types of DNNs such as Stacked Autoencoders (SAEs) used by Bao et al. [14] to propose a prediction approach using LSTM combined with wavelet transform to denoise input features. The combined model outperformed the other three separated single models.

Moreover, discrete wavelet transform (DWT) and multiplicative seasonal algorithm (MSA) are both preprocessing techniques , used to analyze and decompose time series

into sub-components by isolating seasonal and trend elements of the time-series data to produce higher accuracy forecasting performance. Altunkaynak et al. [22], conducted experiment to compare between the impact of (DWT) and (MSA) with regard to water-consumption time-series data. The study train two MLP models each of which combined with one of the preprocessing analytical techniques. The MSA-MLP model tend to produce better performance than DWT-MLP.

## **2.5 Technical Analysis and Technical Indicators**

Another area of data preprocessing techniques, especially for financial time series, is Technical Analysis (TA). It is one of the effective tools used to improve time-series forecasting accuracy. New input features are forged by TIs based on some technical means to represent trends and patterns of the data. The new technical features use raw features such as opening, low, high and closing price values to form a new more expressive and representative features. Number and type of TIs used is subject to the problem domain. TIs, such as RSI, are mathematical operations employed to rule whether a stock is overbought or oversold or a price trend is strong or weak, and thus to predict stock price trend movements. Some studies conducted experiments to examine the impact of TIs on time-series forecasting. They used different kinds of TIs as listed in Tables 2.3 and 2.4. Some research [14, 49, 54, 55] adopted deep learning methodology and use TIs to improve the forecasting results. Yet, no research - as far as we know - analytically compare and investigate the impact of using TIs with respect to deep learning techniques and multiresolution preprocessing methodologies.

Table 2.3: Some technical indicators adopted in other studies to perform stock market forecasting

Reference	TIs
[56]	SMA, EMA, RSI, ROC, MACD, ATR
[57]	RSI, MACD
[58]	SMA, EMA, RSI, ROC, ADX, DEMA, KAMA, Momentum, TEMA, TRIX, WilliamsR%, WMA
[59]	SMA, EMA, RSI, ROC, MACD, ATR, WilliamsR%, ADMI, Stochastic %K
[21]	SMA, RSI, MACD, ATR, Momentum, WilliamsR%, CCI, Stochastic %K, EMVA
[60]	SMA, EMA
[61]	EMA, Stochastic %K, Stochastic %D, Stochastic %J, WMR, OBV
[14]	SMA, EMA, ROC, MACD, ATR, CCI, BOLL, MTM, SMI, WVAD
[62]	SMA, RSI, ROC, MACD, WilliamsR%, Stochastic %K, Stochastic %D, OBV, VR
[63]	SMA, RSI, ROC, MACD, CCI, CMO, PPO, STOCH
[64]	SMA, EMA, RSI, ROC, MACD, ADX, TEMA, WilliamsR%, WMA, ADMI, CCI, CMO, HMA, PPO, CMFI
[1]	EMA, RSI, ROC, MACD, Stochastic %K, Stochastic %D, OBV, HMA, TSI, STOCH
[59]	SMA, EMA, RSI, ROC, ATR, WilliamsR%, ADMI, CCI, Stochastic %K, Stochastic %D
[65]	SMA, EMA, RSI, ROC, Momentum, WilliamsR%, Stochastic %K
[66]	SMA, EMA, RSI, ROC, MACD, ATR, Momentum, WilliamsR%, WMA, CCI, Stochastic %K, Stochastic %D, PPO, DIS, BIAS, BB, MFI, SLOW D, CLV, ADO
[67]	SMA, RSI, MACD, Momentum, WilliamsR%, WMA, CCI, Stochastic %K, Stochastic %D, ADO
[68]	SMA, RSI, Stochastic %K, Stochastic %D, BIAS
[69]	SMA, RSI, MACD, WilliamsR%, BIAS, ADO, MO1, MO2, DIFN, DIFF, DIFE

Table 2.4: Continuous of Table 2.3

Reference	TIs
[70]	SMA, WilliamsR%, BIAS, mo1, DIFN, DIFT
[71]	SMA, RSI, ROC, Stochastic %D, OBV, VR, DIS, PSY, AR
[72]	SMA, EMA, RSI, ROC, Momentum, WilliamsR%, CCI, Stochastic %K, Stochastic %D, DIS, ADO
[73]	SMA, EMA, MACD, ATR, Momentum, WilliamsR%, CCI, Stochastic %K, Stochastic %D, PPO, TSI, BIAS, Ulcer, UO, SignalLine
[74]	SMA, RSI, MACD, WilliamsR%, Stochastic %K, BIAS, DIF, Transaction Volume (TV)
[75]	SMA, EMA, Momentum, WilliamsR%, Stochastic %K, Stochastic %D, SLOW D, ADO
[76]	SMA, EMA, RSI, ROC, MACD, TRIX, WilliamsR%, CMO, STOCH, ADO
[77]	SMA, RSI, MACD, Momentum, WilliamsR%, CCI, Stochastic %K, Stochastic %D, ADO
[78]	MACD, Stochastic %K, Stochastic %D
[79]	SMA, WilliamsR%, Stochastic %K, Stochastic %D
[80]	SMA, RSI, MACD, Momentum, WilliamsR%, BIAS, PSY, DIFN, DIFF, DIFE
[81]	RSI, MACD
[46]	EMA, RSI, WilliamsR%, Stochastic %K, Stochastic %D
[82]	SMA, RSI, Momentum, WilliamsR%, Stochastic %K, Stochastic %D, VR, PSY, AR
[83]	RSI
[84]	SMA, RSI, MACD, Momentum, WilliamsR%, WMA, CCI, Stochastic %K, Stochastic %D, ADO
[85]	SMA, RSI, Momentum, WilliamsR%, Stochastic %K, Stochastic %D, VR, PSY, AR
[86]	SMA, EMA, RSI, WilliamsR%, OBV, STOCH, ADO, PROC, CPACC, HPACC

## CHAPTER 3

# PROPOSED METHODOLOGY

In this chapter, we describe the proposed methodology for short- and long-term forecasting of financial time series. The proposed methodology is composed of three main processing techniques: data preprocessing, multiresolution and technical analysis, and deep neural network learning approach. Overview about the adopted methodology is discussed in Section 3.1. Technical and multiresolution analysis are described in Sections 3.2 and 3.3, respectively. Discussion about the adopted deep learning methodology is included in Section 3.4.

### 3.1 Methodology Overview

The proposed methodology is based on a three-stage architecture as depicted in Figure 3.1. The first stage performs data exploration and preprocessing. The second stage includes technical and multiresolution data analysis. The third stage is based on deep learning networks to build forecasting models. The input data is first analyzed using some time-series statistical tests to check for randomness. The data is then technically



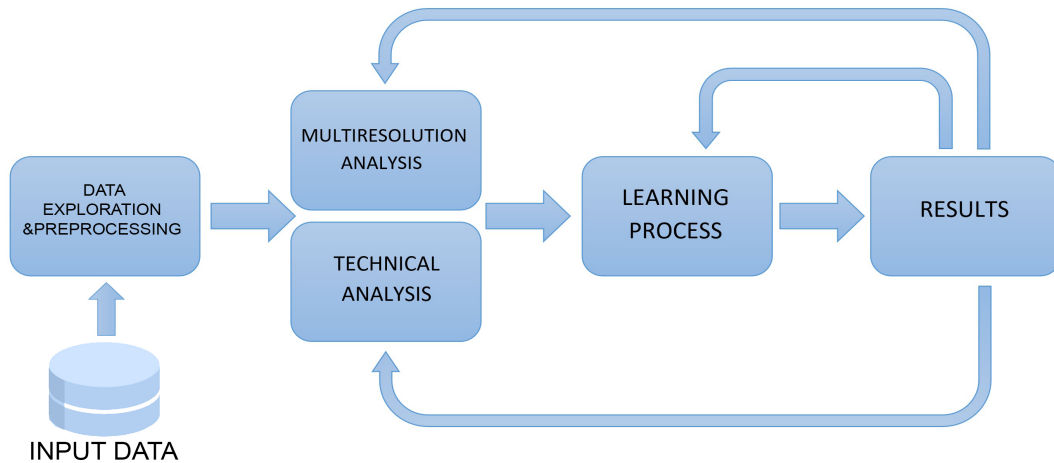


Figure 3.1: Overview of the proposed methodology

analyzed and a group of TIs are computed to extract more informative input features, which are typically used by technical analysts of financial markets.

Data preprocessing is an important factor in learning forecasting models. It plays a significant role in defining the best input features by analyzing, transforming and decomposing the dataset. The effectiveness of forecasting models greatly depends on the statistical stationarity assumption of the time-series dataset. Hence, we first convert benchmark dataset into a stationary series using log returns. Another important step is the decomposition of the time series on different scales using MRA. Multiple streams of data are generated and utilized in building deep learning based forecasting models. Applying different processing mechanisms on multiple data representations is likely to increase the findings and leads to better forecasting results. Choosing the best MRA parameters setup relies on conducting experiments to devise the best configuration that has the highest performance results.

The learning methodology adopted by this work depends on deep learning techniques. The input features are fed into a deep neural network for learning network

weights. We adopt Deep Recurrent Neural Network (RNN) technique which is commonly used to process sequential data. It can process and model temporal patterns in time series. Two variants of deep recurrent neural networks, which are LSTM and GRU, are explored in this study to perform forecasting. Several experiments are conducted to select the best network design and parameters.

Multiple architectures are developed by this work using different configuration setup and multiple time steps for forecasting several time horizon scales in the future. The developed models used raw stock prices and log returns as input features. TIs are included as input features to some of the trained models and compared to those trained without using TIs. The best model is compared with several other approaches in the literature.

The following subsections provide more details about the main components in the proposed methodology: technical analysis, multiresolution analysis, and deep learning models.

## **3.2 Technical Analysis**

Technical analysis is a combination of mathematical and graphical tools used to predict the movement of market prices for a specific period in the future. TIs are a major and significant part of the technical analysis methodology. They are mainly used by traders to predict prices using trend and pattern recognition. Mostly, TIs are derived by applying some mathematical formulas to the stock price data such as the closing or opening prices. Many financial time-series forecasting researchers use TIs as input

features to denoise the data series and stabilize the modeling process.

This work aimed at incorporating TIs as input features into the deep learning models to study its impact and determine the best number and combination of TIs to improve the learning process and forecasting performance. Experiments are conducted to examine different TIs that are likely to create an efficient forecasting model. We select ten of the most commonly used TIs of financial time series based on surveying several studies in the literature, as listed in Tables 2.3 and 2.4. More details about these TIs and their computations are provided in Appendix C.

### **3.3 Multiresolution Analysis**

Financial time series may enclose several temporal and spectral patterns, which can be revealed by applying analysis methods to extract and process each pattern based on multiple resolution scales. MRA decomposes a time series into several but different mini-series that describe parts of the data based on predefined scales. Each part of the decomposed data can be distinguished and analyzed independently then used for data modeling and learning.

MRA allows removing unwanted noisy parts of the data and including the important influential ones by zooming-in into specific detailed data and at the same time gaining an overall picture of the data. Adopting this kind of significant abilities supports building an efficient forecasting model. Our proposed methodology depends on analyzing and decomposing the data into multiple scale data sets using EWT which is an adaptive signal and time-series analysis technique developed based on a combi-

nation of Empirical Mode Decomposition (EMD) and wavelet analysis. Each part or subset of the data is used for learning different instances of deep neural networks. We compare the adopted EWT analysis method to a stationary wavelet analysis method which is widely used in the literature for financial time-series analysis. To the best of our knowledge, we are the first to propose a forecasting model developed by combining EWT for data preprocessing with deep learning methodology to perform financial time-series forecasting. Based on trial and error experimental methodology, we define and determine best parameters and resolution level well-suited for defining more accurate models.

### 3.3.1 Wavelet Transform Concept

Wavelet transform is a generalization of Fourier transform, which is very common to extract features at multiple resolution levels of the time-series data. The decomposition is accomplished by defining basis functions using a mother wavelet then multiscale resolutions are extracted by projecting the given signal onto the basis functions. The main advantage of wavelet decomposition is that it extracts local and global features including the trend and spurious short fluctuations [87, 88]. The basis functions are defined in terms of the mother wavelet as follows [87],

$$\psi_{a,b}(t) = \frac{1}{\sqrt{|a|}} \psi\left(\frac{t-b}{a}\right), \quad a, b \in \mathbb{R}, \quad a \neq 0 \quad (3.1)$$

where  $b$  and  $a$  denote the translation (location) and dilation (scaling) parameters, respectively. For  $|a| > 1$ , the wavelet is stretched in time and captures low frequencies in the series. For  $|a| < 1$ , the wavelet is compressed in time and captures high frequencies in the series.

In order to get a multiresolution representation for the data, all the input data are decomposed by varying the translation and dilation parameters. There are several forms of mother wavelets such as those adopted in Haar, Daubechies and Symlets transformations. Daubechies wavelets are used by many time-series forecasting applications [13, 89, 90]. Discrete Wavelet Transform (DWT) basis function at time location  $n$  and dyadic scaling  $m$  is given by [88],

$$\psi_{m,n}(t) = 2^{-\frac{m}{2}} \psi(2^{-m} \cdot t - n) \quad (3.2)$$

The wavelets of DWT generated by the dyadic grid sampled wavelets are orthonormal. The inner product of the time-series data denoted by  $x(t)$  and the basis function  $\psi_{m,n}$  expressed in Equation 3.2 is as follows,

$$d_{m,n} = \sum_{t=0}^{N-1} x(t) \psi_{m,n}(t) \quad (3.3)$$

The wavelet coefficient  $d_{m,n}$  is defined by Equation 3.3 and parameterized by dilation  $m$  and translation  $n$  to return the detailed information (high frequencies) presented in the time series.

Using the mother wavelet to decompose a signal will result in infinite number of

basis functions to accurately represent the signal. In order to have a finite set of basis wavelet, an auxiliary function  $\phi(t)$ , known as a scaling function or father wavelet, is defined and associated with the mother wavelet  $\psi(t)$  to capture the rest of the signal. The scaling function at level  $m$  and shift  $n$  has similar form to the mother wavelet,

$$\phi_{m,n}(t) = 2^{-\frac{m}{2}} \phi(2^{-m} \cdot t - n) \quad (3.4)$$

The scaling function is orthogonal to itself, but not to its dilations. The smoothing of the time-series data is produced by the inner product of the time series with the scaling function. The obtained samples are called approximation coefficients (low frequencies) and are defined as

$$a_{m,n} = \sum_{t=0}^{N-1} x(t) \phi_{m,n}(t) \quad (3.5)$$

An approximation of the data at level  $m$  can be computed using,

$$x_m(t) = \sum_n a_{m,n} \phi_{m,n}(t) \quad (3.6)$$

Given the approximation coefficients  $a_{m_0,n}$  generated at level  $m_0$  chosen arbitrarily and the wavelet detailed coefficients  $d_{m,n}$  at levels  $1, 2, \dots, m_0$ , the final multiresolution representation of the data can be obtained as follows,

$$x(t) = \sum_n a_{m_0,n} \phi_{m_0,n}(t) + \sum_{m=1}^{m_0} \sum_n d_{m,n} \psi_{m,n}(t) \quad (3.7)$$

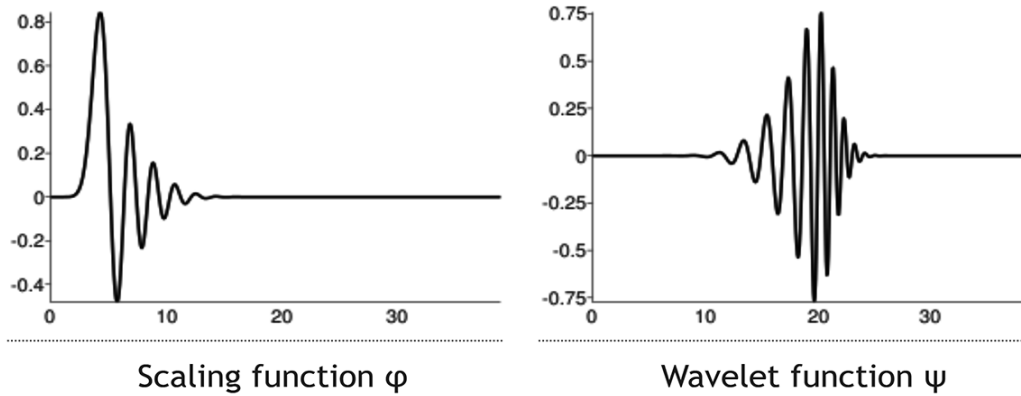


Figure 3.2: Wavelet and scaling functions

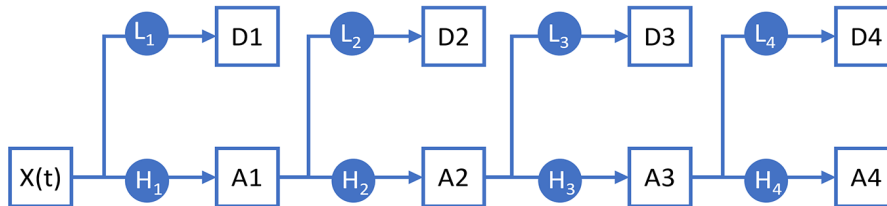


Figure 3.3: Decomposition block diagram of stationary wavelet transform

An example of wavelets is Daubechies wavelets which are orthogonal wavelets with a maximal number of vanishing moments for some given support. We used Daubechies-20 wavelet to perform the multiresolution analysis using stationary wavelet transform, where the index 20 refers to the number of coefficients. Figure 3.2 shows an illustration of the scaling and wavelet function<sup>1</sup>. The stationary wavelet transform is an extension of the typical discrete wavelet transform which is commonly used for exploratory statistical and signal analysis [88]. We noticed that this type of wavelet transformation is commonly used in time-series analysis and decomposition due to its shift-invariant

<sup>1</sup><https://github.com/PyWavelets/pywt>

property [13, 89]. The number of resolution levels used for data decomposition in many studies in the literature [13, 89, 90] is up to four levels. The block diagram of the undecimated wavelet decomposition representation is shown in Figure 3.3, where  $D_j$  and  $A_j$  represent the detail and approximation coefficients at level  $j$ , respectively. The high frequency components  $D_j$  are generated from the high-pass filter  $H_j$ , and the low frequency components  $A_j$  are generated from the low-pass filter  $L_j$ . Figure 3.4 shows an illustrative example of the data before applying wavelet decomposition and Figure 3.5 shows its corresponding multiresolution decomposition into four detailed coefficients and approximation coefficients at level four. It can be noticed that the higher the level of decomposition, the smoother the approximation coefficients and the lower the level of the detailed coefficients, the higher the captured frequencies.

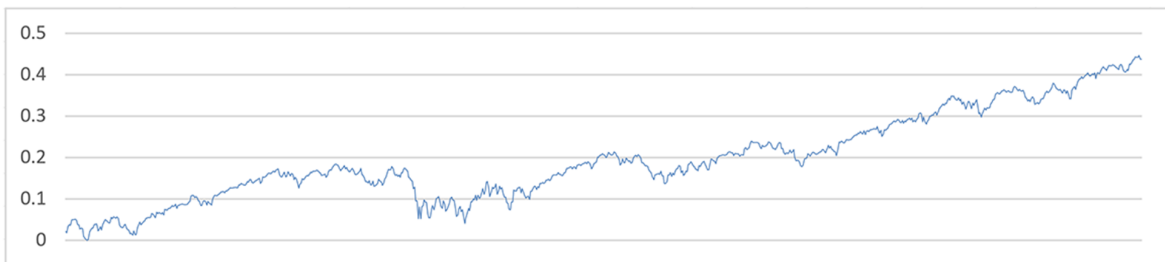


Figure 3.4: Plot of part of the actual S&P data before performing wavelet analysis

### 3.3.2 Empirical Wavelet Transform

Applying different types of wavelet transforms and multiresolution analysis methods aims at exploring and devising better forecasting approaches. In our study, we applied empirical wavelet transform to perform multiresolution analysis of the financial time series in order to build more effective forecasting models. EWT is an adaptive wavelet



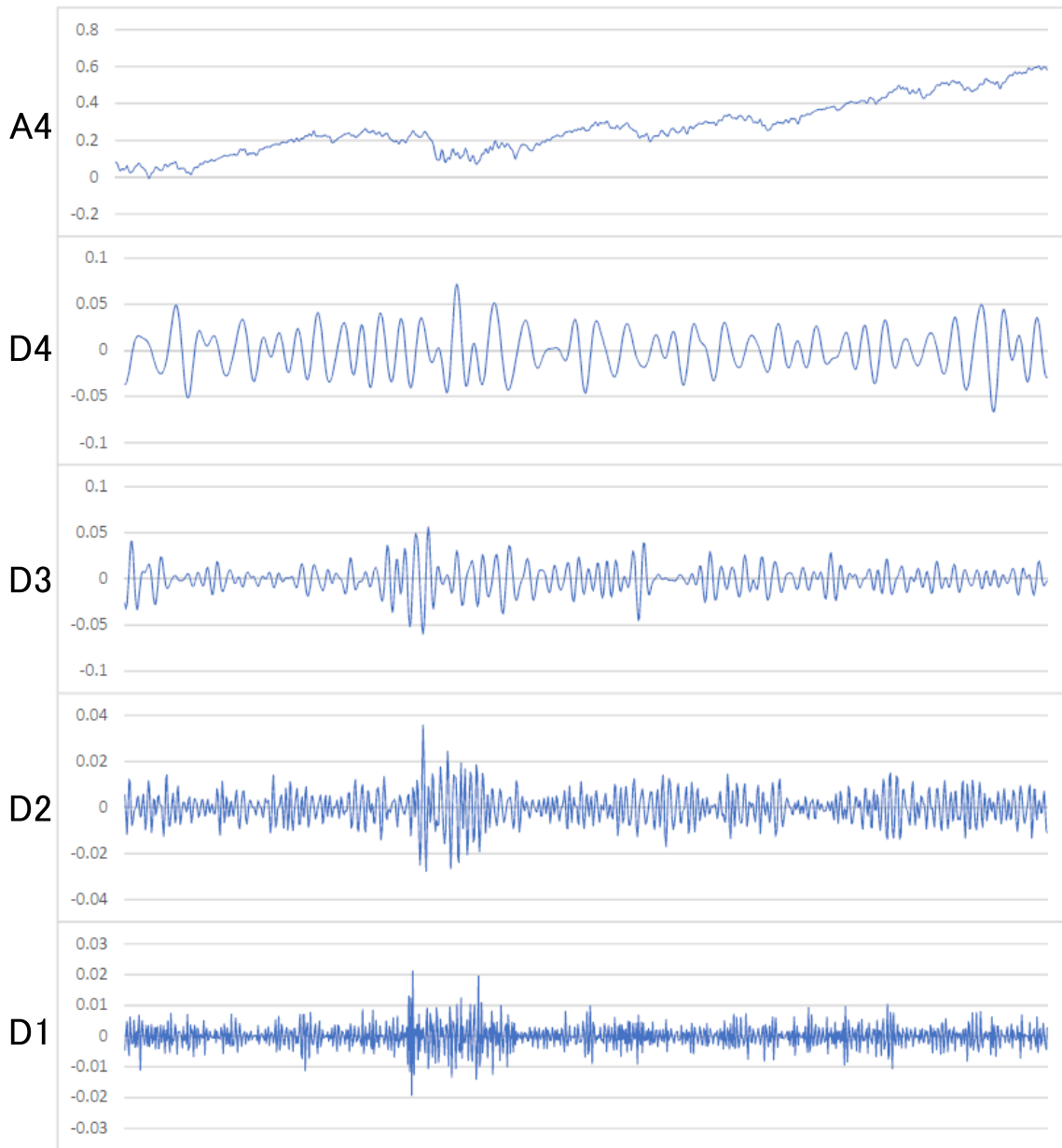


Figure 3.5: Stationary wavelet decomposition for part of the S&P data

transform developed by Jérôme Gilles [91]. The main advantage of this transform is that without any prior information about the data, it automatically analyzes the data and identifies a small number of coefficients to pack the signal information. It facilitates time-series processing and forecasting by generating higher time-frequency

resolution. EWT approach performs data analysis to define a set of adaptive filter banks extracted from the data based on its prominent frequency components. It identifies a set of maxima in the Fourier spectrum ( $X(\omega)$ ) of the signal with a set of corresponding frequency indices ( $\omega_n$ ) by defining frequency and magnitude thresholds. The range of the frequency axis  $[0 - \pi]$  is segmented into  $N$  segments defined as  $\Lambda_n = [\omega_{n-1}, \omega_n]$  using boundary values defined by a selected number ( $N$ ) of maxima [92]. The boundaries  $\omega_i$  are obtained by setting  $\omega_0 = 0$  and  $\omega_N = \pi$ . The Fourier segments will be  $[0, \omega_1], [\omega_1, \omega_2], \dots, [\omega_{N-1}, \pi]$ . The filter bank represented by  $N - 1$  band-pass filters and one low-pass filter, is constructed depending on set boundaries. Supports for filters can be calculated using the following equations [92],

$$S_n = (\omega_n - \omega_{n-1}) + 2\gamma\omega_n \quad (3.8)$$

where  $\gamma$  is small value between 0 and 1 defined to ensure that the two consecutive transition bands are not overlapping. The value of  $\gamma$  can be computed using the following equation,

$$\gamma \leq \min_n \left[ \frac{\omega_{n+1} - \omega_n}{\omega_{n+1} + \omega_n} \right] \quad (3.9)$$

The empirical scaling function  $\hat{\phi}_n(\omega)$  and wavelet function  $\hat{\psi}_n(\omega)$  are given by,

$$\hat{\phi}_n(\omega) = \begin{cases} 1 & \text{if } |\omega| \leq (1 - \gamma)\omega_n \\ \cos \left[ \frac{\pi}{2}\beta \left( \frac{1}{2\gamma\omega_n} (|\omega| - (1 - \gamma)\omega_n) \right) \right] & \text{if } (1 - \gamma)\omega_n \leq |\omega| \leq (1 + \gamma)\omega_n \\ 0 & \text{otherwise} \end{cases} \quad (3.10)$$

$$\hat{\psi}_n(\omega) = \begin{cases} 1 & \text{if } (1 + \gamma)\omega_n \leq |\omega| \leq (1 - \gamma)\omega_{n+1} \\ \cos \left[ \frac{\pi}{2}\beta \left( \frac{1}{2\gamma\omega_{n+1}} (|\omega| - (1 - \gamma)\omega_{n+1}) \right) \right] & \text{if } (1 - \gamma)\omega_{n+1} \leq |\omega| \leq (1 + \gamma)\omega_{n+1} \\ \sin \left[ \frac{\pi}{2}\beta \left( \frac{1}{2\gamma\omega_n} (|\omega| - (1 - \gamma)\omega_n) \right) \right] & \text{if } (1 - \gamma)\omega_n \leq |\omega| \leq (1 + \gamma)\omega_n \\ 0 & \text{otherwise} \end{cases} \quad (3.11)$$

The function  $\beta(x)$  can be defined using the following equation,

$$\beta(x) = x^4(35 - 84x + 70x^2 - 20x^3) \quad (3.12)$$

Other functions can be used as long as they satisfy the following condition,

$$\beta(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ \beta(x) + \beta(1 - x) = 1 & \text{if } x \in [0, 1] \\ 1 & \text{if } x \geq 1 \end{cases} \quad (3.13)$$

Based on this defined set of band filters, the EWT can be defined in a similar way as the normal wavelet transform. The approximation coefficients are obtained by the inner product of applied signal with the empirical scaling function as [92],

$$W_x(0, t) = \langle x, \phi_1 \rangle = IFFT(X(\omega)\Phi_1(\omega)) \quad (3.14)$$

The inner product of the empirical wavelets with applied signal produces the detailed coefficients as,

$$W_x(n, t) = \langle x, \psi_n \rangle = IFFT(X(\omega)\Psi_n(\omega)) \quad (3.15)$$

We applied EWT on part of the S&P financial time series, which is shown in Figure 3.4, and the resulting decomposition is shown in Figure 3.6.

### 3.4 Deep Learning Based Models

Deep learning networks are developed using several stacked hidden layers to train on huge amount of data and massive computing power. Each intermediate layer extracts certain kind of information from the data and redirects the learned features to the following layers to perform another type of information extraction. A hierarchy of patterns are extracted from the training data and employed to perform the forecasting process. The decomposition of the time series patterns into sub-patterns enables learning complicated hierarchies out of simpler ones. The input features are fed into the deep neural network to adjust the network parameters [93].

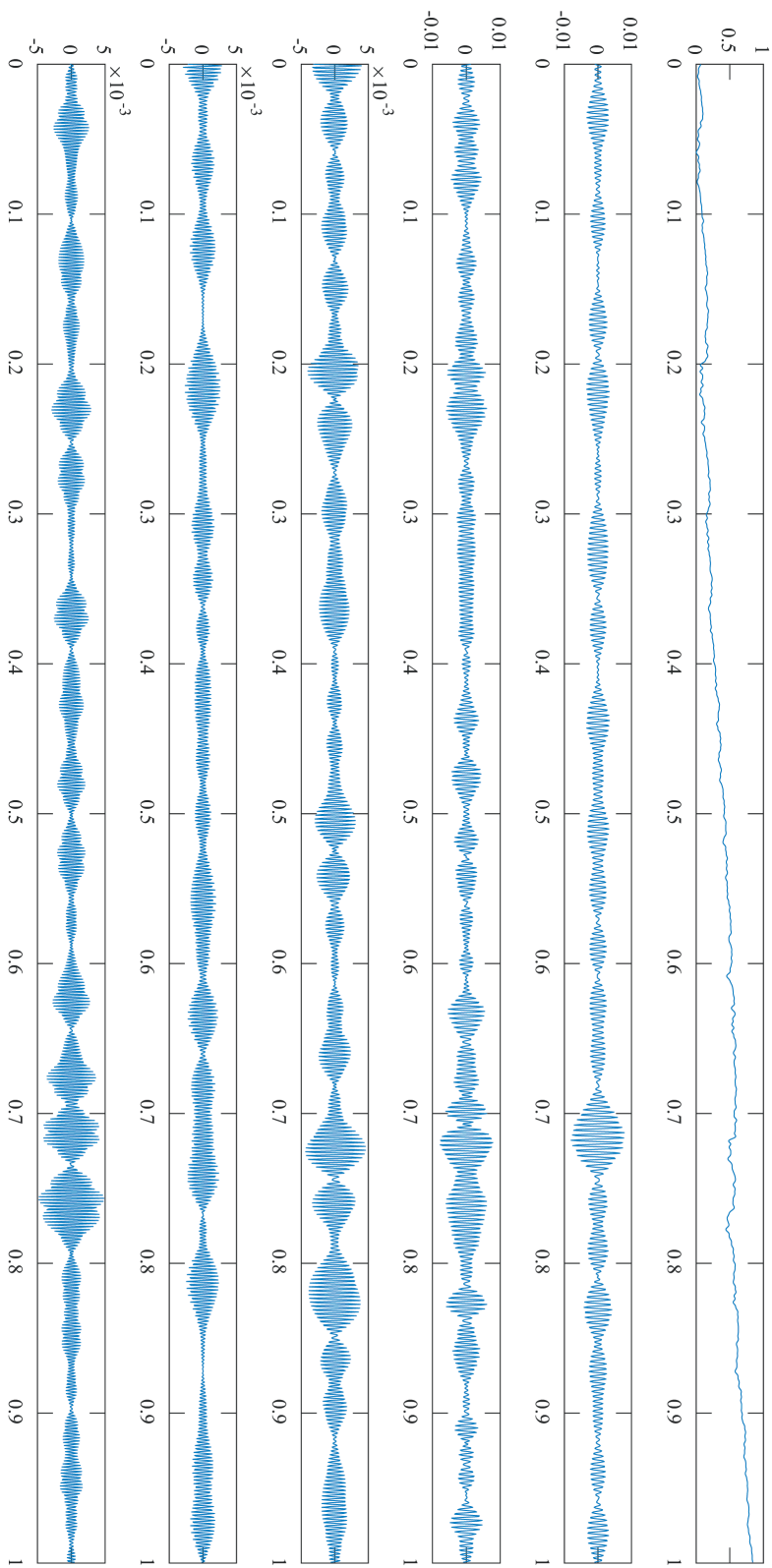


Figure 3.6: Adaptive wavelet decomposition for part of the S&P data.

Defining best learning network architecture requires conducting several experiments to deduce the best network type and parameters that produce the highest accuracy on the presented learning data. Based on different sub-series generated from the multiresolution multiscale analysis and technical analysis, several instances of the deep learning networks are trained and evaluated to select the best network that has the minimum error and best accuracy of forecasting future stock prices. We developed different bidirectional and stacked architectures of deep recurrent neural networks such as LSTM and GRU to forecast financial time series.

The proposed learning architecture as illustrated in Figure 3.7, is divided into three stages. The first stage performs data analysis and decomposition into many sub-series. The second stage represents the learning phase which includes several deep recurrent neural networks trained on the resolution levels produced by the first stage. The forecasts of each network associated to each resolution level in addition to set of TIs are fed into another deep recurrent neural network to perform the final learning stage and produce the final output. Input features are formed according to various time lags adopted in this work. At time step  $t$ , we use a sliding window of  $k$  past values  $(x_{t-k+1}, x_{t-k+2}, \dots, x_{t-1}, x_t)$  to forecast the  $x_{t+h}$  in the future, where  $h$  is the forecast horizon. Figure 3.8 illustrates the methodology used for reforming the data.

### **3.4.1 Deep Recurrent Neural Networks (RNNs)**

Recurrent Neural Network (RNN) constructs learning architectures using current and preceding input data to adjust network weights. The network learns by cascading forward through sequences of input data. The hidden states of the network can preserve

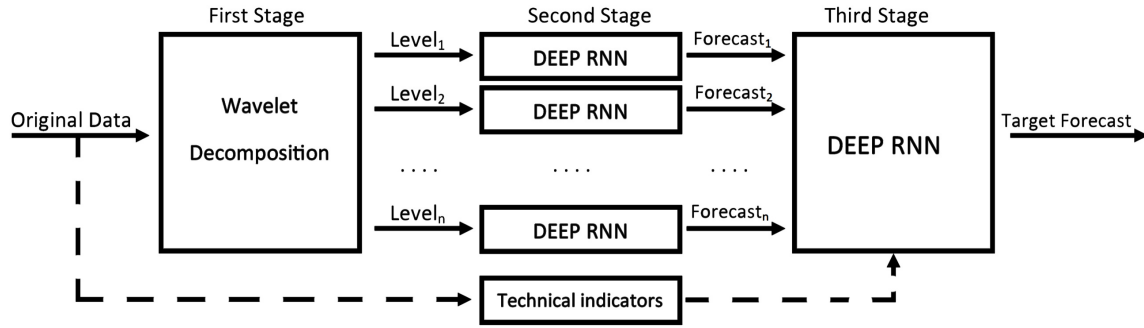


Figure 3.7: Framework of the proposed methodology

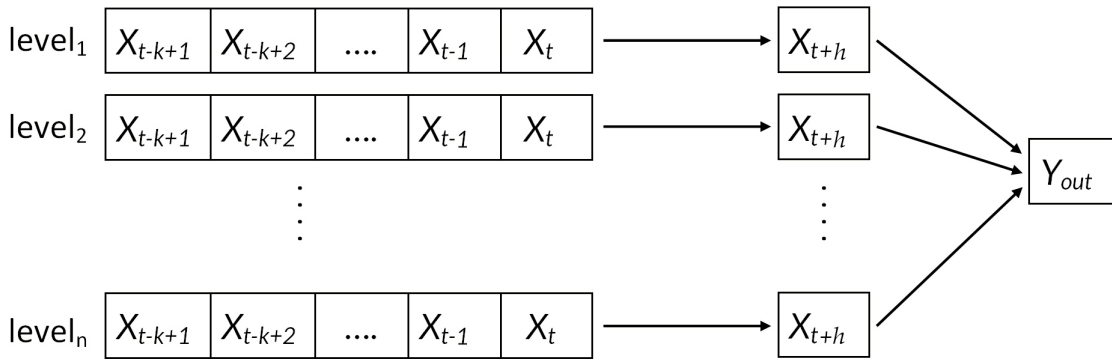


Figure 3.8: Sliding window and input features reshaping to perform forecast

information which is used in the training operation. Backpropagation Through Time (BPTT) is the training algorithm used by recurrent neural networks. It processes data sequences according to its order by linking data element at each time step to data elements in preceding time steps [94].

With the remarkable success of deep learning, deep RNN is becoming popular to process sequential data. It is characterized by the ability to preserve network preceding states and information is retained in its hidden network cells while cascading forward through data sequences [94]. We adopt two deep RNN models (LSTM and GRU) for constructing financial time-series forecasting models.

### 3.4.2 Long-Short Term Memory (LSTM)

Capturing long-term dependencies is infeasible using RNN due to the vanishing gradient problem. LSTM solves this problem by enhancing the hidden states of the RNN to remember longer sequences of data. It is devised by Hochreiter and Schmidhuber [95] by the addition of input, forget, and output gates that control information to or out of the memory cells using point-wise multiplication and sigmoid neural network layer. The gated cells act on the received current input data by revoking or letting information pass based on the import of the data element to the target value. A graphical illustration of a basic LSTM is shown in Figure 3.9 (a)<sup>2</sup>. Its transition equations are as follows [96]:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1}) \quad (3.16)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + V_f c_{t-1}) \quad (3.17)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_o c_t) \quad (3.18)$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1}) \quad (3.19)$$

$$c_t = f_t^i \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (3.20)$$

$$h_t = o_t \odot \tanh(c_t) \quad (3.21)$$

where  $i_t$  denotes the input gate,  $o_t$  denotes the output gate, and  $f_t$ ,  $c_t$ , and  $h_t$  denote the forget gate, memory cell, and hidden state, respectively.

<sup>2</sup><https://isaacchanghau.github.io/post/lstm-gru-formula/>



### 3.4.3 Gated Recurrent Unit (GRU)

GRU is an extended development of LSTM. The network architecture consists of blocks of gated recurrent units to control memory reset and update. GRU achieves comparable performance to that of LSTM, but uses less number of parameters, which makes it faster to train. The only gates used in GRU are the update and reset gates. The update gate is responsible for renewing the current memory of the network which enables the network to remember certain data input based on its importance. The reset gate is responsible of deleting the current memory of the network, which allows the network to forget certain values at any time step. The transition equations in hidden units of GRU are given as follows [97]:

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (3.22)$$

$$\tilde{h}_t = \tanh(W_h x_t + U_h (r_t \odot h_{t-1}) + b_h) \quad (3.23)$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \quad (3.24)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \quad (3.25)$$

where  $h_t$  and  $h_{t-1}$  denote the output of the current and previous states, respectively.  $r_t$  and  $z_t$  denote the reset and update gates, respectively. An illustration of the internal architecture of GRU is shown in Figure 3.9 (b)<sup>3</sup>.

<sup>3</sup><https://isaacchanghau.github.io/post/lstm-gru-formula/>

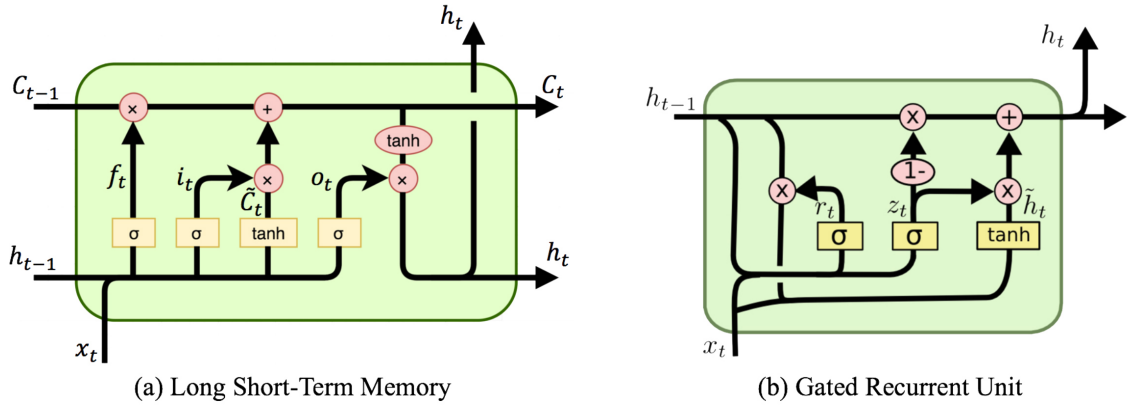


Figure 3.9: Illustration of the internal architecture of the LSTM and GRU hidden units

### 3.4.4 Stacked and Bidirectional RNNs

We developed two architectures by stacking deep recurrent layers as illustrated in Figure 3.10 and 3.11. The first two layers in the first architecture can use LSTM or GRU layers and process the data in the same direction. The second architecture is similar but first two layers process the data in opposite directions. One layer performs the operations following the same flow direction of the data sequence whereas the other layer is reversely applying its operations on the data sequence. Neurons in the first two layers are denoted by  $h_1$  to  $h_n$ , where  $n$  is the number of neurons. The output of the network is computed using a dense layer with linear activation function.

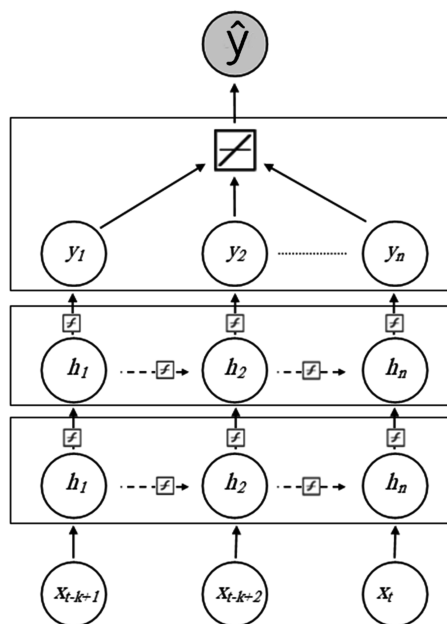


Figure 3.10: Stacked architecture used to train deep RNN

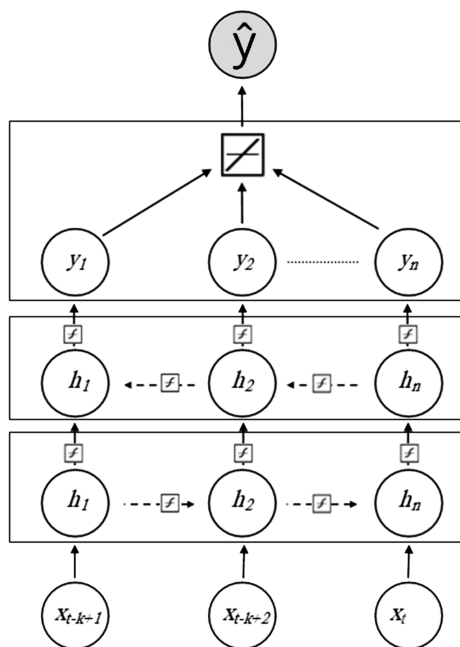


Figure 3.11: Bidirectional architecture used to train deep RNN

## CHAPTER 4

# DATA EXPLORATION AND PREPROCESSING

In this chapter, we discuss benchmark datasets and perform data exploration and analysis. Three financial market datasets are used in our experiments as described in Section 4.1. The data is transformed using natural log as described in Section 4.2. Then the raw and transformed financial data are analyzed by applying autocorrelation, randomness tests, and white noise tests in Sections 4.3, 4.4, and 4.5, respectively.

### 4.1 Benchmark Datasets

Recording stock market data is based on a defined time period such as daily, weekly, and monthly basis. We use daily data in the experiments conducted to meet the objectives of this thesis. Daily closing price is one of the variables recorded to summarize shares trade transactions. It refers to the traded stock price at the end of the day. It is commonly used by researchers as the target of the forecasting models [98–100].

Most stock market traders depend on the closing price in the estimation and analysis process. They use it to derive expectation of future trading prices. More information about stock market data and financial time series is provided in Appendix B.

The evaluation experiments conducted by this study use S&P, DJIA, and TASI stock market indices. Both S&P and DJIA belong to US stock market while TASI is an indicator of Saudi stock market. S&P and DJIA are benchmark datasets commonly employed to evaluate and compare stock market forecasting models. They are widely used to evaluate many forecasting architectures in literature [14, 101–105]. The purpose of using three datasets in the evaluation process, is to make sure that the evaluation process is not biased by or dependent on a certain single dataset and the results can be generalized to stock market and financial time-series forecasting. We use the historical data of the S&P, DJIA and TASI indices to train and evaluate the forecasting models developed by this work. The daily data of the datasets are downloaded from Yahoo finance for period from 01/01/2010 to 29/06/2018 which are used in most of the experiments conducted by this study. Figures 4.1, 4.2, and 4.3 show the data curve of the daily closing price of the three datasets. Some statistical analysis of the data is included in Table 4.1. Description of the three datasets is as follows,

- The S&P dataset is one of the important benchmark and leading indicator for stocks of 500 companies in the U.S. market. It represents the market capitalizations of 500 large companies in the American stock market which have common stock listed on the NASDAQ or New York Stock Exchange (NYSE).

Many traders consider S&P as one of the best commonly followed stock indices, and one of the best representations of the U.S. stock market. It was initially declared when US introduced its first stock index in 1923, and began with a limited number of stocks. After three years, it expanded to 90 stocks and then in 1957 it grew to include its current 500 stocks. It uses the market capitalization of each company based on the number of shares available for public trading.

- The DJIA represents an indication of 30 large publicly owned companies traded on the NYSE and NASDAQ. It is a price-weighted average calculated from the sum of the price of single share of stock for each of the 30 companies. The sum changes whenever one of the companies has a stock split or stock dividend. The DJIA was invented by Charles Dow in 1896 and named after him and his business partner Edward Jones.
- The Tadawul All-Share Index (TASI) is a major stock market index which tracks the performance of all companies listed on the Saudi Stock Exchange. The index has a base value of 1000 as of 1985 and it was reorganized in 2008. Tadawul is the sole entity authorized in the Kingdom of Saudi Arabia to act as the Securities Exchange. It mainly carries out listing and trading in securities, as well as deposit, transfer, clearing, settlement, and registry of ownership of securities traded on the Exchange.

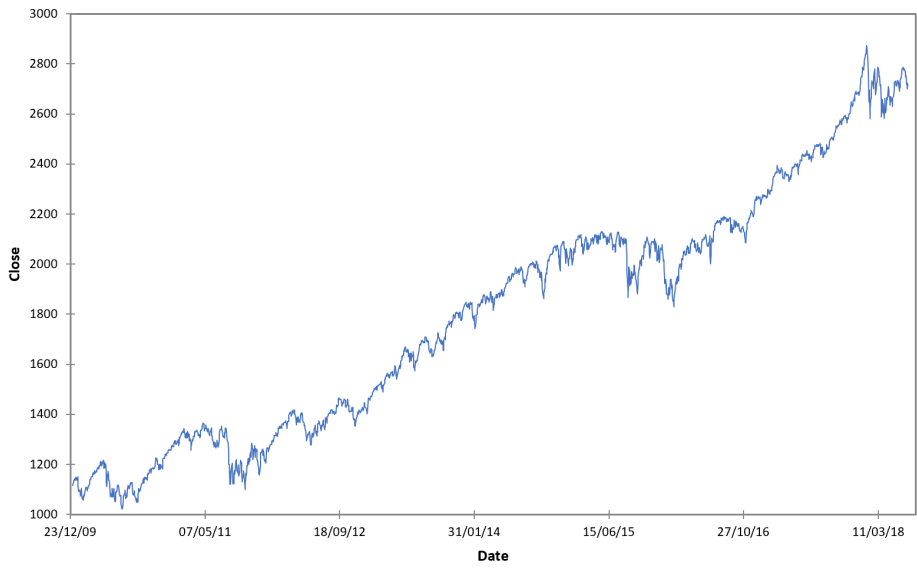


Figure 4.1: Plot of the closing price of S&P index

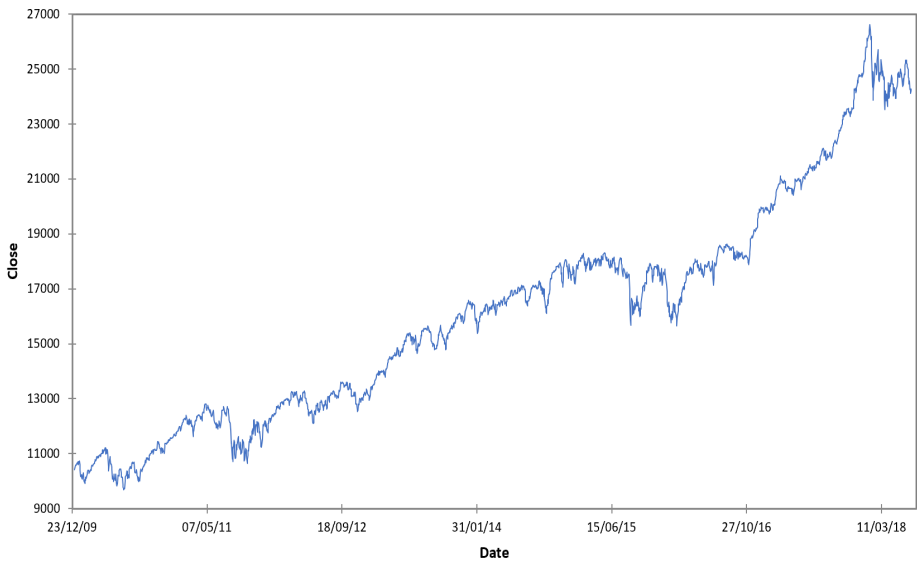


Figure 4.2: Plot of the closing price of DJIA index

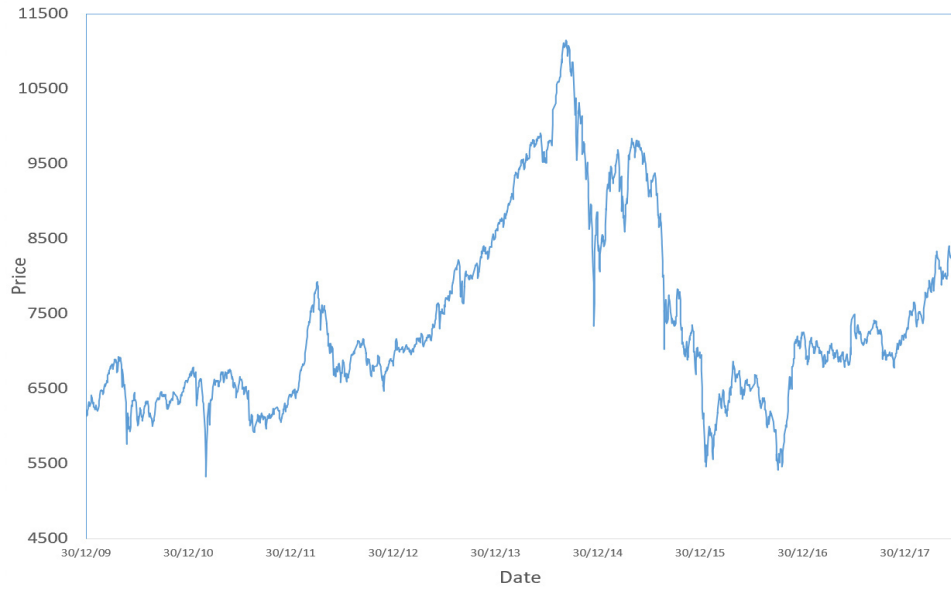


Figure 4.3: Plot of the closing price of TASI index

Table 4.1: Statistical summary for benchmark datasets

Dataset	Observations	Minimum	Maximum	Mean	Std. deviation
S&P	2139	1022.58	2872.870117	1802.39172	483.452688
DJIA	2139	9686.48	26616.71094	16118.34461	4031.16452
TASI	2119	5323.27	11149.36	7384.26	1195.4698



## 4.2 Data Transformation

Most financial time series are not stationary in their original observations. Generally, stationary time series is characterized by constant statistical properties over time. The mean, variance, and autocorrelation are constant in the future as in the past. Financial time series can be transformed into stationary using mathematical transformations to ease forecasting. The transformation process can be reversed to reconstruct the original time series from forecasts using the inverse of the transformation formula. Basically, one of the methods which is widely used to transform trending stock market time series into stationary is using the financial return of prices. Using returns normalizes the data through processing all variables in a comparable metric. Despite the different value measuring ranges of the price series, it enables evaluation of analytic relationships amongst two or more variables. Total returns  $r_t$  at time  $t$  for period  $t-h$  is calculated using prices  $x_t$  and  $x_{t-h}$  by the following equation:

$$r_t = \frac{x_t - x_{t-h}}{x_{t-h}} \quad (4.1)$$

$$1 + r_t = \frac{x_t}{x_{t-h}} \quad (4.2)$$

We transform benchmark time series into stationary using the natural log transformation of stock returns. It helps making the seasonal fluctuations more consistent over time, this means that the model can fit the data more accurately. The log return  $R_t$  of the stock closing price at day  $t$  is calculated by computing the natural logarithm

for the ratio between the closing price  $x_t$  at day  $t$  and at day  $t - h$ , where  $h$  denotes the future time scale used to perform the forecast.

$$R_t = \ln(1 + r_t) = \ln\left(\frac{x_t}{x_{t-h}}\right) = \ln(x_t) - \ln(x_{t-h}) \quad (4.3)$$

In our work we conduct experiments to investigate the performance of the forecast based on both raw closing prices and log returns of closing prices. Plots of the log returns of the three datasets are shown in Figure 4.4.

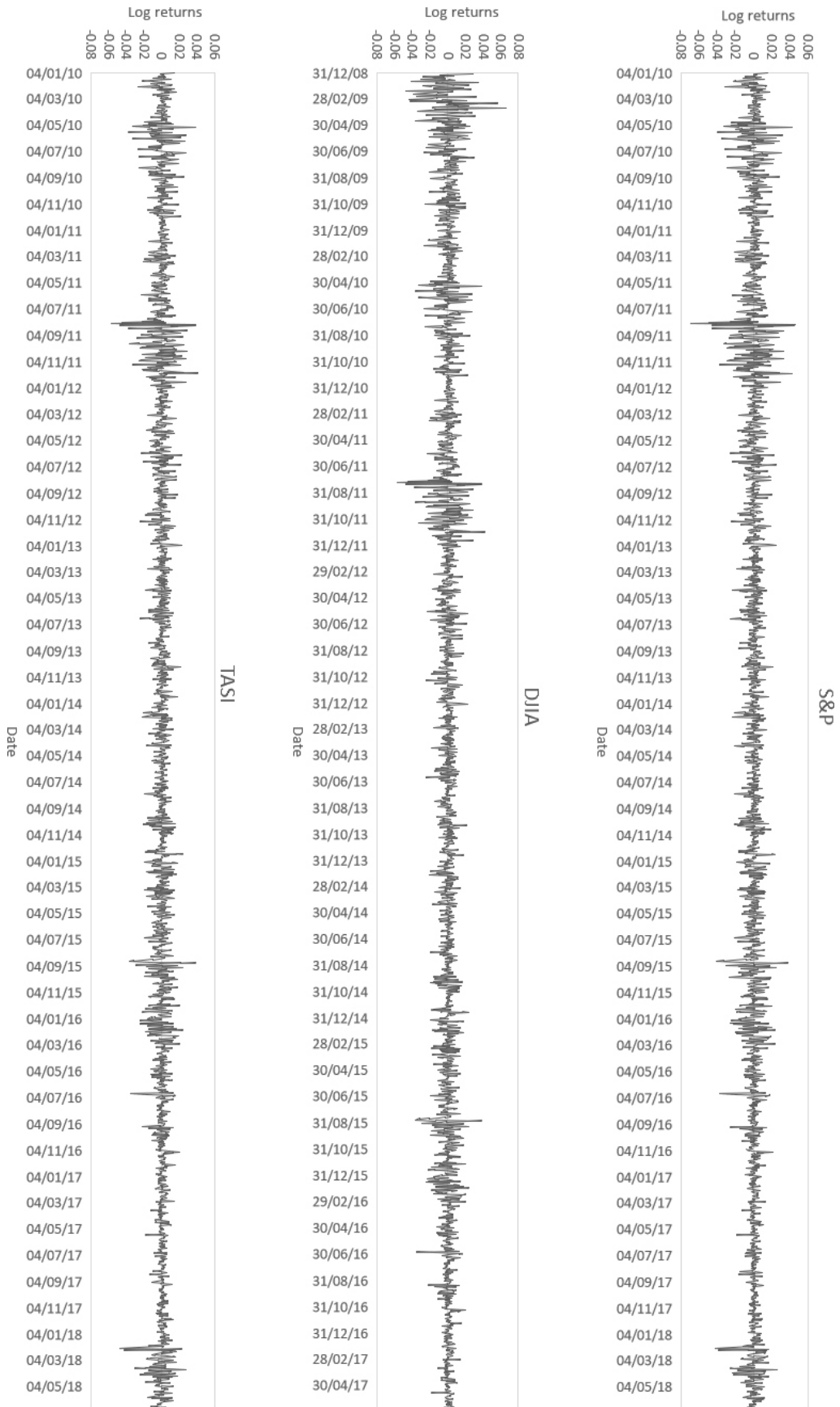


Figure 4.4: Plots of the log returns of the three datasets

## 4.3 Autocorrelation Analysis

Time series generated from random process observations are serially uncorrelated, in that using linear modeling of the past observation variables to account for the behavior of the current variable may not be feasible. Evidently, a serially independent time series implies that the relation between past information and current variable does not exist. Therefore, applying autocorrelation analysis on raw data series provides information which may help improve data modeling by characterizing relations between time lags of time series.

### 4.3.1 Autocorrelation Plot (Correlogram)

It is a graphical representation of the autocorrelation function/coefficient at different time lags denoted as  $\hat{\rho}(k)$ . For  $x_1, x_2, \dots, x_n$ , observations recorded at equally spaced times  $t_1, t_2, \dots, t_n$ , the autocorrelation coefficients (vertical axis) are calculated by the following formulas <sup>1</sup> [106]:

$$\hat{\rho}(k) = C_k/C_0 \tag{4.4}$$

$$C_k = \frac{1}{n} \sum_{t=1}^{n-k} (x_t - \bar{x})(x_{t+k} - \bar{x}) \tag{4.5}$$

<sup>1</sup><https://www.itl.nist.gov/div898/handbook/eda/section3/autocopl.htm>

$$C_0 = \frac{\sum_{t=1}^n (x_t - \bar{x})^2}{n} \quad (4.6)$$

where  $k$  denotes time lags (horizontal axis),  $n$  denotes length of the time series, and  $\bar{x}$  is the mean.  $\hat{\rho}(k)$  always takes values between  $-1$  and  $+1$ .

The autocorrelation estimates the serial interdependency between time lags of the time series. Therefore, we employ it as an indicator of randomness. If the time series exhibits a random pattern, the autocorrelation should be near zero for any and all time lags, otherwise, one or more of the time lags autocorrelations shall be non-zero based on level of significance.

### 4.3.2 Partial Autocorrelation Plot

The partial autocorrelation function (PACF) gives the partial correlation of a time series with its own lagged values, controlling for the values of the time series at all shorter lags. It contrasts with the autocorrelation function, which does not account for other lags. For a time series, the partial autocorrelation between  $x_t$  and  $x_{t-k}$  is defined as the conditional correlation between  $x_t$  and  $x_{t-k}$ , conditional on  $x_{t-k+1}, \dots, x_{t-1}$ , the set of observations that come between the time points  $t$  and  $t - k$ .

The first order partial autocorrelation will be defined to be equal to the autocor-

relation. The  $2^{nd}$  order (lag) partial autocorrelation is defined as follows<sup>2</sup> .

$$\frac{\text{Covariance}(x_t, x_{t-2}|x_{t-1})}{\sqrt{\text{Variance}(x_t|x_{t-1})\text{Variance}(x_{t-2}|x_{t-1})}} \quad (4.7)$$

The third order (lag) partial autocorrelation is defines as follows.

$$\frac{\text{Covariance}(x_t, x_{t-3}|x_{t-1}, x_{t-2})}{\sqrt{\text{Variance}(x_t|x_{t-1}, x_{t-2})\text{Variance}(x_{t-3}|x_{t-1}, x_{t-2})}} \quad (4.8)$$

and so on for the partial autocorrelation of the other lags.

### 4.3.3 Autocorrelation Analysis Results

We calculated the autocorrelation plot (ACF) and partial autocorrelation plot (PACF) for both datasets. Initially, we perform analysis using raw data consisting of the closing price of both datasets for period from 01/01/2010 to 29/06/2018 which are used in most of the experiments. The autocorrelation plots are shown in Figures 4.5, 4.7, and 4.9 while the partial autocorrelation are shown in Figures 4.6, 4.8, and 4.10

It is obvious that there is a strong correlation between time lags for both time series. According to autocorrelation plots, we notice that the further the time lags of the target variable, the less its correlation significance. We may also notice from the partial autocorrelation plots that the first time lag has the highest correlation significance. It indicates that there is a significant correlation at lag one followed by correlations that are not significant which may indicate that time lag one represents

<sup>2</sup><https://onlinecourses.science.psu.edu/stat510/node/62/>

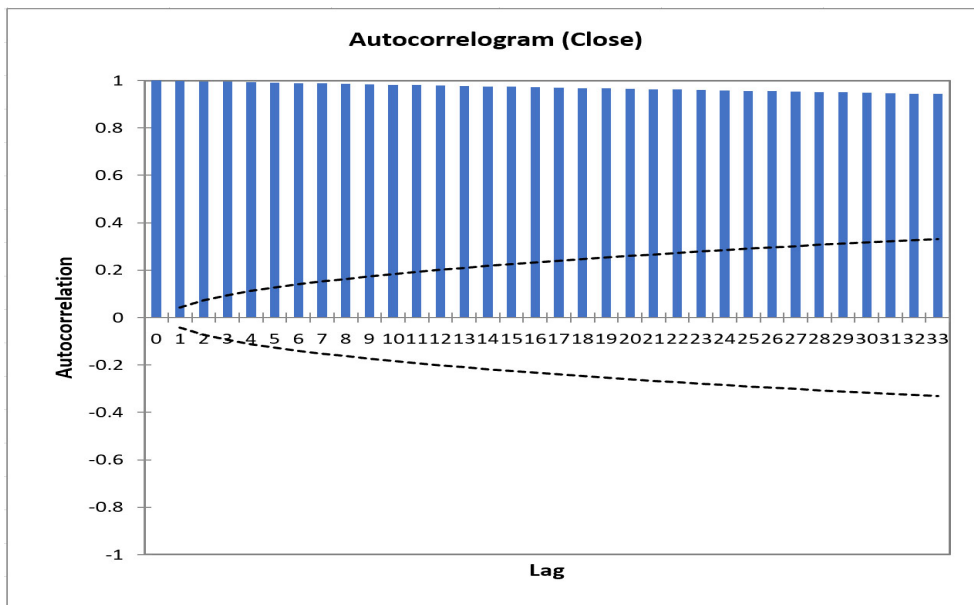


Figure 4.5: Autocorrelation plot of the S&P closing price

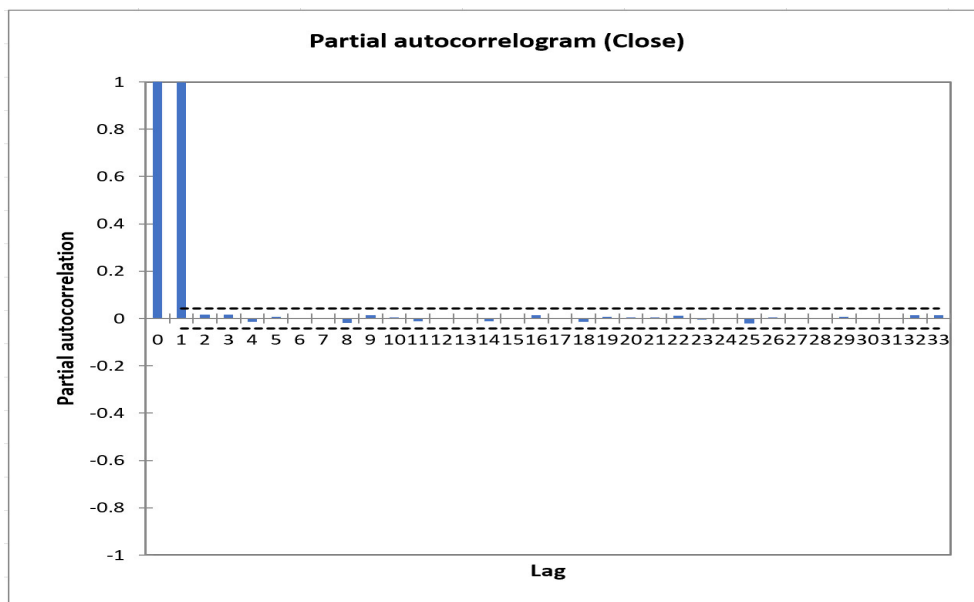


Figure 4.6: Partial autocorrelation plot of the S&P closing price

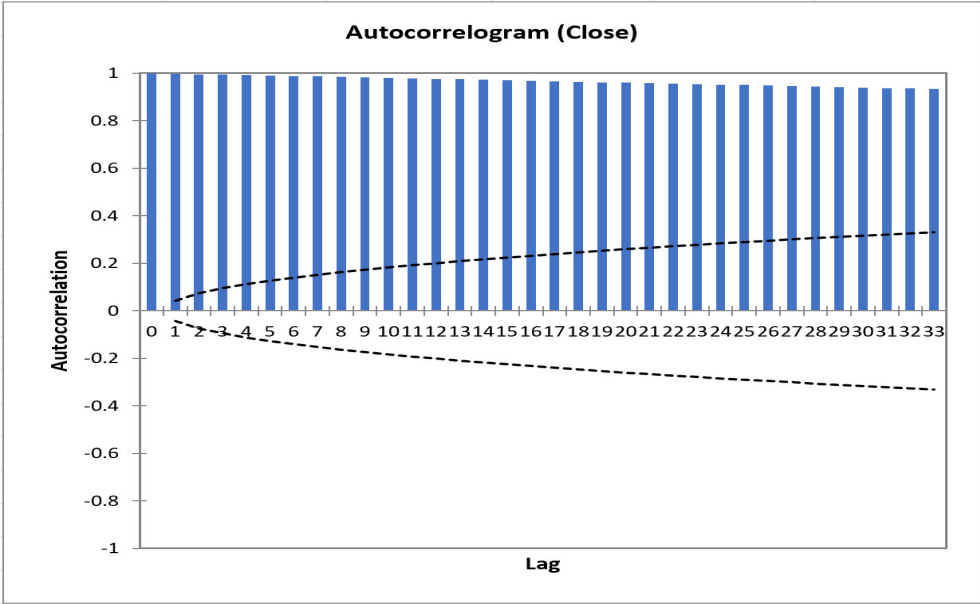


Figure 4.7: Autocorrelation plot of the DJIA closing price

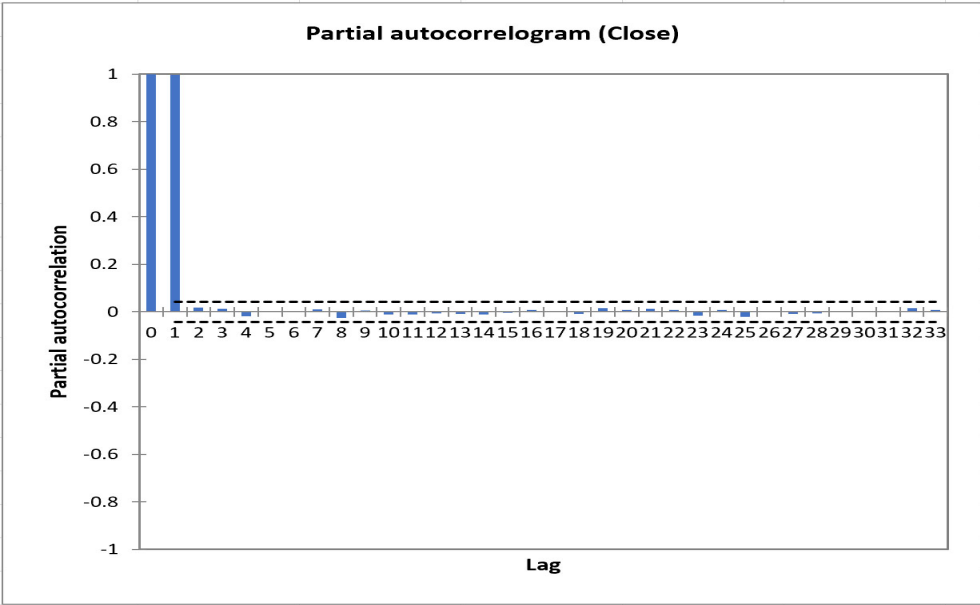


Figure 4.8: Partial autocorrelation plot of the DJIA closing price



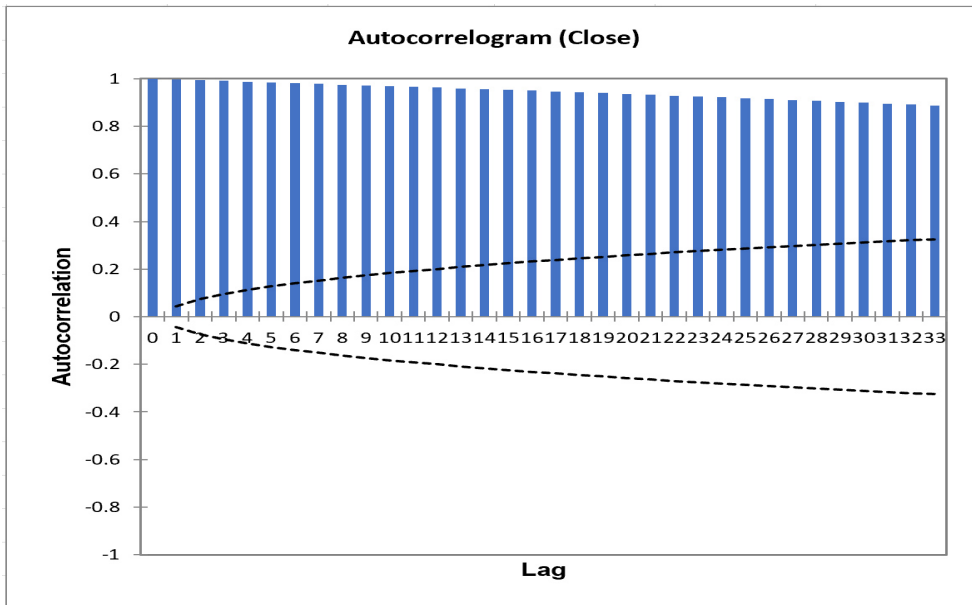


Figure 4.9: Autocorrelation plot of the TASI closing price

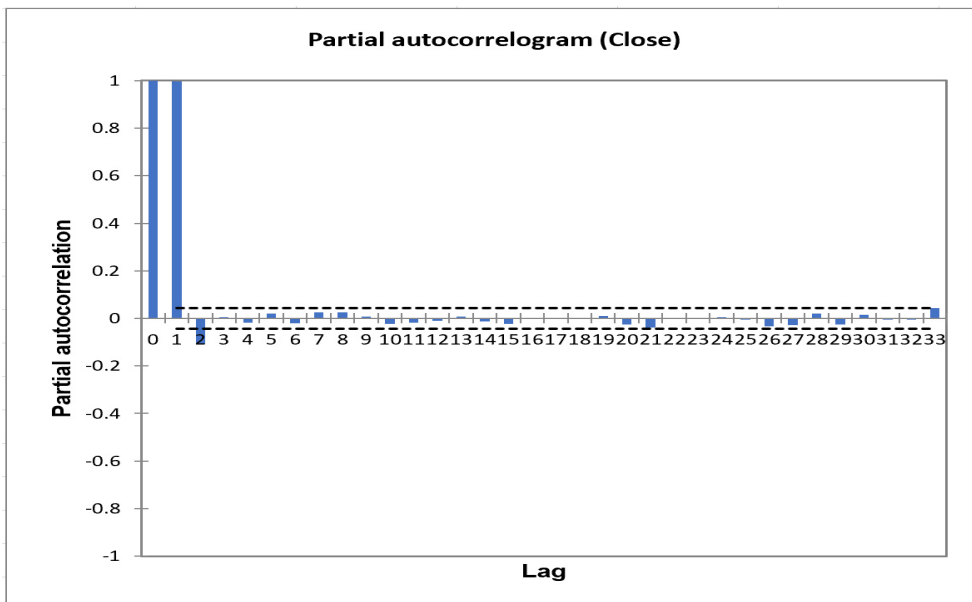


Figure 4.10: Partial autocorrelation plot of the TASI closing price

most of the data needed to perform short-term forecasting. However, performing analysis using different forms of the data may help investigating data patterns and clarify relation between explanatory variables of time series. One of the most common useful data transformation is the natural log transformation which makes the variance more homogeneous throughout the sample [107]. To perform deep analysis to data, we stationarize time series using the first difference of the log transformed data. A stationarized series is relatively easy to predict since its statistical properties are constant and stable during the past and future as well. We apply equations discussed in Section 4.2 to calculate the log return of closing price. The autocorrelation plots of the natural log returns of datasets are illustrated in Figures 4.11, 4.13, and 4.15, while the partial autocorrelation plots are illustrated in Figures 4.12, 4.14, and 4.16.

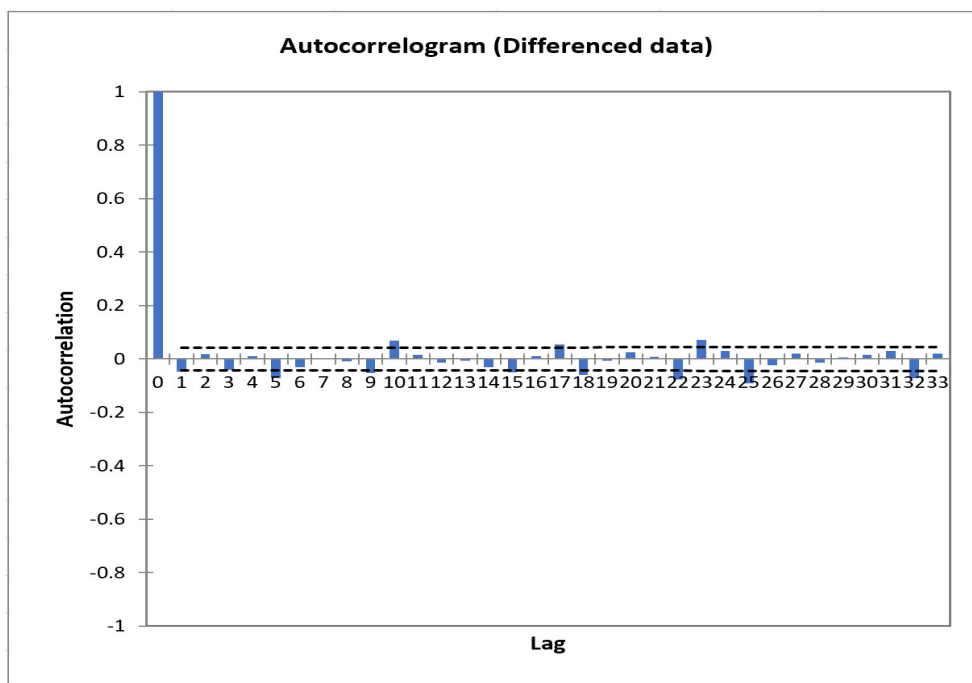


Figure 4.11: Autocorrelation plot of the log returns of S&P closing price

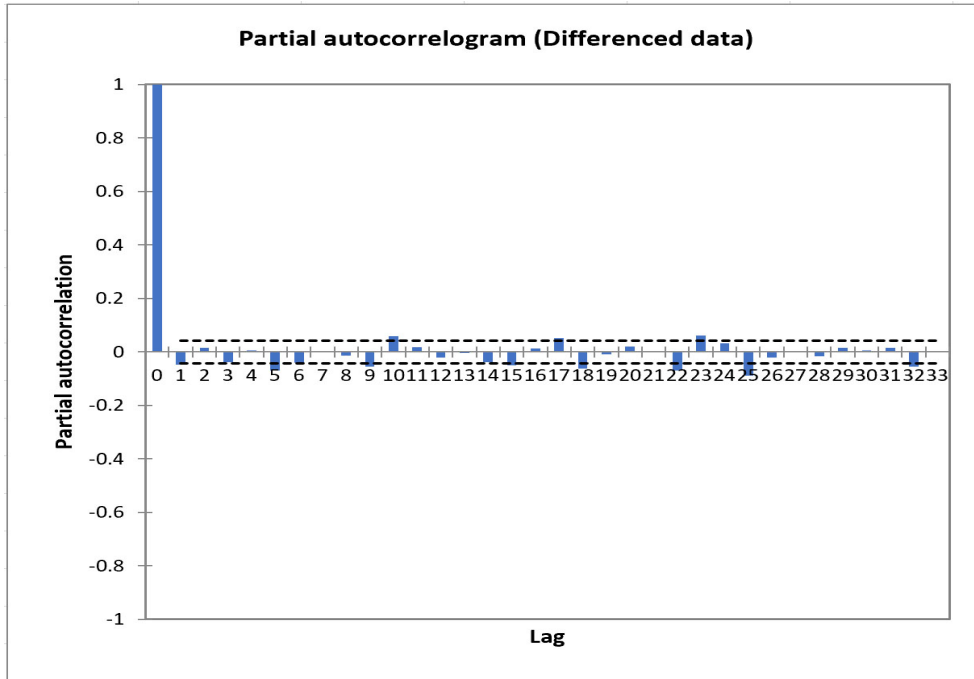


Figure 4.12: Partial autocorrelation plot of the log returns of S&P closing price

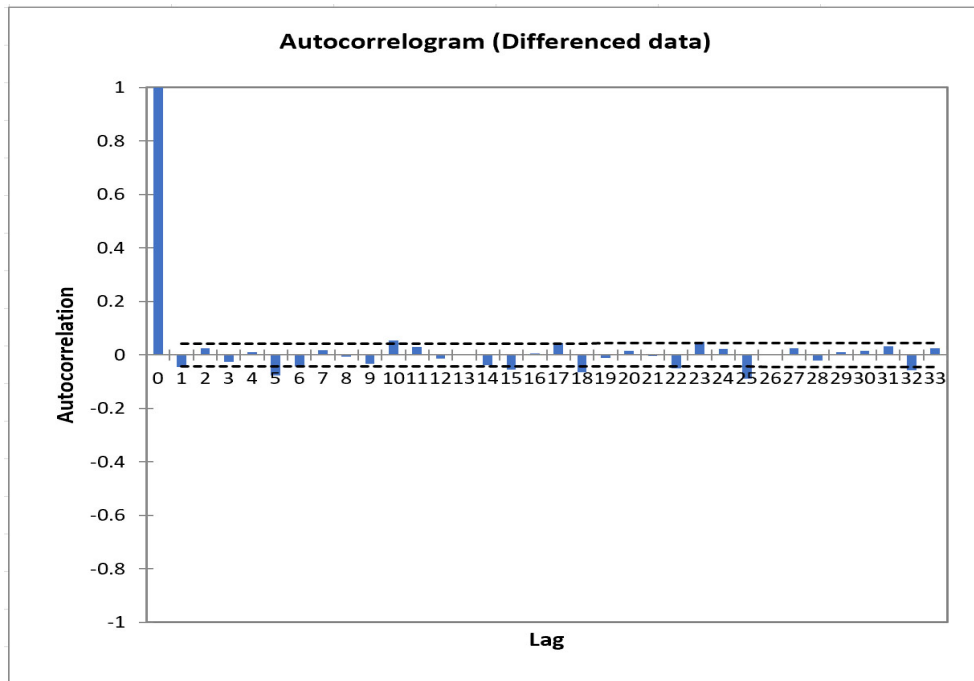


Figure 4.13: Autocorrelation plot of the log returns of DJIA closing price

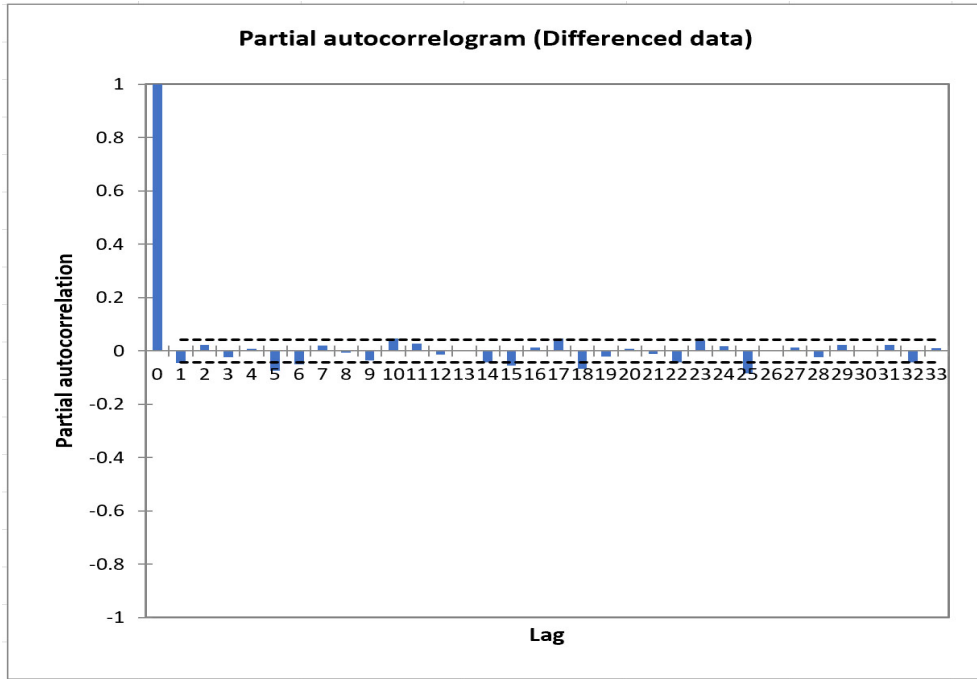


Figure 4.14: Partial autocorrelation plot of the log returns of DJIA closing price

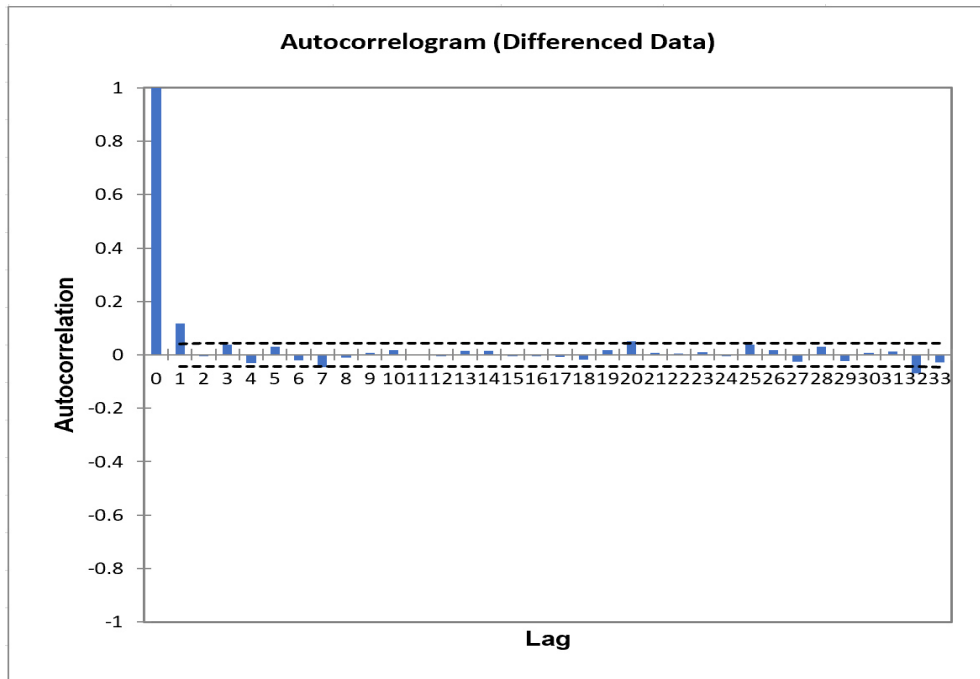


Figure 4.15: Autocorrelation plot of the log returns of TASI closing price.

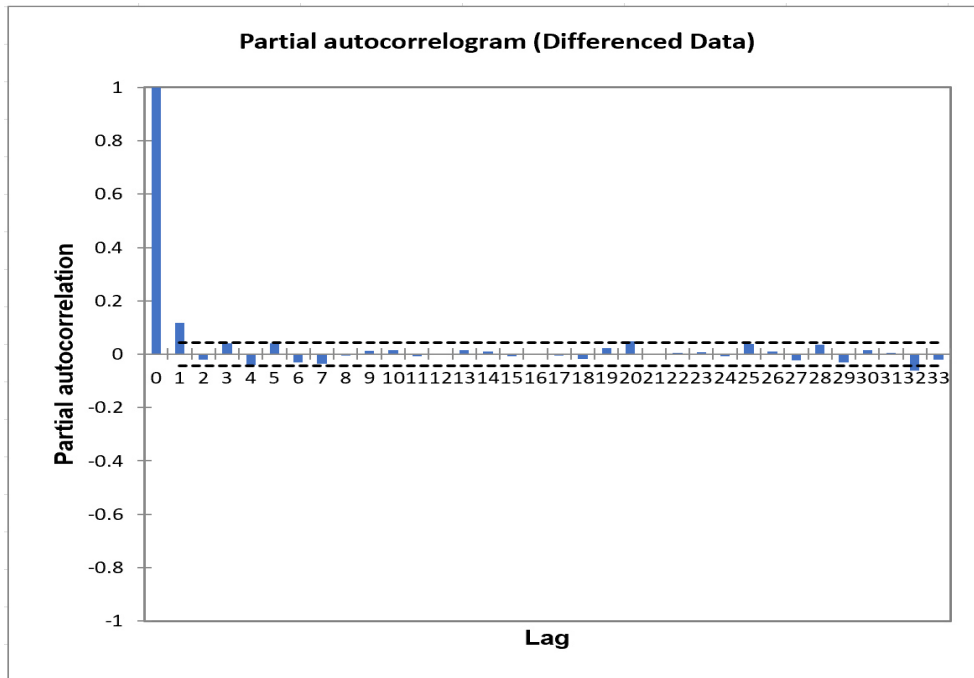


Figure 4.16: Partial autocorrelation plot of the log returns of TASI closing price

The autocorrelation analysis figures of the log returns data of S&P and DJIA show that some of the time lags, such as 1, 5, 9, 10, 18, and 25, show a small level of significance reach to or beyond confidence bands which may represent an indication of predictable pattern in the structure of the data. We also notice that both datasets exhibit semi-identical analysis results due to the relation between the two stock market indices since they represent the same market. Therefore, we integrated TASI dataset which represents the Saudi stock market into some of the experiments conducted by this study to perform the forecasting. The results of the autocorrelation analysis of log returns of TASI data show that time lag one has high significant impact reach beyond confidence bands.

## 4.4 Randomness Test

Random models are the simplest time series, in which the variance is constant and data points vary around a constant mean, and represent independent identical observations. Mainly, a random time series does not have any predictable pattern and values are not trending, meaning that the variance does not increase with time. Developing a system to predict random sequence of observations without prior knowledge of the sequence may not be possible [108, 109]. It means that using past information of a random time series to forecast the future will not produce an accuracy better than that produced by the random walk model. According to the efficient market hypothesis, stock market time-series values are independent of each other and the best way to forecast the price of tomorrow is to use the price of today which implies the random walk model. Applying randomness tests to stock market time series may support the predictability of stock market time series. One of the most popular tests of randomness is the Run test which we use to prove the non-randomness of the benchmark datasets used to conduct experiments. The statistical test is applied using the closing price of both datasets for the period from 01/01/2010 to 29/06/2018.

### 4.4.1 Run Test

Run test is a nonparametric statistical test which is used as an alternative to auto-correlation in the data. It is applied to test if a time series of a particular stock is behaving randomly, or if any predictable pattern is observed in the data. The test assumes that the mean and variance are constant and the probability is independent.

A run is defined as consecutive elements of either increasing or decreasing series. The number of consecutive similar symbols (decreasing or increasing) represents the length of the run. In random time series, the probability that any of the two symbols is larger or smaller than the other follows a binomial distribution which forms the basis of the run test. Applying the run test on time series requires transforming it into sequential elements of two symbols (positive or negative). Values above the median are coded as positive, and negative coding refers to values below the median. The null hypothesis assumes that sequences are produced in a random manner. The test is defined as<sup>3</sup> [106, 110, 111]:

$$Z = \frac{R - \bar{R}}{S_R} \quad (4.9)$$

where the number of observed runs (similar sequences) is denoted by  $R$ , whereas  $\bar{R}$  denotes the expected number of runs, and  $S_R$  is the standard deviation of the number of runs. The values of  $\bar{R}$  and  $S_R$  are defined as follows:

$$\bar{R} = \frac{2m_1m_2}{m_1 + m_2} + 1 \quad (4.10)$$

$$S_R^2 = \frac{2m_1m_2(2m_1m_2 - m_1 - m_2)}{(m_1 + m_2)^2(m_1 + m_2 - 1)} \quad (4.11)$$

where  $m_1$  and  $m_2$  refer to the number of negative and positive values in the series.

<sup>3</sup><https://www.itl.nist.gov/div898/handbook/eda/section3/eda35d.htm>

For large number of runs test, greater than 20, the test statistic follows the standard normal distribution. For a large-sample run test (where  $m_1 > 10$  and  $m_2 > 10$ ), the test statistic is compared to a standard normal table. That is, at the 5 % significance level, a test statistic with an absolute value greater than 1.96 indicates non-randomness.

#### 4.4.2 Results of the Run Test

Initially, we perform differencing on log transformation of the data to remove trend and seasonality from time series as discussed in Section 4.2. Next, we convert the time series by replacing data elements greater than the median/mean by 1 and data elements less than median/mean by -1. Under the null hypothesis of this test, at the 5% significance level, if the absolute value of the test is less than 1.96, the data is random and the null hypothesis cannot be rejected. The results of the run test using median and mean as reference values to divide the data are shown in Table 4.2.

For S&P and TASI time series, the null hypothesis is rejected and the data found to be non-random. On the other hand, the p-value of the DJIA dataset using the median as reference value is slightly greater than 0.05 which may not provide sufficient evidence to reject the null hypothesis or prove the data is non-random. However, the statistic results of using the mean as reference value provide sufficient evidence against the null hypothesis, thus we may consider it an indication that the DJIA dataset is not an output of a random process. Despite the results of the run randomness test which support predictability of benchmark dataset, white noise tests are also applied in the next section to investigate datasets randomness.



Table 4.2: The results of the run test using the median as reference value

Reference Value	Dataset	Raw data		Log Returns	
		Test	P-value	Test	P-value
Median	S&P	-45.427	< 2.20E-16	2.7689	0.005625
	DJIA	-44.389	< 2.20E-16	1.9036	0.05696
	TASI	-44.099	< 2.20E-16	-5.0423	4.60E-07
Mean	S&P	-45.859	< 2.20E-16	2.51	0.01207
	DJIA	-45.08	< 2.20E-16	1.9917	0.04641
	TASI	-44.099	< 2.20E-16	7.65E-06	-4.4748

## 4.5 White Noise Tests

Serial dependence in time series is commonly measured using the autocorrelation which is used by many statistical tests of white noise. Researchers investigated and studied the dependence structure of time-series data by calculating sample autocorrelations and Q test of Box and Pierce (1970) to check the joint significance of these statistics. The Q test proposed by Ljung and Box (1978) is also employed to analyze and test serial dependence in time series. One of the important steps in financial time-series analysis is to check serial correlations of squared series which could be achieved by the Q test of McLeod and Li (1983) [112]. The three white noise tests adopted by this work are discussed in the following.

### 4.5.1 Box-Pierce Test

Box-Pierce test is a statistical test of white noise based on the autocorrelation (ACF) [113]. The Box-Pierce  $Q_{BP}$  statistic is computed as the weighted sum of squares of a sequence of ACF  $\hat{\rho}_1(k)$ , for  $k$  number of time lags, defined as follows [113, 114].

Let  $x_t$  be a time series with the mean  $\mu$  and variance  $\sigma^2$ . So the autocorrelation

function (ACF)  $\hat{\rho}_1(k)$  of lag  $k$  can be defined as follows [115].

$$\hat{\rho}_1(k) = \frac{\sum_{t=1}^{n-k} (x_t - \bar{x})(x_{t+k} - \bar{x})}{\sum_{t=1}^n (x_t - \bar{x})^2} \quad (4.12)$$

$$Q_{BP}(k) = n \sum_{j=1}^k \hat{\rho}_1^2(j) \quad (4.13)$$

If the series is white noise, then the  $Q_{BP}(k)$  statistic is distributed approximately as a chi-square distribution with  $k$  degrees of freedom. It is sometimes known as a portmanteau test.

### 4.5.2 Ljung-Box Test

The Ljung-Box  $Q_{LB}$  test is closely connected to the Box-Pierce test. It is a white noise test for a time series which depends on the accumulated sample autocorrelations (ACF)  $\hat{\rho}_1(k)$  as defined in 4.12. For any range of time lags  $k$ , and the length of the time series  $n$ , the mathematical equation is defined [112, 113] as follows.

$$Q_{LB}(k) = n(n+2) \sum_{j=1}^k \frac{\hat{\rho}_1^2(j)}{n-j} \quad (4.14)$$

Ideally, for any time series generated by a random process, the time lags are assumed to be identically independently distributed from each other, therefore, the ACF is zero [116]. The rationale behind this statistic is that, if all the ACF values of the time lags are very close to zero, then we may conclude that the underlying time series is

white noise. The higher the value of the  $Q_{LB}(k)$ , the more unlikely the time series is a white noise. The null distribution of the  $Q_{LB}(k)$  is approximately a Chi-square  $\chi^2$  distribution with  $k$  degrees of freedom.

### 4.5.3 McLeod-Li Test

It uses the ACF of the squared series  $x_t^2$ , and tests whether the first  $k$  ACF  $\hat{\rho}_2(k)$  for the squared time series are collectively small in magnitude. The  $Q_{ML}$  statistic is as follows [112, 113].

$$Q_{ML}(k) = n(n+2) \sum_{j=1}^k \frac{\hat{\rho}_2^2(j)}{n-j} \quad (4.15)$$

$$\hat{\rho}_2(k) = \frac{\sum_{t=1}^{n-k} (x_t^2 - \hat{\sigma}_n^2) (x_{t+k}^2 - \hat{\sigma}_n^2)}{\sum_{t=1}^n (x_t^2 - \hat{\sigma}_n^2)^2} \quad (4.16)$$

$$\hat{\sigma}_n^2 = \frac{1}{n} \sum_{j=1}^n x_j^2 \quad (4.17)$$

If the series is white noise, then the  $Q_{ML}(k)$  statistic is distributed approximately as  $\chi^2$  distribution with  $k$  degrees of freedom.

#### 4.5.4 Results of White Noise Tests

We apply the three statistical white noise tests on the S&P, DJIA, and TASI datasets. The three tests conducted using the raw closing price and the log returns of the closing price. The null hypothesis of the tests implies that the data are white noise. The results are shown in Table 4.3. The p-value holds a significant evidence against the null hypothesis, meaning that the data is not white noise and we cannot accept the null hypothesis. This result encourages us to forge ahead in developing the forecasting models and start up conducting forecasting experiments.

Table 4.3: Results of white noise statistics for S&P dataset

Dataset	Test	DF	Closing price raw data		Log returns of closing price	
			Value	P-Value	Value	P-Value
S&P	Box-Pierce	6	12675.77	< 0.0001	32.34	< 0.0001
		12	25087.33	< 0.0001	613.24	< 0.0001
	Ljung-Box	6	12708.36	< 0.0001	32.44	< 0.0001
		12	25186.90	< 0.0001	617.15	< 0.0001
	McLeod-Li	6	12689.21	< 0.0001	670.20	< 0.0001
		12	25107.37	< 0.0001	1489.79	< 0.0001
DJIA	Box-Pierce	6	12660.75	< 0.0001	33.26	< 0.0001
		12	12693.30	< 0.0001	33.36	< 0.0001
	Ljung-Box	6	12679.07	< 0.0001	618.13	< 0.0001
		12	25026.83	< 0.0001	601.66	< 0.0001
	McLeod-Li	6	25126.10	< 0.0001	605.48	< 0.0001
		12	25063.54	< 0.0001	1380.71	< 0.0001
TASI	Box-Pierce	6	12443.49	< 0.0001	37.15	< 0.0001
		12	12475.74	< 0.0001	37.21	< 0.0001
	Ljung-Box	6	12493.34	< 0.0001	552.69	< 0.0001
		12	24418.58	< 0.0001	42.79	< 0.0001
	McLeod-Li	6	24516.02	< 0.0001	42.88	< 0.0001
		12	24575.30	< 0.0001	630.80	< 0.0001

## CHAPTER 5

# EXPERIMENTS AND RESULTS

The experiments conducted by this study are initiated by developing and comparing different deep recurrent neural networks, such as, GRU and LSTM, designed using stacked and bidirectional architectures. Description of the experimental settings and performance evaluation measures which are employed to compare and estimate the performance of the forecasting models are discussed in Section 5.1. Section 5.2 discusses experiments conducted to develop many architectures based on different parameters and multiple time steps for forecasting several future horizon scales.

The proposed solution combines MRA with deep recurrent neural network techniques to build a multiresolution deep learning neural network (MRA-DNN) solution for financial time series forecasting. Section 5.3 discusses experiments conducted using BGRU and empirical wavelet transform to perform forecast for short and long-term. Several models are employed to perform forecast using variable rolling windows to reshape the input data. Section 5.5 studies the impact of combining ten of the most common TIs with EWT-BGRU approach. In Section 5.6, we summarize the results and compare our proposed model with some related works.

# 5.1 Experimental Settings

The conducted experiments and evaluation procedure are based on historical data of S&P, DJIA, and TASI indices downloaded from Yahoo finance for the period from 01/01/2010 to 29/06/2018. The closing price at the end of every trading day is used for the purpose of the forecasting. The training process uses daily closing price to forecast the closing price in the future. We use the adopted time lag methodology whereby the data is rearranged using a sliding window of size  $k$  denoted as  $(x_{t-k+1}, x_{t-k+2}, \dots, x_{t-1}, x_t)$  to forecast a time lag  $x_{t+h}$  at time step  $t$ , where  $h$  denotes the forecasting horizon. Figure 5.1 illustrates the input values (predictors) and output (target) for short-term and long-term forecast. The experiments conducted using different sliding window sizes to forecast closing prices at different horizon scales in the future.

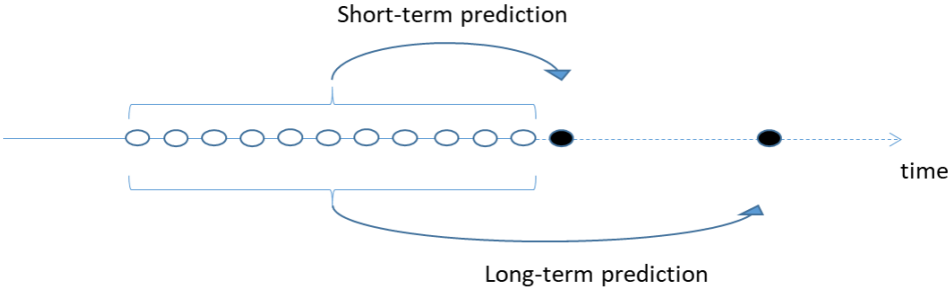


Figure 5.1: Illustration of short-term and long-term forecast

The data is normalized between the range (0, 1) using min-max normalization to ease computations and decrease multivariate values divergence. The normalization is accomplished by performing a linear transformation on the input data values. The following equation represents the min-max normalization process.

$$norm(x) = \frac{x - min_i}{max_i - min_i} \quad (5.1)$$

where  $min_i$  and  $max_i$  denote the original interval,  $x$  represents every value,  $norm(x)$  denotes the normalized value. The data is split into three parts: training, validation, and testing. The first 80% duration of the whole data is allocated for training and the rest 20% duration is allocated for testing. The last 20% of the training data is allocated for validation. Illustration of data split into training, validation, and testing is shown in Figure 5.2.

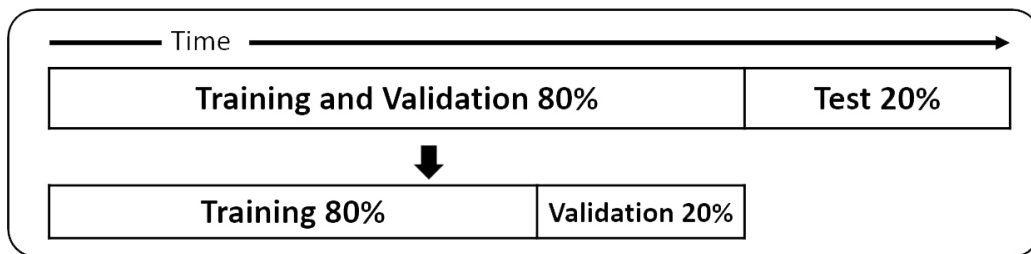


Figure 5.2: Illustration of data split into training, validation and testing

Figure 3.11 illustrates the methodology followed to construct bidirectional architectures for LSTM and GRU. Similarly, Figure 3.10 illustrates the methodology followed to construct stacked architectures for LSTM and GRU. Both LSTM and GRU lay-

ers use tanh transfer function for the output layer and use the hard sigmoid transfer function for the recurrent activation. The final output is produced by a linear activation function of a dense layer. The developed models are tuned by conducting a set of experiments on different structures. Different number of memory cells are used to train many models to deduce best training structure. The training process is performed using training data and best network weights are determined based on the results generated from the validation data. The number of epochs used to train all developed networks is 800 epochs and batch size of 32. After each epoch, based on the evaluation on validation data, the trained network architectures are evaluated and compared depending on the results generated using test data.

To measure how well the model forecasts representing the actual data, we use four performance measures in the experiments: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and coefficient of determination which is denoted as  $R^2$ . For better forecast, MAE, RMSE, and MAPE should be close to zero and  $R^2$  should be close to one. These measures are defined mathematically as follows:

$$MAE = \frac{1}{n} \sum_{t=0}^{n-1} |x_t - \hat{x}_t| \quad (5.2)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{t=0}^{n-1} (x_t - \hat{x}_t)^2} \quad (5.3)$$



$$MAPE = \frac{\sum_{t=0}^{n-1} \left| \frac{x_t - \hat{x}_t}{x_t} \right|}{n} \times 100\% \quad (5.4)$$

$$R^2 = 1 - \frac{\sum_{t=0}^{n-1} (x_t - \hat{x}_t)^2}{\sum_{t=0}^{n-1} (x_t - \bar{x})^2} \quad (5.5)$$

where  $x_t$  and  $\hat{x}_t$  represent the actual and forecast values at step  $t$  for  $0 \leq t < n$ , respectively, and  $\bar{x} = \sum_{t=0}^{n-1} x_t/n$ .

We implement the proposed forecasting methodology in Python using Keras open-source package for deep learning with TensorFlow backend <sup>1</sup>. The stationary wavelet analysis is performed using PyWavelets open-source library <sup>2</sup>. The empirical adaptive wavelet analysis is accomplished using empirical wavelet transforms matlab toolbox [91]. We use Ta-Lib library <sup>3</sup> to define TIs over benchmark datasets. It is used in many studies to compute TIs for stock market forecasting [117–119].

<sup>1</sup><https://github.com/keras-team/keras>

<sup>2</sup><https://github.com/PyWavelets/pywt>

<sup>3</sup><https://www.ta-lib.org/>

## 5.2 Comparison Between Bidirectional and Stacked (GRU, LSTM)

Initially, we start the experiments by investigating different forecasting parameters, we investigate different options such as the number of lags (time steps), number of neurons (memory blocks), different types of techniques, and multiple time scale horizons. The experiments help us decide the best parameters and techniques that can be used to perform forecast. We aim at forecasting multiple time scale horizons in the future, which are 1, 5, 7, 10, 15, 22, and 30 days.

The data is framed using multiple numbers of time lags with overlapping. The numbers of time lags used to reshape the data are 1, 3, 5, 7, and 10 days from the past. Four architectures are used to perform the experiments. The four techniques are BLSTM, Stacked Long-Short Term Memory (SLSTM), BGRU, and Stacked Gated Recurrent Unit (SGRU). We use the same experimental settings discussed in Section 5.1. For each technique we design multiple networks using different numbers of memory blocks using even numbers between 1 and 17.

The number of developed networks for both datasets is large due to the various set of options used to perform the experiments, since we use 4 techniques with 8 different numbers of neurons using 5 different number of time steps for 6 different time scales, the final number of trained networks is 960 networks for each dataset. Therefore, showing detailed results of experiments is impossible, instead, we show the averages

of the RMSE result of the test data for each time scale based on the number of time steps and based on the number of neurons. Figures 5.3 and 5.4 show a clustered column charts for both datasets used in the experiments. They illustrate differences between the performance of the four techniques based on averages of RMSE of test data. The performance of different networks developed using different numbers of neurons are grouped by time steps for all forecast horizons. Likewise, the averages of the performance of the trained networks are grouped by number of memory blocks (neurons) for each time scale. They are illustrated in Figures 5.5 and 5.6. From the results of the experiment we learn the following :

- We can conclude that the bidirectional architecture outperforms stacked architecture.
- It is clear from column charts that the further the time scale forecast horizon in the future, the higher the error scale and the less the performance of the forecast.
- The BGRU networks outperform almost all other types of networks which may opt this architecture as the best candidate for developing the forecasting approach.
- The pattern generated from experiments of forecasting one day ahead in the future does not show big differences between the four techniques (BLSTM, BGRU, SLSTM, SGRU). However, it may show that bidirectional architecture produces slightly higher performance. Increasing the number of epochs may clarify the difference.

- Using one time lag (time step) to forecast the future produces an adequate forecasting performance compared to the performance produced by models trained using more than one time lag, specially for long-term forecast. Therefore, including one time lag training architecture is preferred for planned experiments in the future. For the rest of the time lag options of BGRU and BLSTM, we notice that mostly ten time steps is better and we need to perform more experiments using much longer time lags of the past. For the SLSTM and SGRU, we notice that using much longer time lags may not make the performance any better, since the larger the number of time lags, the higher the mean error.
- The result of networks developed to forecast 30 days ahead in the future does not benefit from increasing the number of time lags, but rather it affects the performance negatively. We may learn that performing forecast of very long time scales in the future may not be efficient, since the further the time duration between the training time lags window (predictors) and the target value, the more the forecasts become noisy and prone to errors.
- We can notice that the error for all networks gets smaller as we increase the number of neurons. Thus, the number of memory blocks (neurons) used in the experiments may not be optimal, therefore we need to perform more experiments using larger number of neurons.

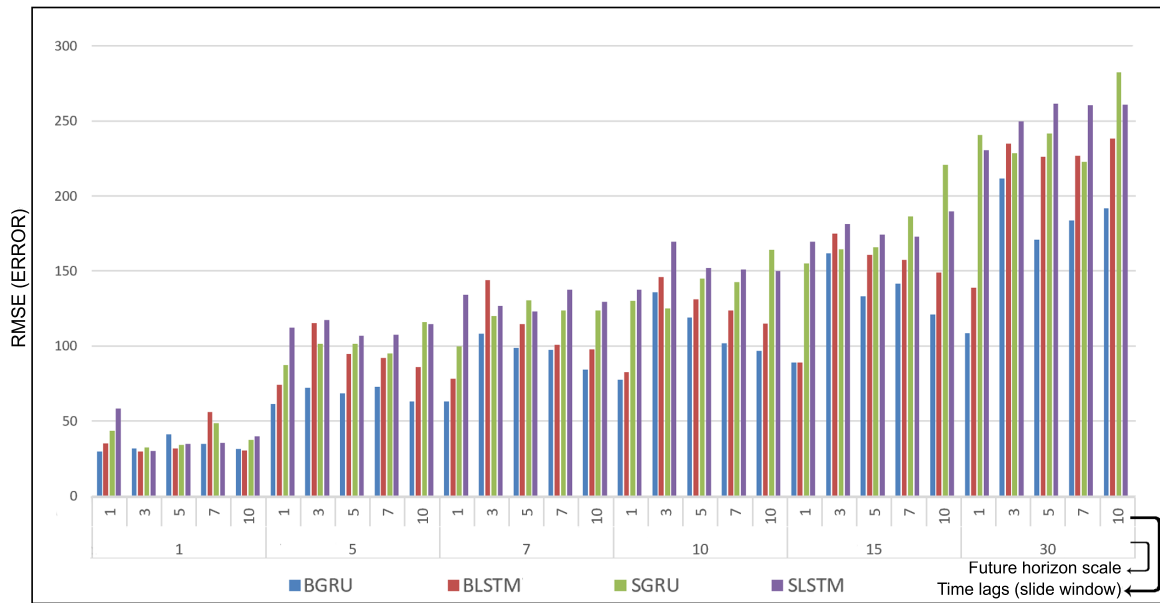


Figure 5.3: Comparison between RMSE averages of S&P test data based on different time lags for multiple time scales

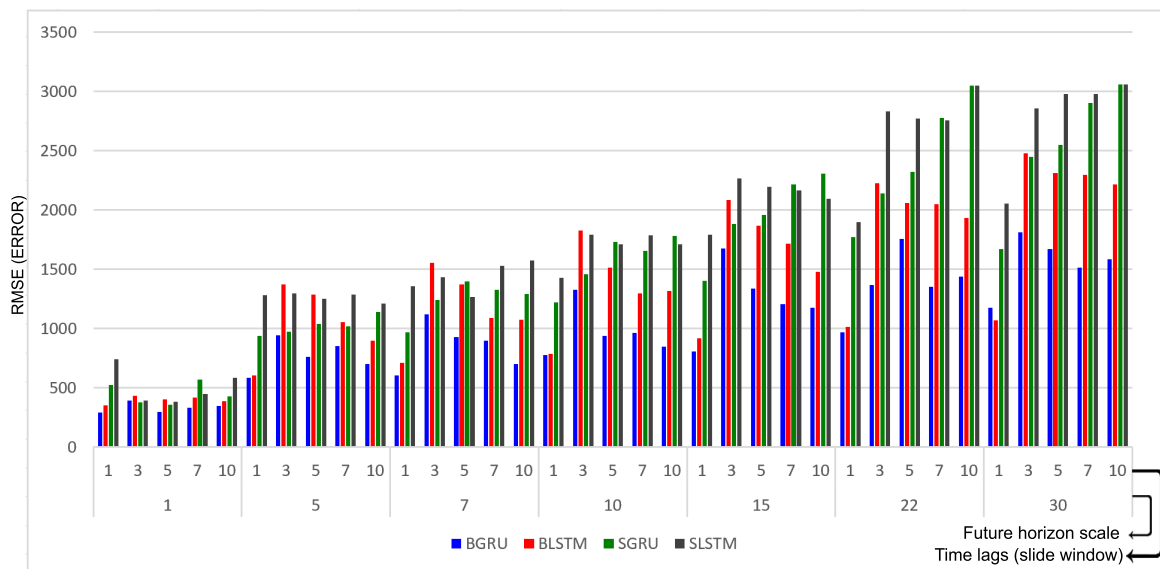


Figure 5.4: Comparison between RMSE averages of DJIA test data based on different time lags for multiple time scales

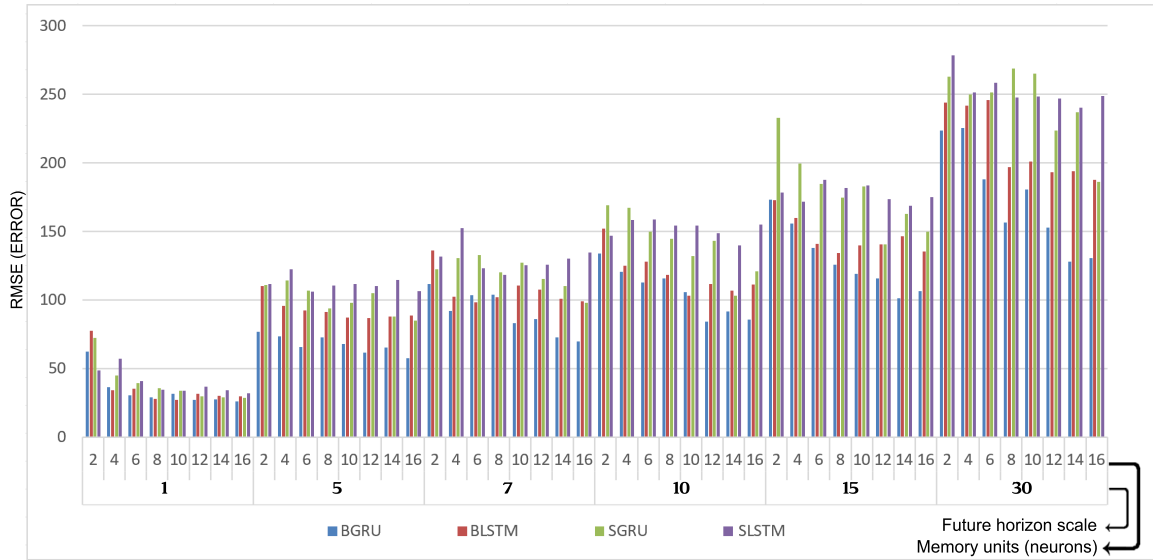


Figure 5.5: Comparison between RMSE averages of S&P test data based on different number of neurons for multiple time scales

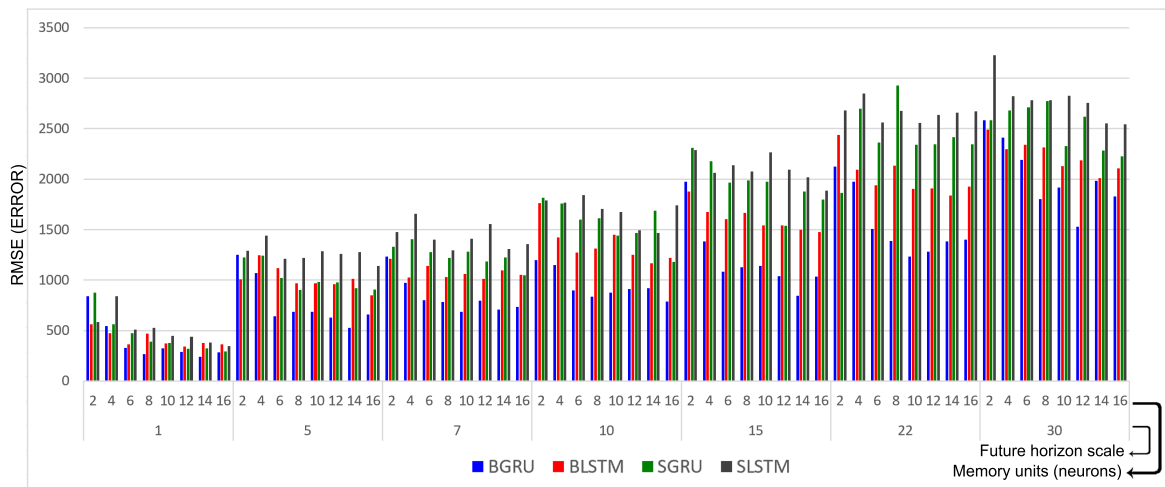


Figure 5.6: Comparison between RMSE averages of DJIA test data based on different number of neurons for multiple time scales

## Extended Evaluation of BGRU and BLSTM

The results of the experiments conducted in previous section show that the more number of neurons we use to build the network, the better the forecasting performance, thus conducting experiments using higher number of neurons and time lags is preferred. The results also show that bidirectional architectures of GRU and LSTM produce performance higher than that of the stacked architectures of GRU and LSTM. Therefore, we perform the experiments using only bidirectional architecture which produces higher forecast performance as shown in the previous section. We investigate additional parameters to compare and evaluate the forecasting performance of BGRU and BLSTM networks. The additional parameters include longer time lags of input data and larger number of neurons to design the networks.

As in the experiments discussed in Section 5.2, the forecasting models are developed to forecast multiple time scales including 1, 5, 7, 10, 15, 22, and 30 days in the future. The data is reshaped using multiple number of time lags with overlapping. The numbers of time lags used to frame the data are 1, 3, 5, 7, 10, 13, 15, 20, and 24 time steps. Two techniques are used to perform the experiments. We use BLSTM and BGRU to build multiple networks based on different number of neurons. In addition to the different number of neurons used in models developed in Section 5.2, we develop additional models using 18, 24, 28, and 32 neurons.

The number of developed networks is 1512 which represents two techniques (BGRU, BLSTM) using 9 time lag window sizes with 12 different neurons numbers for 7 future horizon scales using BGRU and BLSTM architectures for each dataset.

Averages of RMSE measure on test data are illustrated by a clustered column charts in Figures 5.7 and 5.8. The averages of the performance of the trained networks grouped by number of neurons for each time scale are illustrated in Figures 5.9 and 5.10. Based on the experiments discussed in this section in addition to Section 5.2, the best trained models classified by forecasting time scale are shown in Figures 5.11 and 5.12.

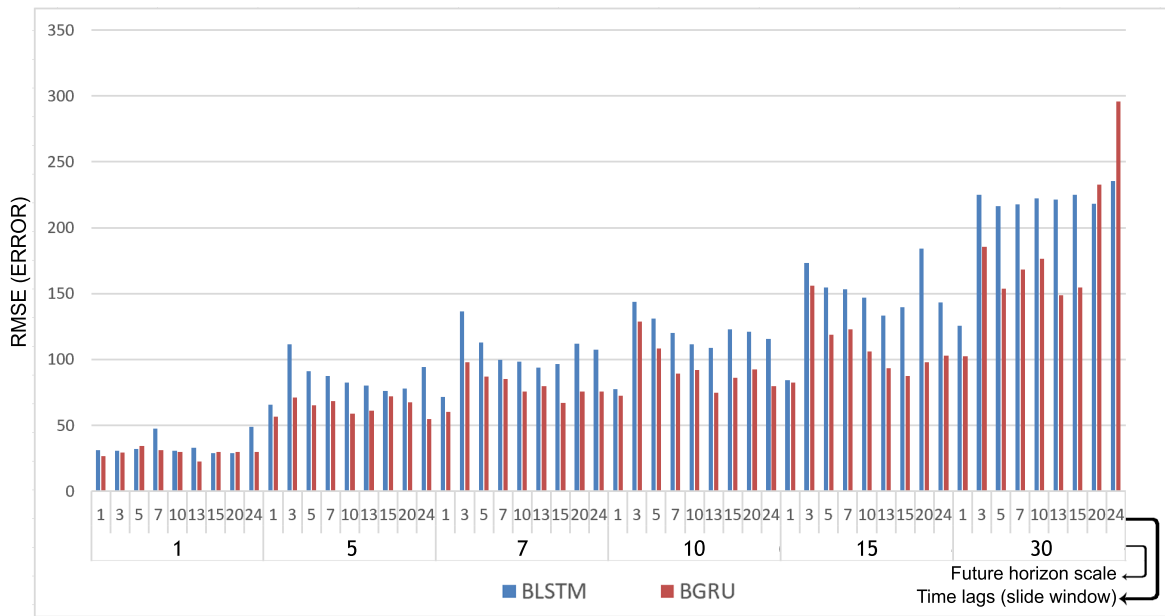


Figure 5.7: Comparison between RMSE averages of S&P test data based on additional different time lags for multiple time scales

From the results of experiments summarized in this section and Section 5.2, we learn the following :

- BGRU models show better performance than all other types of models. Though difference between performance of BGRU and BLSTM for one day ahead forecast is not significant, BGRU tend to produce slightly better forecasts.
- Comparing the performance of BLSTM models based on different time lags,



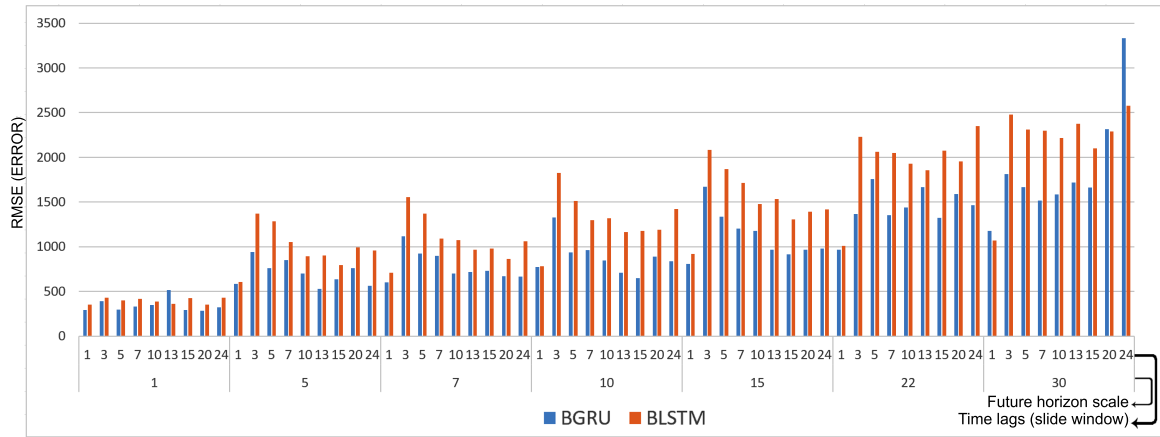


Figure 5.8: Comparison between RMSE averages of DJIA test data based on additional different time lags for multiple time scales

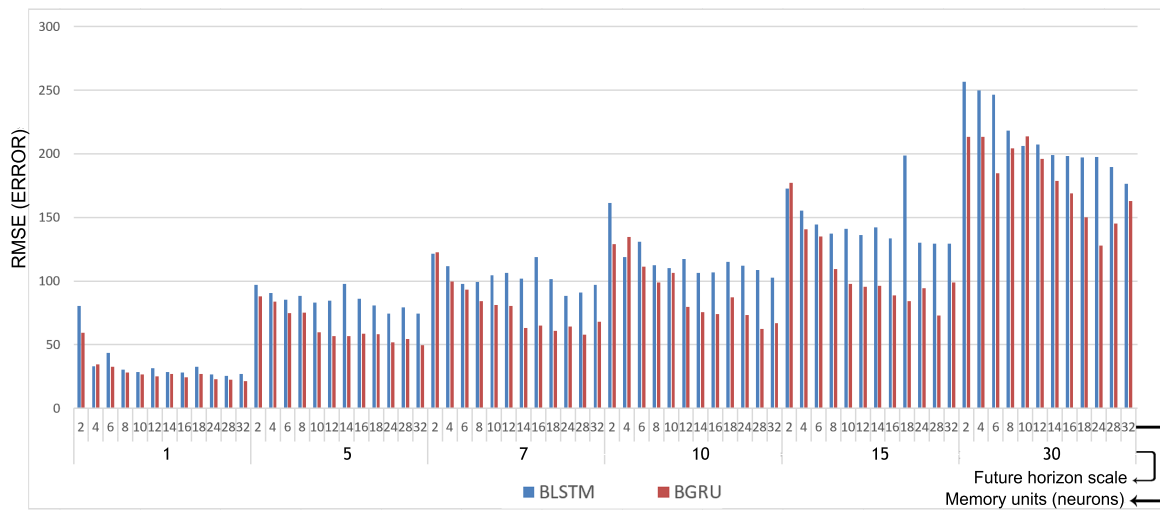


Figure 5.9: Comparison between RMSE averages of S&P test data based on additional different number of neurons for multiple time scales

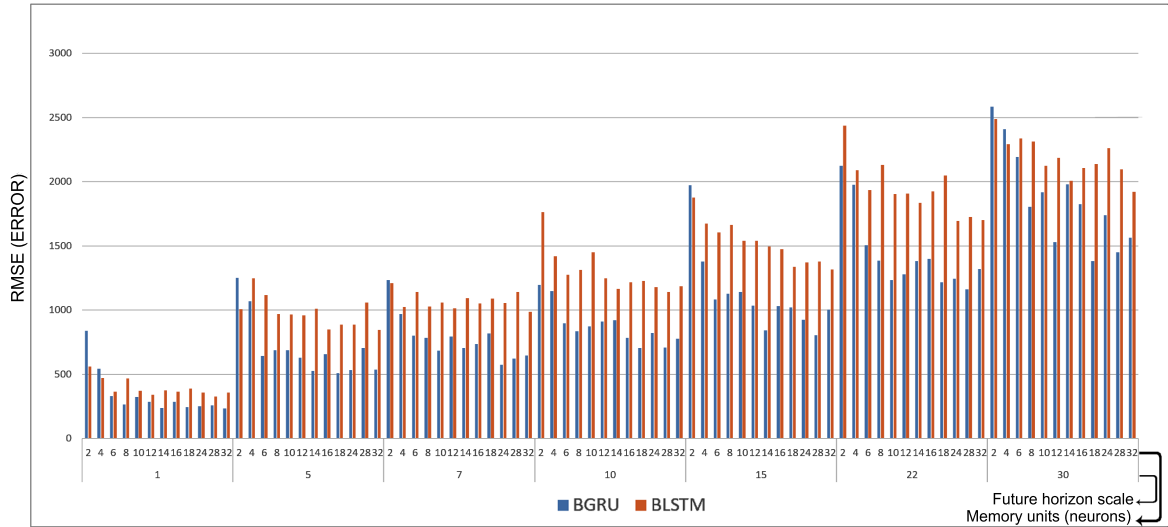


Figure 5.10: Comparison between RMSE averages of DJIA test data based on additional different number of neurons for multiple time scales

we see that performance produced by models that use one time lag as input is better.

- Comparing the performance of BGRU models based on different time lags, we see from Figures 5.7 and 5.8 that not only one time lag produces high performance, but also other time lags such as 13, 15, and 24, produce high performance. BGRU is the only technique that produced high performance using different number of time lags. Therefore, BGRU is a highly preferred solution to perform forecasting using multiple time lags, especially for mid and long-term time scales.
- We can notice from Figures 5.9 and 5.10 that using larger number of neurons (memory blocks) tends to produce higher performance, especially for 30 days ahead forecasting. However, increasing the number of neurons in the developed models more than 32 neurons may not produce any better performance, specially for short-term forecasting such as 1, 5, and 7 days. For the long-term forecasting,

increasing number of neurons might improve the performance.

- It is clear that the performance of the BGRU model is so close to the random walk model performance as shown in Figures 5.11 and 5.12.

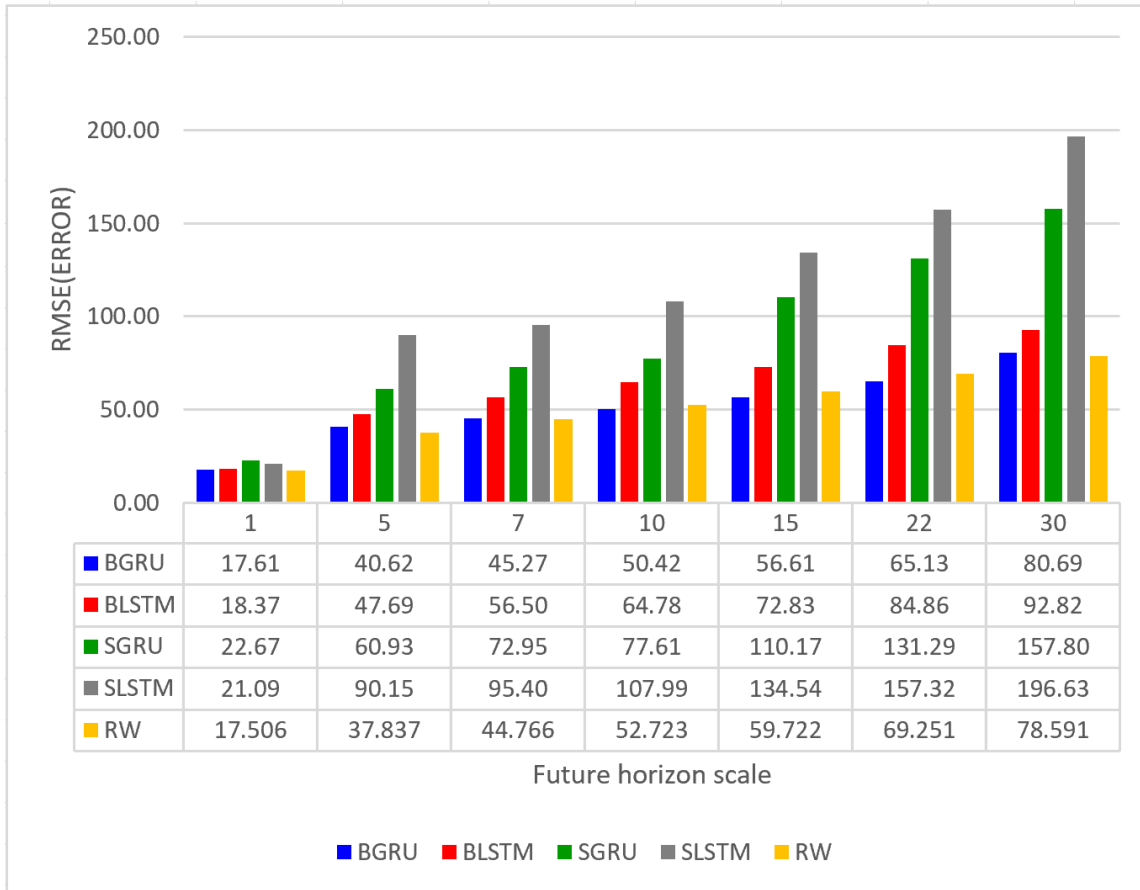


Figure 5.11: Comparison between best RMSE results of S&P test data for multiple horizon scales

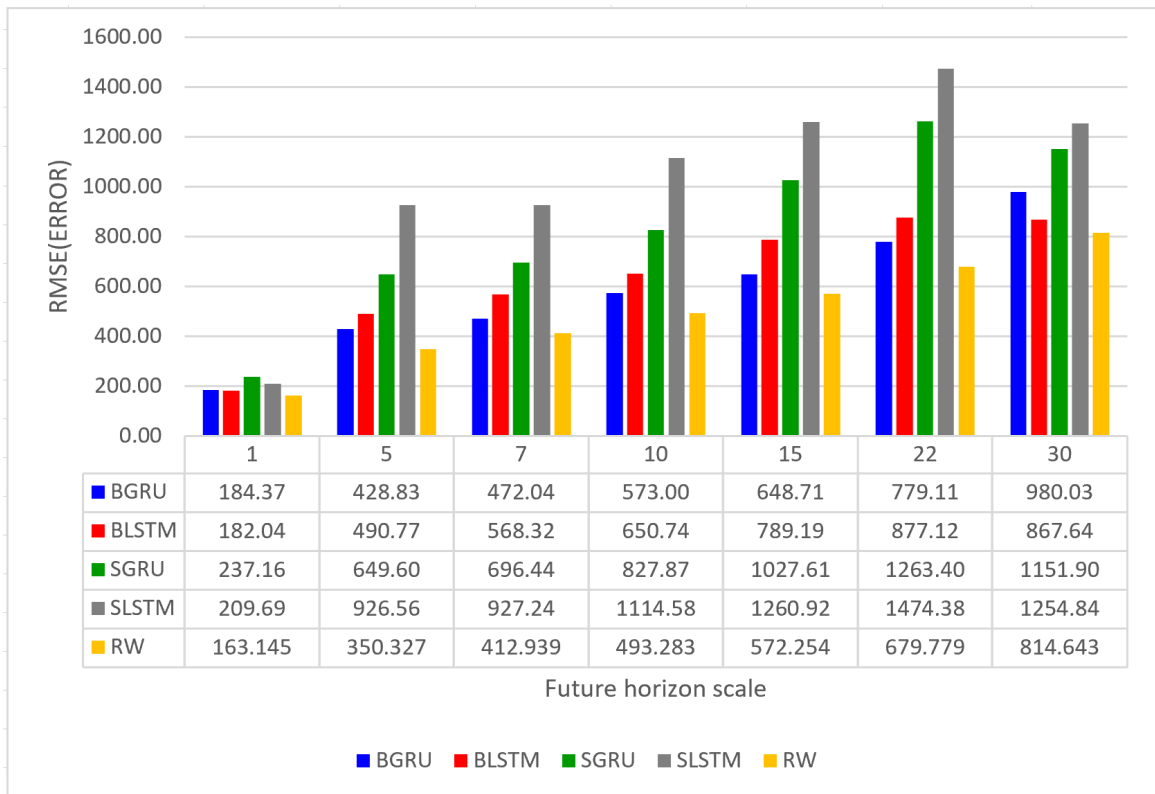


Figure 5.12: Comparison between best RMSE results of DJIA test data for multiple horizon scales

### 5.3 Empirical Wavelet Transform and BGRU

Based on the experiments conducted in Section 5.2, we noticed that BGRU networks attain the best performance compared to other deep learning architectures. Therefore, we develop a forecasting model by a combination of BGRU learning methodology and EWT analysis method. The proposed architecture uses BGRU in learning the decomposed data using empirical wavelet multiresolution analysis. The forecasting approach methodology is illustrated in Figure 5.13.

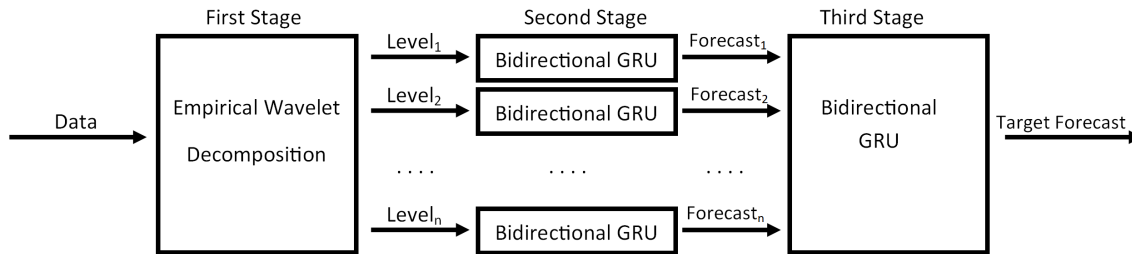


Figure 5.13: The architecture employed to develop the forecasting approach using BGRU learning technique and EWT method (EWT-BGRU)

We evaluate the proposed model which is composed of empirical wavelet analysis and BGRU using S&P and DJIA datasets to forecast closing prices for one day and one month ahead (22 working days) in the future. The experiments conducted in this section used two forms of data. The first type of experiments used raw closing price data which is normalized between 0 and 1. The second type of the experiments used log returns of closing price as input data. The results of both types of experiments are

compared and analyzed. The log return of closing prices are calculated using formula discussed in Section 4.2. Same experimental settings discussed in Section 5.1 are used in the experiments summarized in this section. The parameters configuration considers changing and investigating several slide window sizes (time steps) and different neuron numbers for each model. The different parameters are selected based on models which produce high performance measures in experiments summarized in Section 5.2. Both log returns and normalized raw closing prices are reshaped using slide window using the adopted time lag methodology described in Section 3.4. The different sliding window sizes selected are 1, 5, 10, 15, and 24 days time steps with overlapping. The developed models are tuned using 16, 32, and 48 neurons. Each resolution level is fed into the intermediate deep network designed using sequence of two BGRU layers and a dense layer. The final stage deep network is also designed using another deep network constructed of sequence of a two BGRU layers and a dense layer. The final forecast is generated from the final network which inverts the decomposition process using intermediate forecasts of each resolution level produced by the intermediate deep network. The conducted experiments can be divided into two parts based on time horizon forecast in future; the first part of the experiments aims to accomplish short-term (one day) forecasting and the second part performs the training process to carry out long-term (22 working days) forecasting.

The short-term forecasting results generated from the proposed EWT-BGRU architecture are shown in Tables 5.1 and 5.2 for both S&P and DJIA dataset, respectively. The results are classified based on window size used to reshape the data and number of neurons used to build both intermediate and final stages models. They

show comparison between models which use raw closing price data and models which use log returns of closing prices. To clarify the difference between the two types of models, Figures 5.14 and 5.15 illustrate the results based on the RMSE of test data using clustered column charts. Similarly, Tables 5.3 and 5.3 show the results of long-term forecasts and Figures 5.16 and 5.17 illustrate the results using clustered column charts.

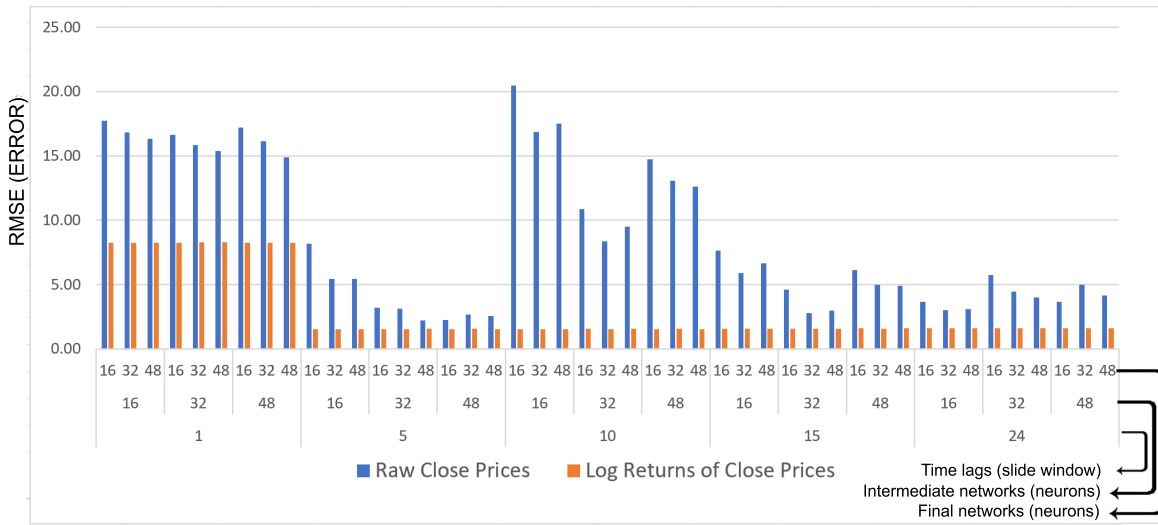


Figure 5.14: Comparison between test data RMSE of trained models to forecast S&P short-term closing prices using raw data and log returns based on different time lags and number of neurons

Based on the results, we may conclude the following:

- Using a combination of EWT analysis method and BGRU learning architecture (EWT-BGRU) improved the forecast performance, compared to results of experiments conducted so far.
- The best trained models for both experiments which use raw closing prices and those which use log returns are shown in Table 5.5. The difference between



Figure 5.15: Comparison between test data RMSE of trained models to forecast DJIA short-term closing prices using raw data and log returns based on different time lags and number of neurons

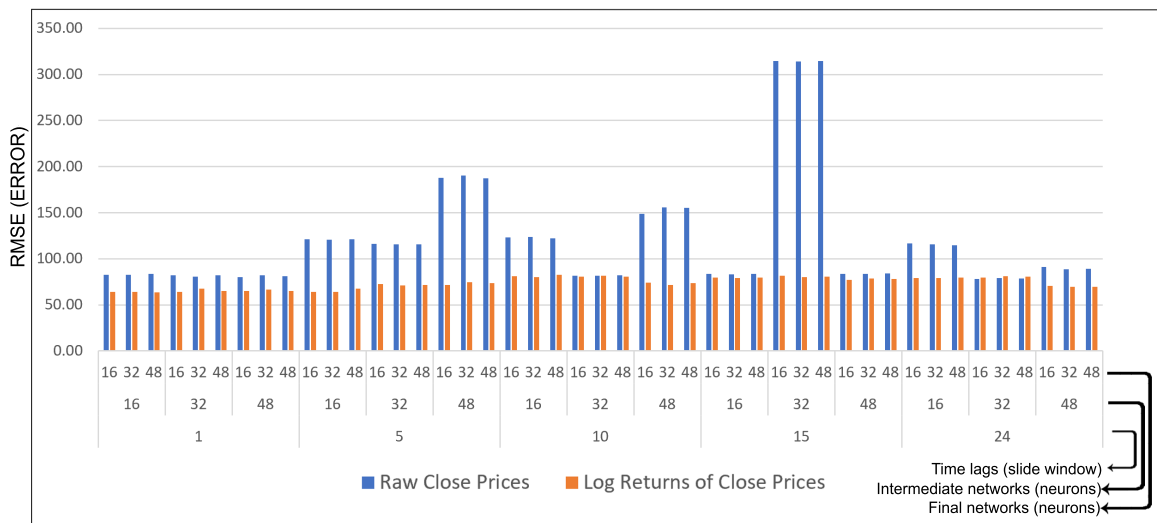


Figure 5.16: Comparison between test data RMSE of trained models to forecast S&P long-term closing prices using raw data and log returns based on different time lags and number of neurons



Table 5.1: Short-term forecast results of S&P test data using raw closing prices and log returns of closing prices

Window Size	Network		Raw closing prices				Log Returns of closing prices			
	1st stage	2nd stage	MAE	RMSE	MAPE	$R^2$	MAE	RMSE	MAPE	$R^2$
1	16	16	13.23	17.75	0.5600	0.99549	8.2566	13.0610	0.3242	0.99517
		32	12.51	16.83	0.5326	0.99595	8.2505	13.0633	0.3240	0.99516
		48	12.12	16.33	0.5178	0.99618	8.2679	13.0628	0.3247	0.99516
	32	16	12.39	16.65	0.5281	0.99603	8.2529	13.0622	0.3241	0.99516
		32	11.75	15.84	0.5038	0.99641	8.2719	13.0637	0.3248	0.99516
		48	11.36	15.39	0.4891	0.99661	8.2705	13.0630	0.3248	0.99516
	48	16	12.87	17.19	0.5471	0.99577	8.2540	13.0630	0.3241	0.99516
		32	12.03	16.13	0.5148	0.99628	8.2574	13.0618	0.3243	0.99517
		48	10.95	14.90	0.4735	0.99682	8.2558	13.0634	0.3242	0.99516
5	16	16	5.91	8.19	0.2369	0.99904	1.5423	2.0432	0.0608	0.99988
		32	4.10	5.42	0.1680	0.99958	1.5231	2.0257	0.0600	0.99988
		48	4.17	5.43	0.1709	0.99958	1.5303	2.0417	0.0603	0.99988
	32	16	2.26	3.20	0.0936	0.99985	1.5368	2.0448	0.0606	0.99988
		32	2.22	3.12	0.0922	0.99986	1.5477	2.0556	0.0610	0.99988
		48	1.69	2.22	0.0724	0.99993	1.5530	2.0554	0.0612	0.99988
	48	16	1.74	2.26	0.0747	0.99993	1.5343	2.0274	0.0605	0.99988
		32	1.89	2.67	0.0799	0.99990	1.5591	2.0578	0.0615	0.99988
		48	1.82	2.55	0.0770	0.99991	1.5373	2.0284	0.0606	0.99988
10	16	16	13.06	20.46	0.5068	0.99402	1.5392	2.0357	0.0608	0.99988
		32	10.72	16.88	0.4169	0.99593	1.5308	2.0279	0.0604	0.99988
		48	11.19	17.50	0.4352	0.99563	1.5312	2.0234	0.0604	0.99988
	32	16	7.03	10.88	0.2749	0.99831	1.5652	2.0683	0.0618	0.99988
		32	5.44	8.37	0.2143	0.99900	1.5425	2.0373	0.0609	0.99988
		48	6.22	9.49	0.2443	0.99871	1.5605	2.0679	0.0616	0.99988
	48	16	9.29	14.74	0.3608	0.99689	1.5420	2.0391	0.0609	0.99988
		32	8.35	13.07	0.3254	0.99756	1.5546	2.0512	0.0614	0.99988
		48	8.05	12.61	0.3138	0.99773	1.5491	2.0449	0.0612	0.99988
15	16	16	5.12	7.65	0.2022	0.99917	1.5689	2.0461	0.0619	0.99988
		32	3.99	5.88	0.1590	0.99951	1.5671	2.0474	0.0618	0.99988
		48	4.58	6.66	0.1820	0.99937	1.5625	2.0362	0.0617	0.99988
	32	16	3.39	4.62	0.1375	0.99969	1.5829	2.0654	0.0624	0.99988
		32	2.11	2.79	0.0882	0.99989	1.5884	2.0702	0.0626	0.99988
		48	2.24	2.98	0.0935	0.99987	1.5847	2.0722	0.0625	0.99988
	48	16	4.21	6.13	0.1679	0.99946	1.5976	2.0874	0.0630	0.99987
		32	3.51	4.98	0.1411	0.99965	1.5804	2.0669	0.0623	0.99988
		48	3.50	4.93	0.1408	0.99965	1.5970	2.0870	0.0630	0.99987
24	16	16	2.90	3.64	0.1212	0.99981	1.5944	2.0849	0.0627	0.99987
		32	2.24	3.00	0.0951	0.99987	1.6133	2.1133	0.0636	0.99987
		48	2.25	3.10	0.0955	0.99986	1.6059	2.0990	0.0633	0.99987
	32	16	3.99	5.75	0.1588	0.99953	1.6107	2.1083	0.0634	0.99987
		32	3.17	4.44	0.1275	0.99972	1.6054	2.1065	0.0632	0.99987
		48	2.91	3.99	0.1177	0.99977	1.6075	2.1031	0.0633	0.99987
	48	16	5.16	3.67	0.1472	0.99962	1.6244	2.1147	0.0640	0.99987
		32	7.11	4.99	0.1980	0.99928	1.6217	2.1112	0.0639	0.99987
		48	5.74	4.14	0.1655	0.99953	1.6167	2.1083	0.0637	0.99987

Table 5.2: Short-term forecast results of DJIA test data using raw closing prices and log returns of closing prices

Window Size	Network		Raw closing prices				Log Returns of closing prices			
	1st stage	2nd stage	MAE	RMSE	MAPE	$R^2$	MAE	RMSE	MAPE	$R^2$
1	16	16	135.86	189.69	0.6343	0.99575	76.7673	128.5205	0.3320	0.99639
		32	125.24	173.76	0.5904	0.99643	76.7604	128.5569	0.3320	0.99639
		48	121.56	168.72	0.5748	0.99664	76.8287	128.5417	0.3323	0.99639
	32	16	129.93	180.20	0.6101	0.99617	76.8030	128.5316	0.3322	0.99639
		32	116.40	160.85	0.5536	0.99694	76.7465	128.5414	0.3319	0.99639
		48	109.11	151.18	0.5229	0.99730	76.7668	128.5592	0.3320	0.99639
	48	16	123.15	170.90	0.5815	0.99655	76.8350	128.5316	0.3323	0.99639
		32	102.22	141.76	0.4941	0.99763	76.7954	128.5284	0.3322	0.99639
		48	99.09	138.09	0.4807	0.99775	76.8928	128.5566	0.3326	0.99639
5	16	16	111.08	165.02	0.4819	0.99679	9.8369	13.2021	0.0432	0.99996
		32	95.44	139.65	0.4163	0.99770	9.8027	13.1246	0.0431	0.99996
		48	93.42	136.37	0.4079	0.99781	9.7769	13.1481	0.0429	0.99996
	32	16	125.63	214.57	0.5312	0.99457	9.7053	13.0540	0.0426	0.99996
		32	109.33	187.43	0.4627	0.99586	9.7552	13.1238	0.0429	0.99996
		48	108.21	185.14	0.4581	0.99596	9.7411	13.0453	0.0428	0.99996
	48	16	22.00	30.52	0.1014	0.99989	9.8308	13.2825	0.0431	0.99996
		32	20.19	27.54	0.0940	0.99991	9.8679	13.2863	0.0433	0.99996
		48	13.51	17.71	0.0662	0.99996	9.7590	13.1663	0.0428	0.99996
10	16	16	19.80	28.26	0.0919	0.99991	8.8311	11.4748	0.0391	0.99997
		32	23.50	37.19	0.1065	0.99984	8.8117	11.4567	0.0390	0.99997
		48	28.15	45.15	0.1255	0.99976	8.7255	11.3871	0.0387	0.99997
	32	16	71.71	115.97	0.3078	0.99842	8.7053	11.2890	0.0386	0.99997
		32	59.02	95.65	0.2541	0.99892	8.7642	11.3008	0.0389	0.99997
		48	55.66	89.26	0.2403	0.99906	8.7800	11.3343	0.0389	0.99997
	48	16	62.54	100.69	0.2691	0.99881	8.7131	11.3165	0.0386	0.99997
		32	39.95	63.12	0.1744	0.99953	8.8906	11.4581	0.0393	0.99997
		48	36.37	57.29	0.1594	0.99961	8.8913	11.5083	0.0393	0.99997
15	16	16	138.86	261.82	0.5801	0.99192	8.7084	11.3327	0.0386	0.99997
		32	126.51	242.71	0.5280	0.99305	8.6418	11.2239	0.0383	0.99997
		48	121.66	234.29	0.5079	0.99353	8.6628	11.2848	0.0384	0.99997
	32	16	103.56	179.52	0.4380	0.99620	8.8980	11.6069	0.0394	0.99997
		32	96.79	166.89	0.4101	0.99672	8.8919	11.6115	0.0394	0.99997
		48	89.75	155.16	0.3806	0.99716	8.9197	11.6494	0.0395	0.99997
	48	16	102.60	172.40	0.4359	0.99649	9.0026	11.6847	0.0399	0.99997
		32	99.23	166.16	0.4220	0.99674	8.9523	11.6501	0.0397	0.99997
		48	97.16	162.10	0.4135	0.99690	8.9973	11.7514	0.0398	0.99997
24	16	16	130.35	223.14	0.5517	0.99414	9.3370	12.3769	0.0413	0.99997
		32	123.31	211.62	0.5220	0.99473	9.2164	12.2164	0.0407	0.99997
		48	109.16	188.44	0.4625	0.99582	9.2293	12.1619	0.0408	0.99997
	32	16	105.05	179.24	0.4452	0.99622	9.2941	12.3704	0.0410	0.99997
		32	98.09	167.61	0.4160	0.99669	9.2924	12.3753	0.0410	0.99997
		48	90.91	156.08	0.3857	0.99713	9.3052	12.2985	0.0411	0.99997
	48	16	44.15	69.82	0.1922	0.99943	9.2894	12.3073	0.0410	0.99997
		32	29.68	45.13	0.1319	0.99976	9.2204	12.2763	0.0407	0.99997
		48	33.20	49.76	0.1471	0.99971	9.2539	12.2885	0.0409	0.99997

Table 5.3: Long-term forecast results of S&P test data using raw closing prices and log returns of closing prices

Window Size	Network		Raw closing prices				Log Returns of closing prices			
	1st stage	2nd stage	MAE	RMSE	MAPE	$R^2$	MAE	RMSE	MAPE	$R^2$
1	16	16	66.18	82.55	2.8688	0.90286	51.8805	64.2244	2.0434	0.87597
		32	66.42	82.71	2.8785	0.90249	51.8010	64.2635	2.0400	0.87582
		48	67.57	83.79	2.9247	0.89993	50.7598	63.3974	1.9985	0.87915
	32	16	66.44	82.34	2.8817	0.90337	51.9673	64.3116	2.0485	0.87564
		32	64.80	80.53	2.8170	0.90756	55.7266	67.4007	2.1975	0.86340
		48	66.50	82.03	2.8853	0.90409	53.0313	65.2688	2.0919	0.87191
	48	16	64.29	79.99	2.7971	0.90881	52.8052	64.9017	2.0807	0.87334
		32	66.27	81.86	2.8765	0.90449	54.3893	66.3863	2.1462	0.86748
		48	65.39	80.93	2.8418	0.90666	52.5934	64.8717	2.0741	0.87346
5	16	16	99.52	121.28	4.1622	0.79065	52.9455	64.2976	2.0886	0.87470
		32	98.89	120.59	4.1375	0.79303	53.0199	64.3084	2.0915	0.87466
		48	99.41	121.14	4.1578	0.79112	56.7592	67.8217	2.2416	0.86059
	32	16	95.29	116.27	4.0116	0.80758	59.3929	72.3776	2.3342	0.84123
		32	94.90	115.76	3.9975	0.80926	58.1324	71.2288	2.2839	0.84623
		48	95.00	115.73	4.0024	0.80936	58.4179	71.7840	2.2945	0.84382
	48	16	142.28	187.78	5.7629	0.49813	57.4819	71.5319	2.2590	0.84492
		32	144.85	190.54	5.8662	0.48328	61.0709	74.3845	2.4037	0.83230
		48	141.61	187.26	5.7352	0.50089	59.9546	73.5670	2.3584	0.83597
10	16	16	101.75	123.14	4.2457	0.78442	60.7164	80.8984	2.3805	0.79987
		32	102.36	123.68	4.2711	0.78254	60.1975	80.3131	2.3598	0.80275
		48	101.11	122.43	4.2205	0.78690	62.2980	82.4420	2.4438	0.79216
	32	16	66.62	81.54	2.8852	0.90547	61.0321	80.7656	2.3963	0.80053
		32	66.76	81.85	2.8897	0.90476	61.2571	81.5504	2.4046	0.79663
		48	67.35	82.26	2.9144	0.90380	61.4139	80.3959	2.4129	0.80235
	48	16	88.68	148.90	3.6729	0.68478	60.1248	74.0956	2.3714	0.83211
		32	92.02	155.60	3.7967	0.65578	56.9387	71.7729	2.2414	0.84247
		48	91.72	155.41	3.7864	0.65662	58.8870	73.4450	2.3203	0.83505
15	16	16	71.50	83.67	3.1171	0.90053	61.1218	79.8333	2.3937	0.80316
		32	70.79	83.03	3.0879	0.90204	60.4139	79.0248	2.3660	0.80713
		48	71.58	83.65	3.1208	0.90058	61.1422	79.6678	2.3949	0.80398
	32	16	211.32	314.77	8.3766	-0.40785	63.7013	81.6884	2.5002	0.79391
		32	210.72	314.37	8.3521	-0.40424	62.0721	80.2226	2.4353	0.80124
		48	211.11	314.63	8.3679	-0.40660	62.1894	80.6221	2.4394	0.79925
	48	16	68.82	83.38	2.9897	0.90122	59.8218	77.1688	2.3484	0.81608
		32	69.36	83.74	3.0123	0.90036	61.5102	78.4130	2.4172	0.81010
		48	69.73	84.18	3.0264	0.89932	61.0519	78.0314	2.3988	0.81195
24	16	16	97.51	116.55	4.1127	0.80719	56.0475	78.9394	2.1964	0.80548
		32	96.49	115.54	4.0718	0.81053	55.8717	79.2463	2.1887	0.80396
		48	95.79	114.49	4.0462	0.81393	56.1111	79.3530	2.1984	0.80343
	32	16	61.33	78.30	2.7008	0.91298	61.5369	79.6791	2.4130	0.80181
		32	61.83	79.05	2.7199	0.91131	63.8904	81.0258	2.5101	0.79506
		48	61.88	78.78	2.7225	0.91190	62.6184	80.3886	2.4574	0.79827
	48	16	75.04	91.02	3.2255	0.88242	54.0709	70.3440	2.1356	0.84553
		32	72.79	88.62	3.1374	0.88852	53.7931	69.8069	2.1247	0.84788
		48	73.29	89.14	3.1571	0.88721	53.6552	69.7929	2.1191	0.84794

Table 5.4: Long-term forecast results of DJIA test data using raw closing prices and log returns of closing prices

Window Size	Network		Raw closing prices				Log Returns of closing prices			
	1st stage	2nd stage	MAE	RMSE	MAPE	$R^2$	MAE	RMSE	MAPE	$R^2$
1	16	16	651.99	817.43	3.2030	0.92130	575.6115	713.0406	2.5355	0.88160
		32	664.60	829.01	3.2610	0.91905	554.1286	689.3551	2.4399	0.88934
		48	660.92	820.38	3.2479	0.92073	552.4919	692.0979	2.4304	0.88846
	32	16	643.14	804.34	3.1675	0.92380	528.2125	673.9007	2.3199	0.89424
		32	647.63	809.81	3.1865	0.92276	602.6155	735.4357	2.6595	0.87405
		48	664.23	823.81	3.2630	0.92006	528.8901	670.5115	2.3248	0.89531
	48	16	629.45	785.82	3.1095	0.92727	561.6852	698.4747	2.4731	0.88639
		32	636.08	793.40	3.1386	0.92586	559.6221	703.4197	2.4616	0.88478
		48	647.90	805.48	3.1912	0.92358	572.4663	714.5664	2.5202	0.88110
5	16	16	1354.25	1745.57	6.1841	0.64157	583.8907	721.9168	2.5735	0.87769
		32	1334.96	1723.52	6.0981	0.65057	610.3459	745.2283	2.6947	0.86966
		48	1333.86	1719.68	6.0949	0.65213	572.9031	710.6742	2.5242	0.88147
	32	16	1392.96	1812.31	6.3396	0.61364	588.0429	721.9520	2.5939	0.87768
		32	1388.94	1807.41	6.3218	0.61573	584.3072	718.3002	2.5771	0.87891
		48	1364.06	1775.97	6.2128	0.62898	582.6229	716.7858	2.5692	0.87942
	48	16	972.64	1206.51	4.5962	0.82877	579.3880	719.9311	2.5540	0.87836
		32	979.46	1212.62	4.6293	0.82703	613.9655	750.9024	2.7124	0.86767
		48	975.88	1210.20	4.6112	0.82772	566.5414	702.0350	2.4950	0.88433
10	16	16	633.07	781.68	3.1392	0.92820	605.2163	750.0340	2.6532	0.86685
		32	623.13	765.99	3.0945	0.93105	613.8751	757.6987	2.6944	0.86412
		48	635.45	779.73	3.1502	0.92855	611.7435	755.8725	2.6840	0.86477
	32	16	804.83	955.57	3.9148	0.89270	638.1729	785.5308	2.8001	0.85395
		32	800.44	950.37	3.8948	0.89386	644.5937	791.5618	2.8293	0.85170
		48	806.60	957.90	3.9223	0.89217	633.4297	782.4309	2.7775	0.85510
	48	16	833.79	1043.21	3.9433	0.87211	663.7819	821.3189	2.9141	0.84034
		32	834.19	1043.13	3.9458	0.87213	666.8277	826.8532	2.9267	0.83818
		48	833.23	1043.09	3.9406	0.87214	673.3980	833.1543	2.9567	0.83571
15	16	16	696.77	862.75	3.4304	0.91258	624.3024	785.3183	2.7397	0.85271
		32	683.33	841.07	3.3712	0.91691	639.8254	798.3629	2.8114	0.84777
		48	704.42	869.59	3.4642	0.91118	636.2081	793.2626	2.7962	0.84971
	32	16	1874.70	2980.88	8.2726	-0.04364	630.1827	773.2048	2.7801	0.85722
		32	1790.50	2831.90	7.9244	0.05807	638.3242	786.8991	2.8140	0.85211
		48	1814.09	2877.54	8.0212	0.02747	636.1582	790.6737	2.8026	0.85069
	48	16	791.57	971.06	3.8305	0.88925	588.5270	742.7324	2.6001	0.86825
		32	789.43	968.47	3.8214	0.88984	596.9240	747.1950	2.6392	0.86666
		48	826.56	1011.38	3.9801	0.87986	598.3657	753.9272	2.6441	0.86425
24	16	16	658.06	828.40	3.2582	0.91950	692.9759	859.0261	3.0610	0.82210
		32	658.20	827.76	3.2590	0.91962	696.9250	863.8191	3.0784	0.82011
		48	655.11	822.76	3.2459	0.92059	703.0613	868.9272	3.1064	0.81798
	32	16	641.84	805.09	3.1891	0.92397	674.5746	845.6055	2.9813	0.82762
		32	650.30	813.83	3.2269	0.92231	672.6436	842.5281	2.9730	0.82887
		48	636.96	797.15	3.1661	0.92546	674.0590	842.5258	2.9796	0.82887
	48	16	1181.81	1451.69	5.5138	0.75279	612.6368	805.2216	2.6996	0.84369
		32	1197.90	1471.97	5.5834	0.74583	605.1485	801.3691	2.6654	0.84518
		48	1191.09	1465.06	5.5527	0.74821	611.3338	805.9974	2.6931	0.84339

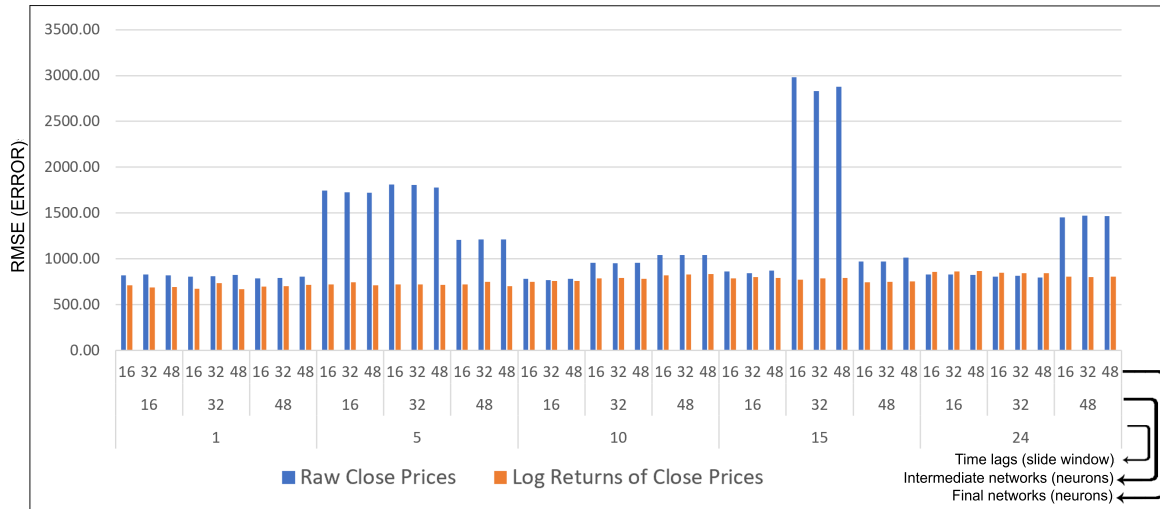


Figure 5.17: Comparison between test data RMSE of trained models to forecast DJIA long-term closing prices using raw data and log returns based on different time lags and number of neurons

the performance of models which use raw closing price and log returns is not significant. However, using log returns of closing prices showed slightly higher and more consistent results.

- It is clear from Figures 5.14 and 5.15 of the short-term forecast, that using one time lag (time step) to forecast the future is not sufficient. Yet, using longer time steps showed better accuracy which does not coincide with the random walk model which assumes that only the recent observation is the best way to forecast prices in the future.
- Comparing EWT-BGRU results shown in Table 5.5 with RW model forecast results shown in Table 5.6, we notice that EWT-BGRU outperforms the RW model for short-term time scale forecast. These findings may contradict the efficient market hypothesis for the short-term forecast since we use historical

price data to produce forecast performance better than RW. We may also notice that for the long-term time scale forecast, both EWT-BGRU and RW models produce comparable performance. Results of long-term forecasts may support the validity of the efficient market hypothesis for the long-term.

Table 5.5: Best trained EWT-BGRU models based on test data forecast results of S&P and DJIA for short and long-term

Datasets	Times cale	Raw closing prices				Log Returns of closing prices			
		MAE	RMSE	MAPE	$R^2$	MAE	RMSE	MAPE	$R^2$
S&P	Short-Term	1.68834	2.21816	0.07239	0.99993	1.53115	2.02342	0.06043	0.99988
	Long-Term	61.33463	78.29714	2.70081	0.91298	50.75976	63.39741	1.99854	0.87915
DJIA	Short-Term	13.51417	17.71492	0.06617	0.99996	8.64185	11.22386	0.03834	0.99997
	Long-Term	623.12674	765.99069	3.09446	0.93105	528.21249	673.90067	2.31995	0.89424

Table 5.6: Forecast results of the RW model using S&P and DJIA for short and long-term

Dataset	Time scale	MAE	RMSE	MAPE	$R^2$
S&P	Short-Term	11.19000	17.49600	0.44111	0.99132
	Long-Term	57.69060	69.25050	2.26590	0.83941
DJIA	Short-Term	103.30290	163.11330	0.46949	0.99580
	Long-Term	541.18800	679.77900	2.44303	0.92281

## 5.4 Multi-Step Forecast Using Hybrid Direct Recursive Method

All long-term forecasting models developed by this work so far use the direct method to forecast the future. The direct method uses rolling window of size  $k$  at time step  $t$  to forecast price after  $h$  time steps in the future. The time duration separating predictors and target variables affects the forecast accuracy significantly. Intuitively, the further the forecast horizon in the future, the higher the error term between forecasts and

actual prices. The direct method performs the forecasting as follows:

$$\hat{Y}_{t+h} = f_h(x_t, x_{t-1}, x_{t-2}, \dots, x_{t-k}) \quad (5.6)$$

However, forecasting one step ahead produces less error scale compared to forecasting more than one step. To use the advantage of the good performance of one-step forecasting models, the recursive long-term forecasting method employs one-step forecasting model to iteratively forecast many steps in the future. The intermediate produced forecasts are progressively used as input to forecast next time steps. Since actual observations of time duration between  $t_1$  and  $t_{h-1}$  cannot be used in the forecasting process, they can recursively be estimated using one-step forecasting models and employed to forecast time step  $t_h$ . Single one-step model can be developed to perform the forecasting as follows:

$$\hat{Y}_{t+1} = f(x_t, x_{t-1}, x_{t-2}, \dots, x_{t-k}) \quad (5.7)$$

$$\hat{Y}_{t+2} = f(\hat{Y}_{t+1}, x_t, x_{t-1}, x_{t-2}, \dots, x_{t-k}) \quad (5.8)$$

$$\hat{Y}_{t+h} = f(\hat{Y}_{t+h-1}, x_t, x_{t-1}, x_{t-2}, \dots, x_{t-k}) \quad (5.9)$$

The recursive method differs from the direct method by employing one step forecasting model to progressively forecast succeeding steps. The hybrid method combines both direct and recursive forecasting by developing many models for each intermediate future horizon scale. Estimation of intermediate time steps provides forecasting models with additional information which can be used to improve the accuracy. The

hybrid method defined as follows:

$$\hat{Y}_{t+1} = f_1(x_t, x_{t-1}, x_{t-2}, \dots, x_{t-k}) \quad (5.10)$$

$$\hat{Y}_{t+2} = f_2(\hat{Y}_{t+1}, x_t, x_{t-1}, x_{t-2}, \dots, x_{t-k}) \quad (5.11)$$

$$\hat{Y}_{t+h} = f_h(\hat{Y}_{t+h-1}, x_t, x_{t-1}, x_{t-2}, \dots, x_{t-k}) \quad (5.12)$$

We investigate forecasting long-term using the hybrid direct recursive method by the EWT-BGRU methodology proposed in Section 5.3. The forecasting experiments aim at forecasting up to one month (22 working days). We use the same experimental setup employed in Section 5.3. Initially, we perform preprocessing to raw closing prices by calculating log returns of 22 days as discussed in section 4.2. The log return series are then decomposed into many sub-series using EWT. We use experimental settings described in Section 5.1. A rolling window of size ten is defined to shape and arrange each resolution level of the decomposed data. The forecasting process starts by learning one-step model using ten time steps rolling window. The succeeding model uses the output of previous models in addition to the ten time steps input data defined at the beginning of the forecasting process. Illustration of the forecasting models with the corresponding input structure is shown in Figure 5.18. The intermediate and final network architectures designed using 48 memory cell units. The performance of the trained networks estimated using the test data. We compare the forecasting results of the proposed methodology with results of BGRU and random walk models. The results of the experiments are shown in Tables 5.7 and 5.8 for both S&P and DJIA datasets, respectively.



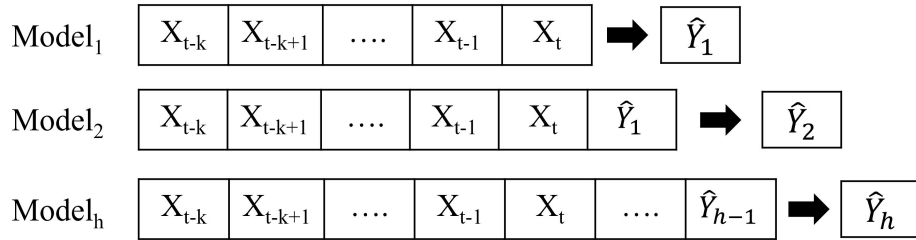


Figure 5.18: Input structure of multi-step forecasting models using hybrid direct recursive method

Table 5.7: Results of S&P data for both BGRU and EWT-BGRU models using hybrid direct recursive multi-step forecasting for up to one month (22 working days)

Horizon	EWT-BGRU				BGRU			
	MAE	MAPE	RMSE	R2	MAE	MAPE	RMSE	R2
1	0.654	0.026	0.819	1.000	15.198	0.596	20.770	0.987
2	1.309	0.052	1.580	1.000	26.492	1.035	33.920	0.965
3	2.505	0.100	2.956	1.000	32.479	1.265	41.446	0.947
4	2.245	0.089	2.946	1.000	56.612	2.219	65.930	0.866
5	3.715	0.147	4.567	0.999	53.814	2.080	67.808	0.858
6	3.233	0.126	4.823	0.999	68.309	2.652	82.648	0.788
7	4.098	0.160	5.735	0.999	74.152	2.879	89.289	0.752
8	3.570	0.140	5.058	0.999	84.083	3.269	99.729	0.689
9	4.371	0.169	6.350	0.999	69.691	2.700	86.403	0.765
10	5.355	0.207	7.911	0.998	54.092	2.095	69.909	0.846
11	7.255	0.282	10.522	0.996	48.377	1.878	62.752	0.875
12	9.984	0.390	14.171	0.993	52.153	2.027	66.672	0.858
13	12.343	0.480	18.081	0.989	52.037	2.027	66.233	0.858
14	13.847	0.538	20.775	0.986	42.561	1.662	55.017	0.902
15	16.459	0.638	24.536	0.980	67.677	2.636	83.318	0.774
16	18.373	0.712	27.180	0.975	68.067	2.648	84.409	0.767
17	21.736	0.843	31.662	0.966	78.177	3.034	95.838	0.698
18	23.874	0.925	35.842	0.956	92.484	3.566	113.979	0.571
19	26.620	1.032	40.055	0.945	111.258	4.271	136.783	0.379
20	36.942	1.432	54.237	0.899	163.710	6.298	191.539	-0.224
21	49.578	1.929	66.873	0.846	218.515	8.419	249.774	-1.094
22	47.097	1.823	68.388	0.838	235.547	9.061	270.281	-1.467

Table 5.8: Results of DJIA data for both BGRU and EWT-BGRU models using hybrid direct recursive multi-step forecasting for up to one month (22 working days)

Horizons	EWT-BGRU				BGRU			
	MAE	MAPE	RMSE	R2	MAE	MAPE	RMSE	R2
1	8.830	0.038	12.180	1.000	206.352	0.885	270.483	0.983
2	13.791	0.061	18.027	1.000	514.276	2.201	635.611	0.908
3	22.332	0.099	28.310	1.000	599.660	2.550	763.237	0.866
4	21.553	0.094	30.456	1.000	909.686	3.918	1067.583	0.736
5	41.623	0.184	50.536	0.999	790.846	3.372	981.706	0.776
6	33.744	0.146	50.571	0.999	931.716	3.990	1125.027	0.704
7	34.058	0.147	49.731	0.999	1114.987	4.784	1317.155	0.592
8	34.839	0.151	50.857	0.999	1046.306	4.520	1212.366	0.652
9	41.775	0.180	59.907	0.999	1302.585	5.601	1511.495	0.457
10	52.313	0.224	75.326	0.999	1378.630	5.916	1606.955	0.383
11	69.080	0.298	98.537	0.998	1332.248	5.692	1582.033	0.398
12	92.370	0.398	131.449	0.996	1444.593	6.186	1687.918	0.310
13	118.820	0.512	171.403	0.993	1536.617	6.579	1786.481	0.221
14	136.132	0.585	202.217	0.990	1507.865	6.436	1775.716	0.226
15	163.466	0.700	241.147	0.985	1729.185	7.409	1986.507	0.025
16	187.603	0.803	275.027	0.981	1798.691	7.690	2079.945	-0.075
17	249.014	1.072	360.844	0.967	1871.133	7.947	2237.633	-0.251
18	290.669	1.244	443.223	0.949	1997.501	8.404	2504.516	-0.577
19	341.055	1.459	505.947	0.934	2475.474	10.372	3156.441	-1.518
20	381.923	1.635	556.146	0.920	2465.197	10.285	3219.789	-1.636
21	453.989	1.952	624.640	0.898	2821.113	11.810	3592.239	-2.302
22	446.720	1.914	652.851	0.888	2767.154	11.565	3554.026	-2.254

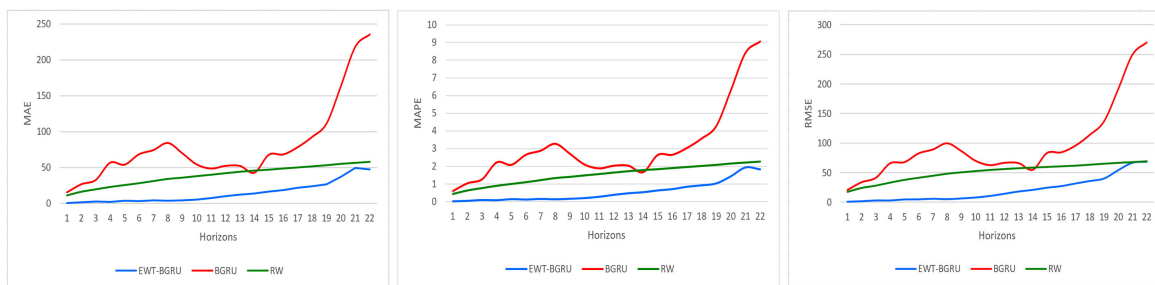


Figure 5.19: Comparison between EWT-BGRU, BGRU, and RW models based on Curves of MAE, MAPE, RMSE for multi-step forecasting of S&P data

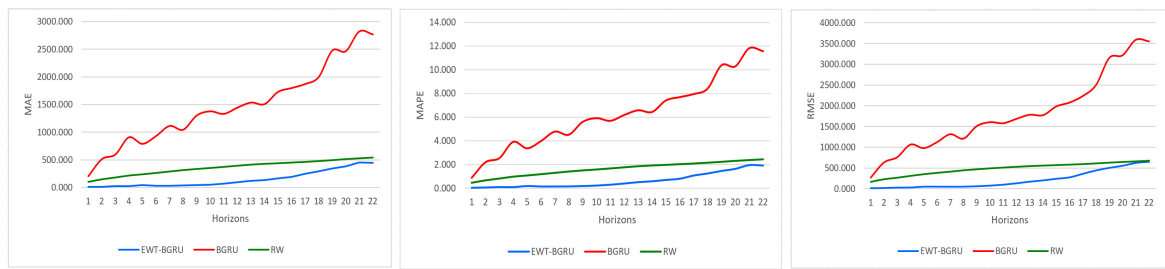


Figure 5.20: Comparison between EWT-BGRU, BGRU, and RW models based on Curves of MAE, MAPE, RMSE for multi-step forecasting of DJIA data

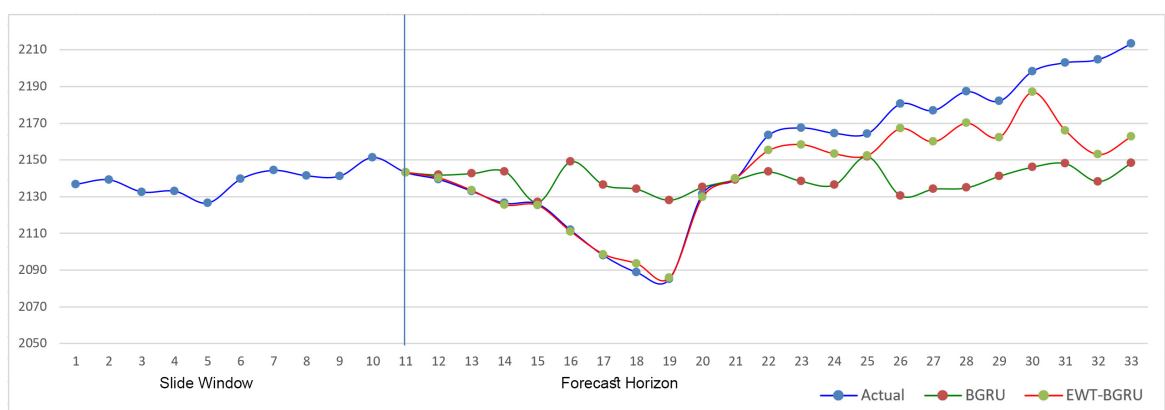


Figure 5.21: Multi-step forecast curves for BGRU and EWT-BGRU models with actual prices of first sample of S&P test data

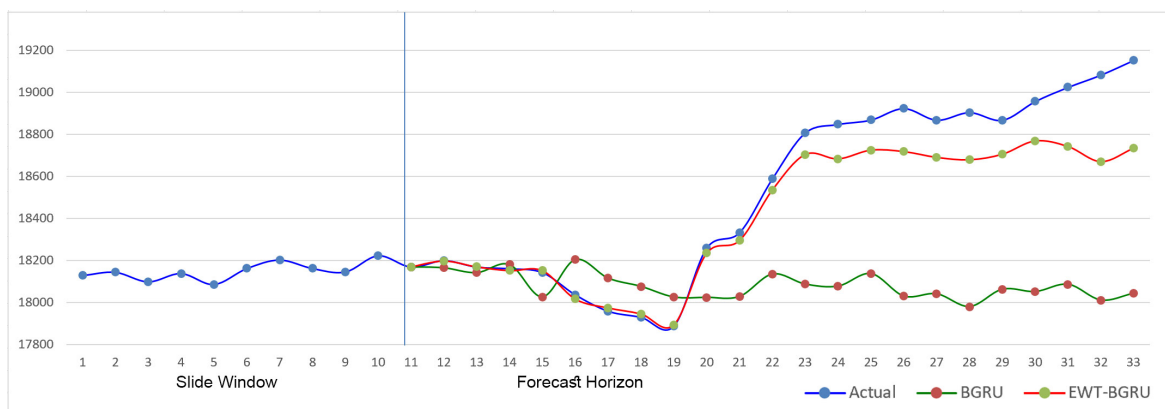


Figure 5.22: Multi-step forecast curves for BGRU and EWT-BGRU models with actual prices of first sample of DJIA test data

The results achieved by EWT-BGRU show superior performance compared to both BGRU and RW models. Figures 5.19 and 5.20 show that the error generated from EWT-BGRU forecasts is less than the error of the RW model, whereas error produced by BGRU model forecasts is higher than the error of the RW model. It is also clear from Figures 5.21 and 5.22 that forecasting the future for up to two weeks (ten working days) produces high accuracy that almost matches actual prices. Forecasts of more than two weeks start to gradually deviate from actual prices. The reason may return to new emerging patterns in prices which may not be captured using series of prices (slide window) recorded before more than two weeks. The reason also might be caused by occurrence of some new financial and economical events that may impact changes of prices. Evidently, keeping level of error scale below random walk model error curve means that we still have an advantage over the market, since the best available information are the most recent observations or current prices, therefore, forecasts of EWT-BGRU even after one month (22 working days) still provide useful information about price movements in the future.

The high accuracy produced by EWT-BGRU forecasting model is attributed to the ability of the EWT to separate each component of the price series based on price fluctuations and frequency bands included in price series. This kind of analysis enables BGRU networks to model data patterns independently. Considering and processing price series as signal, facilitates modeling price abrupt changes since each level of price fluctuation represents certain level in the frequency bands of the signal. Moreover, using a second level training network to combine multilevel resolution forecasts provides extra level of processing to reduce error produced from first level training network. It

also eliminates the need to any information related to the signal for the decomposition inverse process since information about signal in the future is not available.

## 5.5 Multiresolution Analysis and Technical Indicators

Performing forecast by means of TIs is one of the most common forecasting methods in literature. Many research used TIs as input to the learning methodologies. In this section we investigate combining TIs to our multiresolution architecture proposed in Section 5.3 which attains the highest forecasting performance so far. Our proposed architecture described in Section 5.3 is designed using two learning stages. The first stage performs learning and training using the decomposed data generated by the empirical wavelet decomposition method. It is designed to forecast new future values for each resolution level. The second stage uses forecasts generated from the first stage to forecast final prices. We add TIs values as input to the second stage to perform final forecast. The updated architecture is illustrated by Figure 5.23.

There are several TIs in literature employed to forecast stock prices. Determining which TIs to use in the forecasting process is an important factor in developing efficient forecasting models. To determine which TIs to use as input to our forecasting model, we survey 37 studies to identify most common TIs adopted by different studies to perform forecast operations. The surveyed studies are listed in Tables 2.3 and 2.4. We identify ten TIs that are widely used in forecasting approaches. We select TIs that are used by higher number of studies. The selected TIs along with number of studies

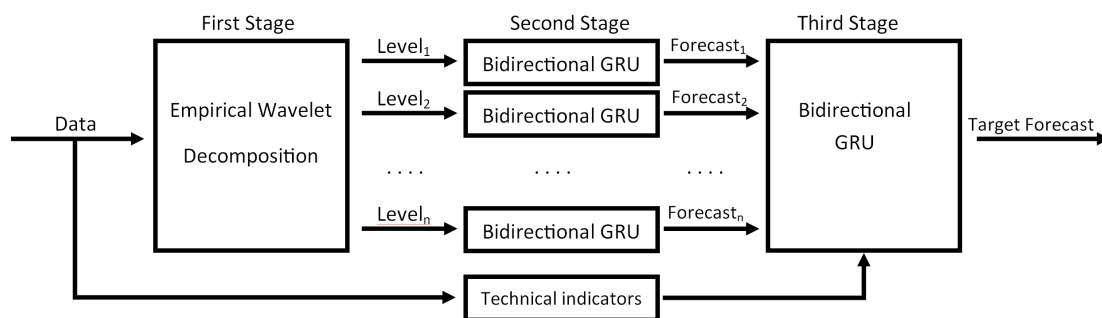


Figure 5.23: Design of decomposition and learning methodology combined with technical indicators

that use each indicator are shown in Table 5.9. Therefore, we use these indicators as input features into our forecasting approach. Description of the TIs used by this study is included in Appendix C.

Table 5.9: Selected technical indicators and the corresponding number of studies

Indicators	No. of Studies
SMA	29
RSI	28
Williams R%	24
Stochastic %K	21
MACD	20
Stochastic %D	18
EMA	17
ROC	14
Momentum	13
CCI	11

Basically, most TIs require deciding time interval to be used in the calculation process. Commonly, in stock market technical analysis, choosing time period depends on personal experience and type of stocks. Based on experiments conducted in Section 5.3, using log returns as input data to the proposed architecture produces high, stable,

and consistent results. We adopt the same experiment setup used in Section 5.3, where log returns of closing price from historical data is used as input to the forecasting model to forecast log returns in the future. We use a slide window of size fifteen time lags to reshape the data as per the time lags methodology adopted by this work which has been described in section 3.4. The developed models designed using 48 neurons which found to be effective in many models developed in most experiments conducted in previous sections.

Experiments are conducted to evaluate the impact of using TIs in financial time series forecasting. Therefore, we keep all experimental settings fixed and just change TIs which are used as input data along with forecast of the decomposed historical time lags as described earlier. We investigate and evaluate all combinations of the ten TIs for the two datasets and for the two time scales. Accordingly, we execute (4096) trials representing all possible TIs combinations for both S&P and DJIA benchmark datasets for both short-term (one day) and long-term (22 working days) one month in the future. We show only the results of ten models which produce the highest performance accuracy based on evaluation measures using test data. Tables 5.10 and 5.11 show the results of short-term forecast for the two benchmark datasets. Similarly, Tables 5.12 and 5.13 show the results for long-term forecast.

Table 5.10: Best trained models for short-term forecasting based on performance of S&P test data with corresponding employed technical indicators

<b>TI<sub>s</sub></b>	<b>RMSE</b>	<b>MAE</b>	<b>MAPE</b>	<b><math>R^2</math></b>
SMA,MACD	2.072891	1.589026	0.062623	0.999875
SMA,MOM, MACD	2.074915	1.585859	0.062517	0.999875
SMA	2.07868	1.59368	0.062821	0.999875
MACD	2.087522	1.598501	0.063055	0.999873
SMA,EMA, MACD	2.090606	1.589391	0.062692	0.999873
SMA,MOM	2.091433	1.598672	0.063069	0.999874
MOM, MACD	2.094229	1.602426	0.063147	0.999873
MOM,	2.095015	1.604614	0.063291	0.999873
SMA,EMA, MOM	2.095263	1.596851	0.06299	0.999873
SMA,EMA, MOM, MACD	2.103967	1.610576	0.063486	0.999871

Table 5.11: Best trained models for short-term forecasting based on performance of DJIA test data with corresponding employed technical indicators

<b>TI<sub>s</sub></b>	<b>RMSE</b>	<b>MAE</b>	<b>MAPE</b>	<b><math>R^2</math></b>
SMA,MACD	11.55025	8.867546	0.039323	0.99997
MACD	11.58436	8.971536	0.039784	0.99997
MOM	11.58753	8.953808	0.039727	0.99997
SMA,MOM	11.60751	8.894454	0.039431	0.99997
SMA,MOM, MACD,	11.64987	8.938591	0.039607	0.99997
MOM, MACD	11.70321	8.896335	0.039426	0.999969
SMA,EMA	11.70649	8.951302	0.039631	0.999969
SMA,EMA, MOM, MACD	11.81264	8.95661	0.039656	0.999969
EMA, MOM	11.82494	8.984909	0.039797	0.999969
SMA,EMA, MACD	12.01208	9.07649	0.040163	0.999968



Table 5.12: Best trained models for long-term forecasting based on performance of S&P test data with corresponding employed technical indicators

<b>TIs</b>	<b>RMSE</b>	<b>MAE</b>	<b>MAPE</b>	<b><math>R^2</math></b>
SMA,CCI	77.12196	53.82822	2.104637	0.816305
ROC, MOM	74.71854	54.03905	2.114248	0.827575
RSI, ROC, MOM, CCI	74.52202	54.42259	2.140761	0.828481
EMA, CCI	76.08336	54.48944	2.137529	0.821219
SMA,EMA, MOM, CCI	82.08397	55.14226	2.149598	0.791906
SMA,ROC,	75.51996	55.33812	2.172967	0.823857
RSI, MOM, MACD	74.27474	55.37726	2.172906	0.829618
RSI, MOM	73.2972	55.40349	2.17635	0.834073
ROC, MACD, StochasticD	74.81034	55.50442	2.172406	0.827152
SMA,RSI	73.40194	55.69163	2.185955	0.833598

Table 5.13: Best trained models for long-term forecasting based on performance of DJIA test data with corresponding employed technical indicators

<b>TIs</b>	<b>RMSE</b>	<b>MAE</b>	<b>MAPE</b>	<b><math>R^2</math></b>
EMA, MOM, CCI, MACD	681.7108	550.9256	2.451498	0.889008
SMA,RSI, MACD	701.1583	548.7164	2.41944	0.882585
ROC, CCI, MACD	702.2335	555.5717	2.471731	0.882225
SMA,EMA, CCI, MACD	708.2653	557.3193	2.478914	0.880193
SMA,EMA, MOM, CCI, MACD	711.2577	574.8473	2.554347	0.879178
SMA,ROC, MACD	713.2154	565.9677	2.513816	0.878512
MOM, CCI	716.9817	561.0728	2.488989	0.877226
SMA,RSI, ROC, MACD	718.3025	567.8869	2.496063	0.876773
ROC, MOM, CCI, MACD	718.3445	567.6002	2.52711	0.876759
EMA, RSI, MOM, CCI	720.4978	580.2866	2.569296	0.876019

The results of the experiments show that:

- The best ten trained models for short-term use only four TIs to produce the best performance using both datasets. The best four TIs are SMA, EMA, MOM, and MACD. To support and validate this conclusion, we counted the number of times each technical indicator is used in the top (100) best models. Table 5.14 lists all TIs used in the top (100) best models along with number of models for which each corresponding technical indicator is used as input.

Table 5.14: Number of times each technical indicator is used in the top best 100 models for both S&P and DJIA datasets for short-term forecast

<b>Dataset</b>	<b>S&amp;P</b>	<b>DJIA</b>
MACD	53	52
EMA	53	49
MOM	50	50
SMA	49	51
ROC	22	20
WR	16	16
RSI	19	12
StochasticK	12	16
StochasticD	11	16
CCI	11	9

- The results of the long-term forecast show that five TIs have higher impact on the forecasting accuracy. The five TIs are MACD, SMA, MOM, ROC, and CCI. Table 5.15 lists all TIs used in the top (100) best models along with number of models for which each corresponding technical indicator is used as input.
- We notice that MACD, SMA, and MOM have higher impact on forecasting both short and long-term which emphasizes the positive impact of these three

indicators and suggests using them in the forecasting models for short and long-term forecast.

- The results of the experiments conducted in this section, shown in Tables 5.10, 5.11, 5.12, and 5.13, produce similar performance to that of the EWT-BGRU model which uses only log returns of closing prices without the integration of TIs. Apparently, adding TIs as input data to the EWT-BGRU models may not improve the performance, yet it produces an accuracy results higher than results of random walk model for the short-term forecast.

Table 5.15: Number of times each technical indicator is used in the top best 100 models for both S&P and DJIA datasets for long-term forecast

<b>Dataset</b>	<b>S&amp;P</b>	<b>DJIA</b>
MACD	51	58
SMA	49	47
MOM	45	49
ROC	49	40
CCI	36	44
EMA	21	44
RSI	31	32
WR	15	25
StochasticD	22	5
StochasticK	16	9

## 5.6 Benchmarking of Proposed Forecasting Models

Three kinds of forecasting approaches are proposed by this work:

- The first approach is developed using deep learning techniques by adopting variants of deep recurrent neural networks. The proposed architectures are

SLSTM, BLSTM, SGRU, and BGRU.

- The second type of forecasting models is based on performing data preprocessing and decomposition using multiresolution analysis. Two types of wavelet analysis which are stationary wavelet transform and empirical wavelet transform are used in the forecasting operations (more details are provided in Appendix D.3). The preprocessed data is fed into two types of deep architectures namely SLSTM and BGRU. The proposed architectures are EWT-BGRU, EWT-SLSTM, and SWT-SLSTM. Experiments are conducted using raw and log returns data.
- The third forecasting approach uses TIs as input to the best selected model which is developed using BGRU with empirical wavelet decomposition (EWT-BGRU). The experiments aim at investigating the impact of adding TIs to forecast short and long-term time scales.

Best models selected based on the forecasting performance using test data for each developed approach is shown in Tables 5.16, 5.18, and 5.17. The results are based on data for the period from 01/01/2010 to 29/06/2018 which are used in most of the experiments conducted by this study. The results of the TASI dataset are produced using forecasting models based on best parameters concluded from the experiments conducted using S&P and DJIA datasets. We investigate different networks architectures using 32, 48, and 16 cell units (neurons). We use different sizes of rolling windows varied between 1, 10, and 24 timesteps. The developed networks are trained for up to 800 epochs. The same experimental configuration used to develop the forecasting models is also employed for the experiments conducted using TASI dataset.

Performing comparison between variants of deep recurrent neural networks namely, SLSTM, BLSTM, SGRU, and BGRU, results in proposing BGRU as the best among the four evaluated architectures. Despite the high performance of the BGRU models for short-term forecast, they are unable to outperform random walk model. On the other hand, the integration of empirical wavelet transform into the forecasting process improves the performance significantly. The results of performing forecasts for each decomposed resolution level reveal patterns in the data and help the learning models to extract useful information from the data to forecast the future.

Table 5.16: Performance of forecasting models trained on S&P dataset for each proposed approach

Model	Data	Short-Term				Long-Term			
		RMSE	MAE	MAPE	$R^2$	RMSE	MAE	MAPE	$R^2$
SLSTM	Closing price	21.089	15.445	0.60210	0.98730	157.321	138.523	5.38350	0.28090
SGRU	Closing price	22.668	17.271	0.67290	0.98540	131.286	116.205	4.55167	0.49620
BLSTM	Closing price	18.370	12.750	0.50112	0.99020	84.861	71.012	2.79080	0.79070
BGRU	Closing price	17.610	11.570	0.45580	0.99104	65.126	49.260	1.95200	0.87520
Random Walk	Closing price	17.496	11.190	0.44110	0.99132	69.251	57.690	2.26500	0.83900
SWT-SLSTM	Closing price	90.506	70.688	3.70000	0.51700	261.659	443.101	23.13000	-13.17100
EWT-SLSTM	Closing price	11.785	7.920	0.56514	0.99572	200.177	151.348	6.37400	0.60570
Autoencoder-BGRU	TIs and Closing price	19.453	12.815	0.50710	0.98910	91.486	77.084	3.02400	0.71420
EWT-BGRU	Closing price	2.218	1.688	0.07239	0.99993	78.297	61.335	2.70081	0.91298
EWT-BGRU	Log Returns of Closing price	2.023	1.531	0.06043	0.99988	63.397	50.760	1.99854	0.87915
EWT-BGRU	TIs and Log Returns of Closing price	2.073	1.589	0.06262	0.99988	77.122	53.828	2.10464	0.81631

The combination of empirical wavelet analysis with SLSTM network surpasses the random walk model performance and produces higher accuracy in terms of short-term forecast (more details are provided in Appendix D.3). Interestingly, combining both BGRU with EWT outperforms all other forecast models developed by this work. Hence the random walk model implies that the best way to forecast new values in the future is to look into the most recent values in the past. The goal of producing forecasts

Table 5.17: Performance of forecasting models trained on DJIA dataset for each proposed approach

Model	Data	Short-Term				Long-Term			
		RMSE	MAE	MAPE	R2	RMSE	MAE	MAPE	R2
SLSTM	Closing price	97.06617	73.72221	1.01842	0.958114	312.5247	245.4714	3.368251	0.530398
SGRU	Closing price	95.91118	72.6193	1.002081	0.957368	344.6899	264.5928	3.64488	0.42876
BLSTM	Closing price	98.37343	74.74841	1.03172	0.955151	303.1167	240.838	3.300468	0.542834
BGRU	Closing price	93.47912	70.69195	0.978782	0.961153	341.0954	271.0361	3.713915	0.421097
Random Walk	Closing price	57.00601	41.50527	0.576057	0.985241	309.7339	236.9174	3.238895	0.446407
SWT-SLSTM	Closing price	102.7647	80.87954	1.234189	0.946732	438.9384	333.9867	4.76625	0.43871
EWT-SLSTM	Closing price	5.709127	4.49609	0.064597	0.999944	433.0479	320.5417	4.612354	0.653442
Autoencoder-BGRU	TIs and Closing price	73.85378	53.954	0.801523	0.98659	365.8683	287.2014	4.266223	0.671611
EWT-BGRU	Closing price	5.358447	4.197835	0.059955	0.999953	423.5878	313.1841	4.507908	0.668418
EWT-BGRU	Log Returns of Closing price	4.46899	3.56689	0.04969	0.999902	305.8661	237.5797	3.205838	0.456346
EWT-BGRU	TIs and Log Returns of Closing price	5.49201	4.288375	0.061255	0.99994	309.8572	235.5669	3.231088	0.462559

Table 5.18: Performance of forecasting models trained on TASI dataset for each proposed approach

Model	Data	Short-Term				Long-Term			
		RMSE	MAE	MAPE	$R^2$	RMSE	MAE	MAPE	$R^2$
SLSTM	Closing price	209.695	152.219	0.65490	0.99030	1474.376	1278.246	5.52350	0.51144
SGRU	Closing price	237.116	178.084	0.76437	0.98760	1263.396	1081.616	4.69570	0.63882
BLSTM	Closing price	182.042	124.435	0.54267	0.99260	877.117	726.986	3.19800	0.82700
BGRU	Closing price	184.373	128.010	0.55930	0.99240	779.106	530.350	2.81310	0.86060
Random Walk	Closing price	163.110	103.300	0.46900	0.99500	679.770	541.180	2.44300	0.92200
SWT-SLSTM	Closing price	3293.859	1472.678	6.35287	-1.35830	4927.731	3164.494	14.37543	-2.67653
EWT-SLSTM	Closing price	80.087	55.039	0.29228	0.99946	2423.028	1832.389	8.65374	0.50854
Autoencoder-BGRU	TIs and Closing price	194.802	128.000	0.56160	0.99160	1154.292	1019.787	4.52240	0.65410
EWT-BGRU	Closing price	17.715	13.514	0.06617	0.99996	765.991	623.127	3.09446	0.93105
EWT-BGRU	Log Returns of Closing price	11.224	8.642	0.03834	0.99997	673.901	528.212	2.31995	0.89424
EWT-BGRU	TIs and Log Returns of Closing price	11.550	8.868	0.03932	0.99997	681.711	550.926	2.45150	0.88901

closer to the actual values than the most recent observations in the past is satisfied for short-term using EWT-SLSTM and EWT-BGRU models and for long-term using EWT-BGRU model.

The adaptive decomposition of data using empirical wavelet analysis provides the proper mechanism to extract sparse representations of the data based on information included in the dataset. The ability to design basis wavelet function constructed based on the information involved in the data leads into developing forecasting models suitable to each resolution level. Performing prior analysis of Fourier spectrum of the data to identify optimal number of resolution levels is a key point to providing a successful decomposition process for time series forecasting. In addition, the use of bidirectional architecture to design GRU network improves the forecast accuracy significantly. The methodology of the bidirectional architecture in using past information in both directions (forward and backward) results in a better accuracy. The context of the input data extracted by the bidirectional architecture which is used in feeding GRU network produces efficient forecasting models especially for the short-term.

The experiments conducted by this work to study the impact of using TIs suggest that MACD, SMA, and MOM have the highest positive impact on the short and long-term forecasting using EWT-BGRU model. Despite the high performance produced by the integration of TIs into the input data of the forecasting process, the accuracy performance did not outperform best produced accuracy by EWT-BGRU models using only log returns as input data.

## Comparisons with Related Work

The best results of this work are achieved by EWT-BGRU model using log returns of closing prices. Our proposed architecture shows superior results compared to some recent related work in literature. Comparisons between the performance of our proposed architecture and some related work based on S&P and DJIA benchmark datasets are shown in Tables 5.19 and 5.20 for S&P and DJIA datasets, respectively. Same result values in the related work articles are used for comparison purposes. We use the same time duration used in experiments of the related work and the same settings of data split into testing and training. The results show that our proposed architecture produces superior performance compared to the list of related works in Tables 5.19 and 5.20. We use evaluation measures described in section 5.1 which are used by the related works to perform comparison experiments. The definition of the Theil U and SD measures are as defined in [14, 120], respectively.



Table 5.19: Comparison between our proposed approach and some related work using S&P dataset

Reference	Period		Model	Time scale	Measures	Related Work	EWT-BGRU
	from	to					
[14]	01/07/08	30/09/16	WSAEs-LSTM	1	Theil U	0.007	0.00041
					R	0.946	0.9997
					MAPE	0.011	0.00067
[103]	02/01/09	15/03/13	Firefly	1	MAPE	0.0141	0.00209
					MSE	0.00019	0.00000441
					RMSE	0.0141	0.0021
[101]	03/01/94	23/10/06	BFO	1	MAPE%	0.8108	0.305
				3		1.0105	0.4204
				5		1.2399	0.5142
				7		1.3912	0.662
				15		1.8365	1.73437
[102]	02/01/13	30/08/13	APMS	1	RMSE	0.0454	0.0365
					MAE	0.0357	0.0117
[120]	03/03/11	10/03/16	LDBN	1	MSE	0.00251	0.000016
					MAE	0.0403	0.00327
					SD	0.049	0.0039
[121]	03/01/94	23/10/06	BFO	1	MAPE%	0.606	0.305
				7		1.3852	0.662
				15		1.8091	1.73437
[122]	01/10/98	31/01/08	ARIMA-MLP	1	RMSE	12.62	1.3092
					MAPE%	0.64	0.0689
					MSE	159.33	1.71400464
					MAE	9.02	0.982
[123]	04/08/06	31/08/12	PCA-STNN	1	MAE	15.5181	1.2946
					RMSE	19.2467	1.661
					MAPE%	1.1872	0.1003

Table 5.20: Comparison between proposed approach and some related work using DJIA dataset

Article	Period		Model	Time Scale	Measures	Related Work	EWT-BGRU
	from	to					
[14]	01/07/08	30/09/16	WSAEs-LSTM	1	Theil U	0.007	0.0005
					R	0.949	0.9996
					MAPE	0.011	0.000821
[124]	02/01/03	31/12/05	ICA-CCA-SVR	1	MAPE	0.011	0.00065
					RMSE	16.54	8.55
					MAE	12.638	6.93
					MSE	273.56	73.1025
					$R^2$	0.9486	0.9977
[101]	03/01/94	23/10/06	BFO	1	MAPE%	0.6623	0.17
				3		0.9534	0.1848
				5		1.187	0.355
				7		1.3811	0.4083
				15		1.8893	1.289
[123]	04/08/06	31/08/12	PCA-STNN	1	MAE	192.1769	11.446
					RMSE	220.4365	14.566
					MAPE%	1.5183	0.0935
[121]	03/01/94	23/10/06	BFO	1	MAPE%	0.5848	0.17
				7		1.3229	0.4083
				15		1.8529	1.289

## CHAPTER 6

# CONCLUSIONS & FUTURE WORK

Financial market is affected by political and economical changes. Several kinds of factors play important role in determining daily prices. These factors in a way or another are reflected in historical prices which form the basic element of financial time series. Consequently, forecasting financial time series is considered a challenging problem which requires sophisticated approach to solve. This work explores different aspects of integrating a combination of deep learning, multiresolution analysis, and technical indicators into financial time series forecasting. The conclusions drawn by this study are discussed in Section 6.1. Scope and limitations of this study are discussed in Section 6.2. Suggestions for future work are discussed in Section 6.3.

## 6.1 Conclusions

In this study, we initially use autocorrelation analysis in addition to statistical randomness and white noise tests to generate a clue about the possibility of performing forecasting using the benchmark datasets. Consequently, several deep architectures such as stacked and bidirectional layers are exploited to train deep networks designed using deep recurrent neural networks such as LSTM and GRU. Three of the benchmark datasets namely, S&P, DJIA, and TASI indices, are used in the evaluation process. The training process depends on examining and investigating historical data based on multiple input structures including different sliding window sizes and several TIs. Four deep architectures are proposed, evaluated and compared using multiple input structures and different future horizon time scales. Through analytical comparative experiments established based on trial and error, best models are selected and compared to determine best deep learning technique, input structure, and network architecture.

The best selected deep learning architecture is combined with multiresolution analysis method to improve the forecasting accuracy. Performing preprocessing to input data using multiresolution analysis with empirical wavelet transform shows superior performance and produces accurate forecasting models. We compare and apply stock market forecasting using both stationary wavelet transform and empirical wavelet transform (more details are provided in Appendix D.3). To the best of our knowledge, our work is the first to introduce integrating empirical wavelet analysis into financial time series forecasting and the first to propose a time series forecasting approach us-

ing combination of empirical wavelet analysis and BGRU deep learning architecture. We perform experiments using raw closing prices and log returns of closing prices. Data transformation using log returns attains the best accuracy results for short and long-term future time scale forecast. The achieved results surpass the performance produced by the random walk model especially for short-term forecast which holds an evidence against the efficient market hypothesis discussed in Appendix A. Comparisons with related work summarized in Section 5.6 show the superiority of the forecasting approach proposed by this work.

To study the impact of TIs on the forecasting accuracy, several models are developed using different combinations of TIs. To determine which TIs are the most likely to improve forecast accuracy, we survey several financial time series forecasting studies which use TIs as input data. Based on survey results, ten of the most common TIs are selected to perform forecasting experiments using best architecture developed using EWT and BGRU. Experiments are conducted by trial and error to evaluate all the ten TIs combinations for short and long-term forecast. MACD, SMA, and MOM TIs found to have higher positive impact on forecasting both short and long-term time scale. Despite the high performance exhibited by models developed using TIs, it is not high enough to outperform models composed of empirical wavelet transform and BGRU. Similar conclusion devised by Hsu et al. [125] who argued that informational value of TIs may not hold very significant impact on forecasting financial time series. Same conclusion is supported by other studies in literature [15, 126].

Generally, we find that forecasting long-term error scale is higher than that of the short-term due to the time period separating between explanatory and target

variables. During this time period, emerging changes in trends may not be captured by the forecasting models. Accordingly, using big rolling window of time lags from the past to forecast the future may not help increasing the performance since distant past has less impact on the future.

Furthermore, the results of our study support conclusions drawn by studies [16, 18, 127] which argued against the weak form of the efficient market hypothesis which states that all consecutive price changes represent random departures from historical prices. Assuming isolation and independency between sequential prices implies the random walk model. Thus, developing a forecast methodology which utilizes previous prices to forecast the future with an accuracy higher than that of the random walk model can be employed as an evidence against the efficient market hypothesis. The higher the difference between performance of the two models, the stronger the evidence.

Evidently, our proposed methodology achieves high difference in the performance compared to random walk model for the short-term forecasting, but for the long-term, the difference is not significant. Therefore, it could be the case that we can approve the efficient market hypothesis for long-term forecast and argue against it for the short-term which coincides with arguments stated by Fama [128] and aligns with conclusions drawn by some studies [129–131].

## 6.2 Limitations

- The aim of this study is to forecast exact prices of financial time series considering S&P and DJIA indices as two selected benchmark financial time series.

The two benchmark datasets are used in comparisons with many related work discussed in Section 5.6.

- All conclusions made by this study is subject to experiments conducted using daily closing prices of S&P and DJIA indices datasets downloaded from Yahoo finance.
- All experiments of this work are implemented in Python using Keras open-source package for deep learning with TensorFlow backend <sup>1</sup>. The stationary wavelet analysis are performed using PyWavelets open-source library <sup>2</sup>. The empirical adaptive wavelet analysis is accomplished using empirical wavelet transforms Matlab toolbox [91]. We use Ta-Lib library<sup>3</sup> to define TIs over both benchmark datasets.
- Results of experiments in Section 5.6 conducted to compare our proposed methodology with some related works are subject to experimental training and testing data split and time periods described in the related work publications.
- Reproducing same results of experiments conducted by this study depends on the random initializations for network weight layers established by Keras library random initialization modules. Executing the experiments several times using seed number seven will lead to similar results.

<sup>1</sup><https://github.com/keras-team/keras>

<sup>2</sup><https://github.com/PyWavelets/pywt>

<sup>3</sup><https://www.ta-lib.org/>

## 6.3 Future Work

- Forecasting the long-term is a challenging problem which needs more analysis and investigation. The time distance separating the data employed to forecast the future has a significant impact on the accuracy. The further the time scale forecast period in the future, the weaker the forecasting model to capture movements in trend and abrupt changes of prices. Although our proposed methodology has produced slightly better performance than that of the random walk model, suggesting one methodology for both short and long-term forecast may not be sufficient and more sophisticated approach could be dedicated to long-term forecast only.
- Forecasting real time online intra-day prices is a more challenging problem which requires special access to data. More sophisticated methodology could be proposed to forecast new prices in real time.
- Due to the big time complexity required to study the impact of using large number of TIs, our study is restricted to ten TIs. The impact of higher number of TIs using multiple time periods can be addressed and investigated.

## APPENDIX A

# EFFICIENT MARKET HYPOTHESIS (EMH)

The question of whether financial stock markets can be predicted is an issue of discussion and debate. Some research doubts the possibility of forecasting financial stock markets which coincides with what EMH implies [132]. Other research works claim that this is not a clear-cut and it cannot be generalized on all financial markets cases [18, 127]. There are also some researchers who believe that forecasting stock market is possible, limiting their claim to short term forecasting only [129, 130]. Generally, three forms of EMH is discussed by [132], as follows:

- The weak form of the EMH states that all historical information is already reflected in the stock prices, which implies that forecasting future prices by analyzing past prices is impossible. It is also implied that the sequential dependency or pattern between future and past stock prices does not exist, meaning that information which determines price changes is not contained in the price



time series. This form of EMH rejects any kind of forecasting models which depends only on technical analysis methods based on past stock prices. Yet, it leaves a space for the possibility of using forecasting methods designed based on fundamental analysis methods.

- The semi-strong EMH implies the weak form and also states that prices always change as a response to new public information. Thus, future information or events are not predictable which makes forecasting stock returns impossible. This implies that neither technical analysis nor fundamental analysis will be able to consistently produce stock returns. This form of EMH rejects the possibility of using technical or fundamental analysis methods to perform stock forecasting.
- The strong EMH incorporates both of the two previous forms and further claims that hidden information contributes to the instant changes of prices, meaning that no one is able to predict stock markets no matter what information is available.

Obviously, the weak form of the EMH implies that the stock market price changes follow the Random Walk (RW) model meaning that all new price changes represent random departures from past prices. It also means that new price changes reflect only news in the future and past prices are independent of price changes in the future. However, Some studies [16–18] argue that there is no ultimate evidence for the validity of the EMH. They suggest that whether the market is predictable or not is still an open question.

## APPENDIX B

# FINANCIAL TIME SERIES

A time series is a series of values obtained at successive equal time intervals which can be graphed and represented as a sequence of discrete-time data. Stock market prices, wind speed observations, rain rate reads, etc, are examples of time series. Contrary to cross-sectional data, time-series data are data points ordered in time and can be plotted via line charts. The analysis of time series mainly comprises statistical techniques to characterize, describe, and model data patterns. Meaningful statistics can be extracted from the time series by means of statistical methods to perform analysis and capture other characteristics of the data. In the context of financial data, the price that a particular security is traded at can be recorded as a time series, where the last or the first recorded traded price at a given period of time is used as data point in the financial time series. In general, the time series then can be defined as a simple curve of historical prices for any particular security. Section B.1 of this Appendix introduces some concepts about stock market. Section B.3 describes overview about stock market data. We discuss technical and fundamental analysis for stock market forecasting in Section B.2.

## **B.1 Financial Stock Market**

The financial market is a general term referring to any marketplace wherein trading of securities occurs enclosing equities, bonds, derivatives and currencies. Stock markets are financial markets where investors buy and sell shares of publicly traded companies. It enables companies to access to capital from investors by giving them partial ownership represented by number of shares. Companies collect the money raised by providing shares of their stock to public investors. Once stock shares start trading by selling and buying them by traders, companies do not receive any funds from the trading process. Stockbrokers are the licensed authorized persons to sell and buy securities on behalf of investors. They act as a trading agents who provide investors with information, advice, and act as intermediaries between investors and stock exchanges. Stock market indices are indicators of the whole market or for a group of stocks in the market. They give an idea about stock market relative value at any time.

Stock analysts are professionals who perform research and market analysis to help investors make decisions, such as, buy, sell, or hold stock shares. They use some analysis methods to forecast and predict the behaviour of the stock market in the future. Stock analysis is a set of methodologies that can be used to make stock trading decisions by studying and evaluating current or historical data related to stock market. It uses many measurement tools to evaluate certain trading strategy for the goal of achieving best returns in as short time as possible. There are two types of stock market analysis methods, technical and fundamental analysis.

## **B.2 Technical and Fundamental Analysis**

Technical and fundamental analyses are basically used for forecasting stock prices in the future. Technical analysis mainly includes performing different types of statistical methods using past prices and volumes to predict future stock prices. Fundamental analysis on the other hand uses publicly available information, such as, accounting earnings, growth factors, and dividends to study intrinsic values of securities.

The theory behind technical analysis is that the constantly changing attributes of investors force share prices to move in trends. Technical analysts use methods such as price and volume charts to predict stock prices in the future. The main concern about this method is that following trading rules extracted by the study of charts is highly subjective, thus multiple trading rules can be extracted by different technical analysts using same charts.

Fundamental data includes global economy and firms data, such as debt, earnings, and other data that can be extracted from trading interactions, trading activities, the financial growing of companies, and the global economy. Professional financial analysts use this information to perform analysis on regular basis to companies and international economies to measure and estimate the intrinsic value of stock shares of companies and firms. This kind of analysis is intended to give an indication of the true value of stocks. Determining the best set of fundamental data to perform fundamental analysis of stock shares of companies is still an open question, moreover the accessibility to this kind of information may not be possible since most companies consider this information classified.

## B.3 Stock Market Data

In the context of stock market time series, not only the last traded price (closing price) of stock shares at time period is recorded, but also the first traded prices (open price) is also recorded in addition to the highest and lowest traded prices over the same time period. These time series are usually accompanied with the trading volume of stock share are also recorded and represented as time series.

The data generated from stock market trade interactions as it is quoted on the market are represented by seven main variables mainly used to reflect an estimation of stock price level and fluctuation during specific period of time. The available data classified using three main time frames, daily, weekly, and monthly. Based on the assumption that weekly and monthly data are already represented in daily data since it is a summarized form of daily prices, daily raw data and other types of data derived from it, is the main input data used in this thesis to develop the forecasting methodology. Many financial websites provide free access to this data in three time frames represented by the following seven variables:

**Date:** The time or date when the data was recorded.

**Opening:** The price of the first trade of the stock at the beginning of the trading time period specified by the date variable. Usually the opening price is different from the closing price of the previous time period due to the fluctuation of supply and demand forces, which determine the prices at which stocks are bought and sold upon the beginning of the trading time period.

**Closing:** Refers to the last price at which the stock is traded by at the end of the trading time period. Most daily, weakly, monthly traders consider the closing price in deciding trading decisions.

**High:** The highest price of the stock during the trading time period.

**Low:** The lowest price of the stock during the trading time period. The highest and lowest prices are used as input data in multivariate financial time-series forecasting. They are used as input to some TIs formulas.

**Adjusted Close:** Refers to the stock closing price adjusted for any stock offerings, dividends, and stock splits.

**Volume :** Refers to the amount of stocks traded during the time period.

## APPENDIX C

# TECHNICAL INDICATORS

Technical indicators are a technical analysis mathematical tools developed for interpreting and estimating market trend and behavior. Technical indicators can be used to measure different types of factors, such as the number of shares traded, the ratio of stocks rising to those declining, and the number of stocks making a new high or low<sup>1</sup>. Figure C.1 illustrates the application of different types of technical indicators including simple moving average, exponential moving average, weighted moving average, relative strength index, momentum, moving average convergence / divergence, and Stochastic oscillators<sup>2</sup>.

<sup>1</sup><http://www.businessdictionary.com/definition/technical-indicator.html>

<sup>2</sup><https://wikifinancepedia.com/e-learning/definition/trading-terms/>

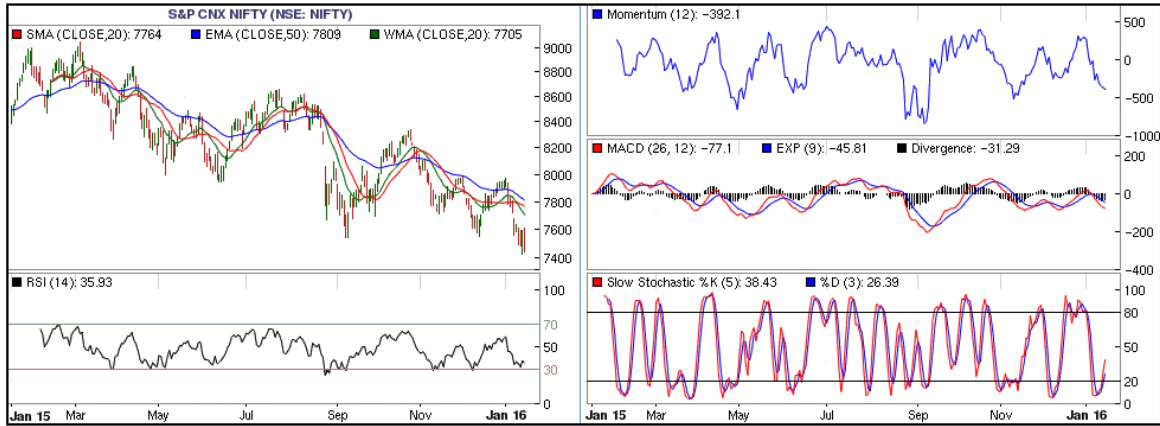


Figure C.1: Illustration of some technical indicators applied on S&P CNX Nifty index

## C.1 Simple Moving Average (SMA)

It uses a sliding window over specified period of time. Calculating the simple moving average (MA) for  $n$  samples would average out the prices for the first  $n$  prices as the first data element. The next data element would not include the earliest price, rather, it adds the price at  $n + 1$  and takes the average. The simple moving average of the previous  $n$  values ( $X_t, X_{t-1}, \dots, X_{t-(n-1)}$ ) can be computed as follows,

$$\begin{aligned}
 MA_t &= \frac{X_t + X_{t-1} + \dots + X_{t-(n-1)}}{n} \\
 &= \frac{1}{n} \sum_{i=0}^{n-1} X_{t-i}
 \end{aligned} \tag{C.1}$$



## C.2 Exponential Moving Average (EMA)

The exponential moving average applies weighting factors which decrease exponentially. For a series  $\bar{X}$ , it can be calculated recursively as follows:

$$EMA_t = \begin{cases} X_1, & t = 1 \\ \alpha \cdot X_t + (1 - \alpha) \cdot EMA_{t-1}, & t > 1 \end{cases} \quad (C.2)$$

Where the weighting is represented by a constant factor  $\alpha$  between 0 and 1 and  $X_t$  is the price at a time  $t$ .

## C.3 Weighted Moving Average (WMA)

A weighted average has multiplying factors to give different weights to each element of the elements for which the average is calculated. The weighted moving average is computed by multiplying each value in a given time duration by a decreasing factor. For duration of  $n$ -elements, the latest element has weight  $n$ , the second latest has  $n-1$ , and etc, down to zero

$$WMA_t = \frac{nX_t + (n-1)X_{t-1} + \dots + 2X_{t-n+2} + X_{t-n+1}}{n + (n-1) + \dots + 2 + 1} \quad (C.3)$$

## C.4 Momentum (MOM)

Momentum is a simple technical indicator because it is used to show the difference between prices for the first and the last of set of prices between  $t$  and  $t - n$  duration. It usually uses closing prices as input.

$$MOM_t = X_t - X_{t-n} \quad (C.4)$$

## C.5 Rate of Change (ROC)

Rate of change is calculated based on the difference between prices at the beginning and ending of certain time duration, but it is scaled by the old price,

$$ROC = \frac{X_t - X_{t-n}}{X_{t-n}} \quad (C.5)$$

## C.6 Relative Strength Index (RSI)

The RSI is a momentum indicator which is used to measure the speed and magnitude of directional price movements. The calculation of RSI depends on upward change ( $U$ ) or downward change ( $D$ ) for each element at time  $t$ .  $U$  is calculated by the difference

between prices at time  $t$  and time  $t - n$ .

$$D = 0$$

$$U = X_t - X_{t-n} \quad (\text{C.6})$$

Similarly  $D$  for at time  $t$  is calculated by the difference between price at time  $t - n$  and time  $t$ .

$$U = 0 \quad (\text{C.7})$$

$$D = X_{t-n} - X_t \quad (\text{C.8})$$

For certain set of  $n$  elements, Relative Strength  $RS$  is the ratio of exponential moving averages for  $U$  and  $D$ ,

$$RS = \frac{EMA(n) \text{ of } U}{EMA(n) \text{ of } D} \quad (\text{C.9})$$

Relative Strength Index (RSI) is calculated between 0 and 100 using the following,

$$RSI = 100 - 100 \times \frac{1}{1 + RS} \quad (\text{C.10})$$

## C.7 Moving Average Convergence / Divergence (MACD)

It is used to reflect an idea about the difference between long and short exponential moving average (EMA) of prices. The value of MACD is calculated by subtracting the longer exponential moving average ( $m$  elements  $EMA$ ) from the shorter exponential moving average ( $n$  elements  $EMA$ ). The signal line is the exponential moving average of  $k$  elements of MACD line. Traders use the MACD's histogram to identify when bullish or bearish momentum is high. When  $MACD < MACD\ Signal$  ||  $MACD\ Difference < 0$ , the price trend will be bearish and investors expect losses, otherwise the price increases (bullish).

$$MACD = EMA(n) - EMA(m) \quad (C.11)$$

$$MACD\ Signal = EMA_k(MACD) \quad (C.12)$$

$$MACD\ Difference = MACD - MACD\ Signal \quad (C.13)$$

Typical values of  $n$ ,  $m$ , and  $k$  are 12, 26, and 9, respectively<sup>3</sup>.

<sup>3</sup><https://www.investopedia.com/terms/m/macd.asp>

## C.8 Stochastic Oscillator

It is one of the fast indicators used in analysis. The values of stochastic oscillator is calculated as follows ,

$$\%K = 100 \times \frac{(X_t - L_n)}{(H_n - L_n)} \quad (\text{C.14})$$

$X_t$  denotes the most recent closing price,  $L_n$  denotes the low of the  $n$  previous trading period, and  $H_n$  denotes the highest price traded during the same  $n$  period. The %K refers to the current market rate for the currency pair whereas %D calculated 3-period moving average of %K. Stochastic Oscillator predicts the future trend which is for short time frame, that is why it is called as fast indicator.

## C.9 Williams R% (WR)

Similar to Stochastic Oscillator, Williams R% is a fast and simple technical indicator.

Below is the formula of the calculation,

$$WR = \frac{H(t \text{ to } t - n) - X_t}{H_{(t \text{ to } t-n)} - L_{(t \text{ to } t-n)}} \times 100 \quad (\text{C.15})$$

## APPENDIX D

# ADDITIONAL EXPERIMENTS

In this Appendix, we discuss some initial experiments conducted to perform basic analysis and investigation to datasets. They are conducted to help us form basic understanding of the underlying methodology proposed by this study. Evaluation and comparison between bidirectional and stacked LSTM with shallow neural networks are discussed in Section D.1. In Section D.2, multivariate input structure is used to perform the learning process using LSTM and GRU networks for short-term forecast. Section D.3 summarizes experiments conducted to investigate and compare using stationary wavelet transform and empirical wavelet transform for financial time series forecasting. Section D.4 evaluates using two of the most important TIs. The learning approach employs data preprocessing and denoising using deep GRU autoencoder. Both denoised data and TIs are fed into BGRU to accomplish forecasting.

## D.1 Evaluation of Bidirectional and Stacked LSTM

LSTM is one of the deep learning techniques proposed to process sequential data including time series. It uses gated units to remember and process long sequences of data. The structure of a unidirectional SLSTM depends on dealing and learning from data inputs on which its hidden states have passed through. It only sees information from the past. It has no mechanism to enable it to consider information in the future while predicting the present. On the other hand, BLSTM has the ability to process and learn from data in both directions; from future to past and from past to future. BLSTM while going through data combines forward and backward contextual information and use it to make forecasting or classification operations. To investigate and compare the two architectures and use the advantages of these features in stock market forecasting, we conduct experiments to build the forecasting models.

In this experiment, the historical data of the S&P for the period from 04/01/2010 to 30/11/2017 is downloaded from Yahoo finance to perform experiments. We split the data into two parts: training and testing. The first 80% duration of the whole data is allocated for training and the rest 20% duration is allocated for testing. We use a sliding window of size 10 (working days of two weeks) to forecast one-day ahead (short term) and 30-days ahead (long term). Initially, we conduct four sets of experiments by applying both BLSTM and SLSTM networks for short- and long-term forecast. Four network structures are designed for each applied technique by varying the number of units to be 4, 8, 16 and 32 neurons (memory cells). The networks are trained using one to ten epochs using training batches of size one for several runs and based on random

initials for each epoch. We also design MLP-ANN and simple LSTM networks for comparison and analysis.

The final results are dependent on averages of the performance of many different executions of the learning operations for every structure. The averages of the results per network structure and the total average of all networks structures for each technique are shown in Tables D.1, D.2, D.3, and D.4. For short-term forecast, the MAE and RMSE averages using normalized data of the BLSTM network are 0.023 and 0.0289, respectively. In contrast, the MAE and RMSE averages of the short-term SLSTM networks are 0.031 and 0.0382, respectively. For long-term forecast, the MAE and RMSE averages using normalized data of BLSTM networks are 0.0633 and 0.0746, respectively, and the MAE and RMSE averages of SLSTM network are 0.076 and 0.090, respectively. We can see that both LSTM networks show high performance for forecasting short-term and long-term prices. BLSTM networks produce higher performance and better convergence for short-term forecast and the difference gets much higher for long-term forecast.

Table D.1: MAE, RMSE, and  $R^2$  of BLSTM network results for short-term forecast

Networks	MAE				RMSE				$R^2$
	Testing	Training	Normalized Testing	Normalized Training	Normalized Testing	Normalized Training	Testing	Training	
4 Neurons Network	54.651	20.065	0.034062	0.012506	0.044202	0.015934	70.920	25.566	0.993
8 Neurons Network	34.768	19.352	0.021669	0.012061	0.027032	0.015135	43.372	24.284	0.994
16 Neurons Network	28.664	18.883	0.017865	0.011769	0.022718	0.015340	36.451	24.613	0.994
32 Neurons Network	29.677	18.496	0.018496	0.011527	0.021747	0.014549	34.893	23.344	0.994
Total average	36.940	19.199	0.023023	0.011966	0.028925	0.015240	46.409	24.452	0.994

For more analysis and investigation, we select the best produced models and compare them based on normalized RMSE, MAE, and  $R^2$ . The comparisons are designed to investigate the performance of BLSTM and SLSTM network structures for short-



Table D.2: MAE, RMSE, and  $R^2$  of SLSTM network results for short-term forecast

Networks	MAE				RMSE				$R^2$
	Testing	Training	Normalized Testing	Normalized Training	Normalized Testing	Normalized Training	Testing	Training	
4 Neurons Network	65.463	23.162	0.040801	0.014436	0.052363	0.018228	84.014	29.247	0.991
8 Neurons Network	54.990	22.902	0.034273	0.014274	0.042244	0.017729	67.779	28.445	0.992
16 Neurons Network	44.120	19.278	0.027498	0.012015	0.033103	0.015353	53.113	24.633	0.994
32 Neurons Network	34.950	18.384	0.021783	0.011458	0.025862	0.014663	41.495	23.526	0.994
Total average	49.740	20.891	0.031001	0.013020	0.038264	0.016465	61.393	26.418	0.993

Table D.3: MAE, RMSE, and  $R^2$  of BLSTM network results for long-term forecast

Networks	MAE				RMSE				$R^2$
	Testing	Training	Normalized Testing	Normalized Training	Normalized Testing	Normalized Training	Testing	Training	
4 Neurons Network	120.988	46.813	0.075407	0.029176	0.089146	0.036763	143.031	58.985	0.968
8 Neurons Network	110.066	48.459	0.068600	0.030203	0.079403	0.037441	127.399	60.072	0.966
16 Neurons Network	91.857	47.883	0.057251	0.029843	0.067557	0.037199	108.393	59.684	0.967
32 Neurons Network	83.687	48.576	0.052159	0.030276	0.062648	0.038041	100.517	61.035	0.965
Total average	101.650	47.933	0.063354	0.029875	0.074689	0.037361	119.835	59.944	0.966

Table D.4: MAE, RMSE, and  $R^2$  of SLSTM network results for long-term forecast

Networks	MAE				RMSE				$R^2$
	Testing	Training	Normalized Testing	Normalized Training	Normalized Testing	Normalized Training	Testing	Training	
4 Neurons Network	135.944	47.340	0.084729	0.029505	0.101257	0.036984	162.463	59.340	0.967
8 Neurons Network	124.323	47.807	0.077486	0.029796	0.091482	0.036919	146.780	59.235	0.967
16 Neurons Network	119.455	51.293	0.074452	0.031969	0.087484	0.039332	140.365	63.107	0.963
32 Neurons Network	108.131	47.780	0.067394	0.029779	0.080033	0.037301	128.411	59.849	0.967
Total average	121.963	48.555	0.076015	0.030262	0.090064	0.037634	144.504	60.382	0.966

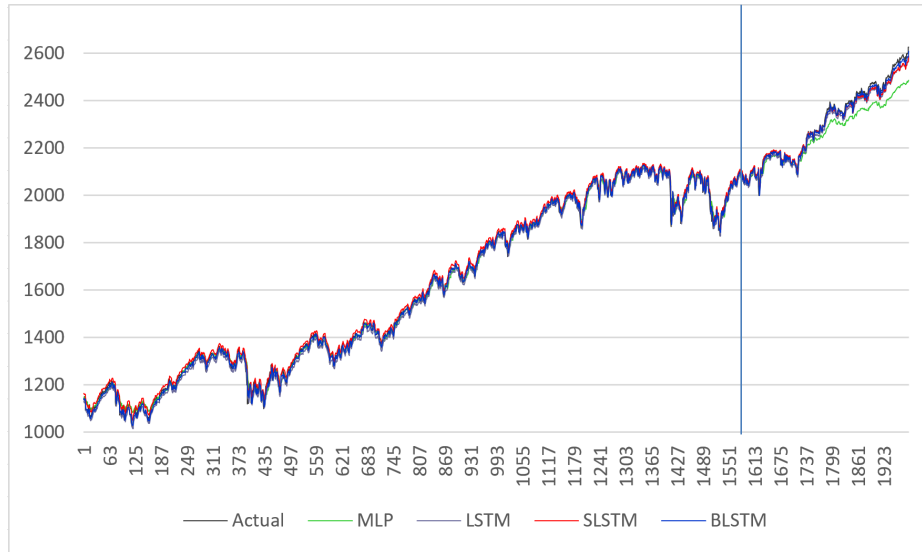


Figure D.1: The short-term forecast versus target curve for BLSTM, SLSTM, LSTM, and MLP Models

and long-term forecast. The selected models are compared with another two models developed using simple LSTM and MLP-ANN. The short-term forecast comparison is shown in Table D.5. The long-term forecast comparison is shown in Table D.6. The short- and long-term forecast versus target curves for all compared models are illustrated in Figures D.1 and D.2, respectively. The forecast curves plotted to clarify the difference between the four compared models. The results of the experiment show that:

- The BLSTM network attains higher performance and better convergence for short- and long-term forecasts.
- The difference between the performance of the short-term and long-term forecasts is big and the error scale of the long-term forecast is much higher than the short-term.

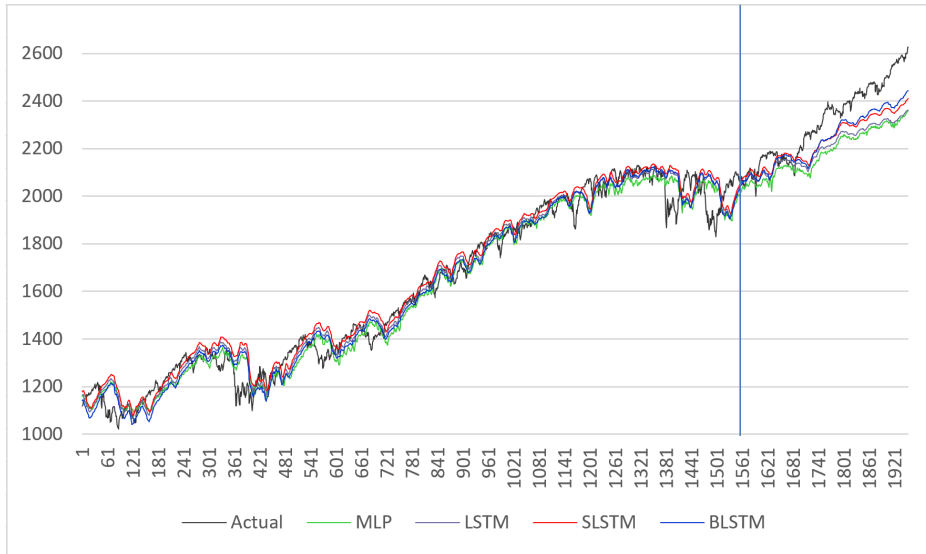


Figure D.2: The long-term forecast versus target curve for BLSTM, SLSTM, LSTM, and MLP Models

Table D.5: Comparison between best selected models for short-term forecast

	Raw Data			Normalized Data		
	MAE	RMSE	$R^2$	MAE	RMSE	$R^2$
<b>MLP-ANN</b>	51.37	62.18	0.995	0.03202	0.03875	0.995
<b>LSTM</b>	22.43	25.38	0.996	0.01398	0.01582	0.996
<b>SLSTM</b>	15.84	20.02	0.996	0.00987	0.01248	0.996
<b>BLSTM</b>	11.82	15.20	0.997	0.00736	0.00947	0.997

- Both SLSTM and BLSTM show better performance than both simple LSTM and traditional MLP-ANN.
- The random walk model forecast results are shown in Table D.7. Despite the good performance produced by BLSTM networks, it does not produce accuracy better than random walk model which coincides with what the efficient market hypothesis implies. Meaning that the best way to forecast tomorrow's price is by looking at today's.

Table D.6: Comparison between best selected models for long-term forecast

Network	Raw Data			Normalized Data		
	MAE	RMSE	$R^2$	MAE	RMSE	$R^2$
<b>MLP-ANN</b>	132.94	150.32	0.96	0.08285	0.09369	0.96
<b>LSTM</b>	114.29	134.31	0.96	0.07123	0.08371	0.96
<b>SLSTM</b>	90.95	106.50	0.95	0.05668	0.06637	0.95
<b>BLSTM</b>	84.11	97.16	0.96	0.05242	0.06055	0.96

Table D.7: Results of random walk model forecast for short and long-term

Dataset	Time scale	MAE	RMSE	$R^2$
S&P	Short-Term	8.47310	12.36740	0.99399
	Long-Term	50.67000	61.38590	0.83391

## D.2 Forecast Using Multivariate Analysis

In this section, we use multivariate input structure to perform the learning process.

We conduct several experiments to address the application of variants of deep RNN for forecasting financial time series. We compare and evaluate the use of SLSTM , SGRU , BLSTM, and BGRU architectures for short-term stock market forecasting.

We use multivariate input structure to perform the learning process. Moreover, we compare the performance of deep RNN variants with shallow neural networks. The

conducted experiments and evaluation procedure is based on historical data of S&P index for period from 01/01/2010 to 30/11/2017 downloaded from yahoo finance.

The experiments conducted using a window of size ten (working days of two weeks)

to forecast closing prices of one-day and thirty days ahead in the future. We apply

differencing to time series to transform it into stationary by stabilizing mean, avoiding

changes in time series levels, and reducing seasonality and trend. The used differencing

equation is :  $diff(x_t) = x_t - x_{t-1}$ .

After differencing the dataset, the multivariate time series values are normalized between the range  $(0, 1)$  using min-max normalization. The data consists of five input variables: Closing price, Opening price, Low price, High price, and Volume. We aim at forecasting the closing price at the end of every trading day using a range of ten preceding days for training. Each data sample formed by a sliding window of ten consecutive days with overlapping. The five daily price variables mentioned earlier are used as input to the training process to forecast the closing price in the future. The data is split into two parts: training and testing. The training part consists of the first 70% duration of the whole data and the testing part consists of the rest 30% duration of the data.

Several models developed in this experiment to forecast prices in the future using BLSTM, SLSTM, SGRU, BGRU, and MLP. Many networks are designed for each technique by varying the number of memory cells to be 8, 16, or 32 neurons. Each network is trained several times using from one to twenty epochs with different random initials each run. The averages of the performance of all network structures executions for each technique are collected to compare the final results of the experiments. The MLP network are constructed using different architectures with one, two, and three hidden layers and different number of neurons including (10|20|30) neurons and different number of iterations including (100|1000|10000|100000).

Table D.8 shows results averages of normalized data for each corresponding architecture. Comparisons between best models of each architecture are shown in Table D.9. The results show that all developed architectures produced close forecasting performance with slightly small differences. However, we can notice that models

Table D.8: Comparison between forecasting performance averages of several network structures using normalized data

Arch	Training			Testing		
	MAE	RMSE	$R^2$	MAE	RMSE	$R^2$
<b>BGRU</b>	0.06959	0.09249	0.99	0.07879	0.11146	0.99
<b>BLSTM</b>	0.06822	0.09158	0.99	0.07702	0.10991	0.99
<b>SGRU</b>	0.06851	0.09192	0.99	0.07734	0.11007	0.99
<b>SLSTM</b>	0.06793	0.09177	0.99	0.07604	0.10884	0.99
<b>MLP</b>	0.07701	0.10360	0.99	0.08641	0.12211	0.99

Table D.9: Comparison between best forecasting performance models based on testing results

Arch	Neurons	Epochs	Raw Data			Normalized Data		
			MAE	RMSE	$R^2$	MAE	RMSE	$R^2$
<b>BGRU</b>	8	7	11.286	16.381	0.99	0.07385	0.10719	0.99
<b>BLSTM</b>	8	4	11.212	16.337	0.99	0.07337	0.10690	0.99
<b>SGRU</b>	32	14	11.229	16.305	0.99	0.07348	0.10669	0.99
<b>SLSTM</b>	16	3	11.136	16.283	0.99	0.07287	0.10655	0.99
<b>MLP</b>	10	100	11.494	16.89	0.99	0.07556	0.10719	0.99

developed using SLSTM architecture attained higher performance and better convergence. Based on the RMSE and MAE measures on testing data, the best performance is produced by SLSTM network trained for 3 epochs using a structure of 16 neurons. The forecasting curves of the developed models versus actual data curve are illustrated in Figure D.3. The forecasting curves are plotted using the last 100 days of the test data to illustrate the difference between developed models. The learning curves are illustrated in Figure D.4.

According to results we may conclude the following:

- In Table D.9, we can see that the stacked architectures including SLSTM and SGRU produce performance better than BLSTM and BGRU models developed using bidirectional architecture. However, the compared averages of the results in Table D.8 show that both BLSTM and SLSTM produce RMSE and MAE

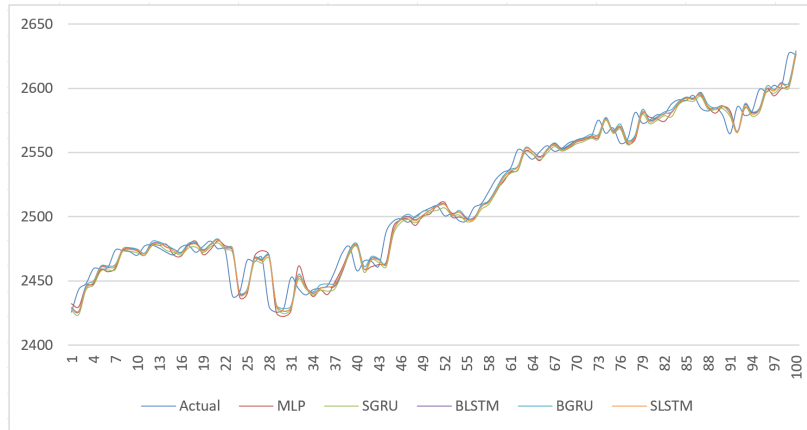


Figure D.3: The short-term forecasting curve for actual data and forecasts using BLSTM, SLSTM, SGRU, BGRU, and MLP Models

values better than BGRU and SGRU.

- Generally, all developed architectures showed a performance better than that of the MLP network.
- The difference between trained models is so small which may not help making valid conclusions about the best architecture based on this result.
- Though they have close performance accuracy, the best results produced by experiments in this section does not outperform those conducted in Section D.1.
- The RMSE of the developed architectures in this section is close to the RMSE of the random walk model results shown in Table D.7, which urges us to perform deeper analysis and conduct more experiments to improve forecast results.

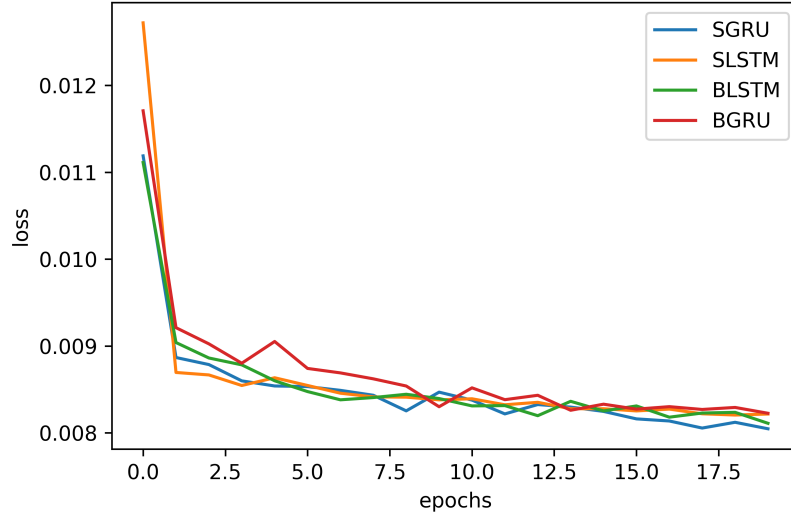


Figure D.4: Learning curve of BLSTM, SLSTM, SGRU, and BGRU Models for short-term forecasting

## D.3 Evaluation of Empirical and Stationary Wavelet Transforms

The experiments summarized in this section apply, compare and evaluate two types of multiresolution analysis methods; SWT and EWT. We investigate the application of EWT for multiresolution analysis in financial time series forecasting based on deep learning approach. The methodology followed to construct the network architecture is illustrated in Figure D.5. The proposed architecture consists of three stages. The first stage performs data analysis and decomposition into many sub-series. The second stage trains multiple intermediate networks based on the number of resolution levels generated from the previous stage. Forecasts of each sub-series is produced by corresponding intermediate networks which are constructed using two layers of LSTM and one dense layer. The first LSTM layer uses  $N$  memory units which varies between



8, 16, and 32 based on experiments conducted to select the best network configuration, whereas the second LSTM layer uses  $N/2$  memory units. The forecast of each resolution level is produced by a dense layer which uses a linear activation function. The third stage of the proposed architecture receives forecasts produced by the second stage as input features. Similarly, It is constructed using two stacked layers of LSTM and one dense layer. We use the same experimental settings discussed in Section 5.1. The numbers of time steps used to form data are five and ten working days (working days of one and two weeks).

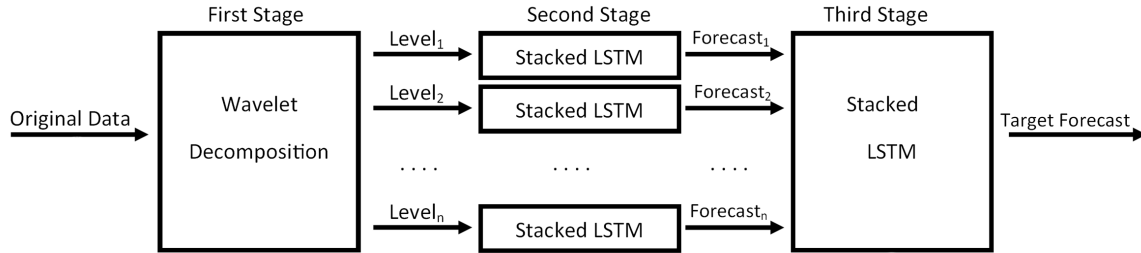


Figure D.5: Methodology proposed to design the MRA-SLSTM architecture

In order to evaluate the proposed approach, we use Mackey-Glass (MG) [133] chaotic time series introduced as a model of white blood cell production. The following time-delay ordinary differential equation used to generate the time series,

$$\frac{dx(t)}{dt} = -bx(t) + \frac{ax(t - \tau_1)}{1 + x(t - \tau_1)^{10}} \quad (\text{D.1})$$

Where  $\tau$  is a delay parameter which must be greater than 16.8 for generating chaotic time series. The parameters selected for generating the time series are  $a = 0.2$ ,  $b$

$= 0.1$ ,  $\tau = 17$ , and the initial value  $x_0 = 1.2$ . The time series composed of the last 2000 of 2123 samples generated for the differential equation. The MG time series is split into two parts; the first 75% duration of the data is allocated for training and validation; the remaining 25% is allocated for the testing part. Figure D.6 shows plot of data generated by the Mackey-Glass time series.

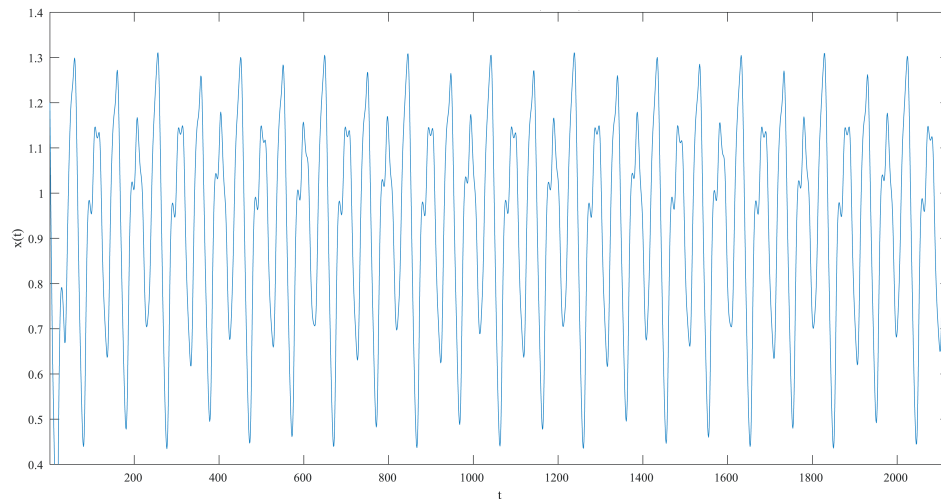


Figure D.6: Plot of data generated by the Mackey-Glass time series ( $a = 0.2$ ,  $b = 0.1$ ,  $\tau = 17$ , and  $x_0 = 1.2$ )

The short-term forecasting results generated from the proposed EWT-SLSTM and SWT-SLSTM architectures are as shown in Table D.10. The results are classified based on different number of neurons for the intermediate and final networks layers and based on different window sizes varying between five and ten time steps. It is also classified based on normalized and original raw data and based on the type of multiresolution analysis method using EWT and SWT. The  $R^2$  is calculated for the whole dataset including training and testing parts. Figure D.7 illustrates the difference between actual data and forecasts of the two best short-term forecasting

Table D.10: Results of forecasting using EWT-SLSTM and SWT-SLSTM models for Mackey-Glass time series normalized and original test data

Window	Network		Stationary wavelet transform					Empirical wavelet transform								
	1st stage	2nd stage	MAE		RMSE		$R^2$	MAE			RMSE		$R^2$			
			Norm.	Origin.	Norm.	Origin.	Value	Norm.	Origin.	%Improv	Norm.	Origin.	%Improv	Value	%Improv	
5	8	8	0.11434	0.10015	0.15808	0.13846	0.67061	0.00255	0.00223	97.77	0.00326	0.00286	97.94	0.99987	49.10	
		16	0.09990	0.08750	0.15038	0.13171	0.74139	0.00252	0.00221	97.48	0.00322	0.00282	97.86	0.99987	34.86	
		32	0.06293	0.05512	0.10755	0.09421	0.87956	0.00252	0.00221	96.00	0.00322	0.00282	97.01	0.99987	13.68	
	16	8	0.11190	0.09801	0.15973	0.13990	0.69960	0.00256	0.00224	97.71	0.00331	0.00290	97.93	0.99986	42.92	
		16	0.08177	0.07162	0.12885	0.11286	0.81233	0.00255	0.00223	96.89	0.00327	0.00286	97.46	0.99987	23.09	
		32	0.07829	0.06857	0.12525	0.10971	0.82800	0.00252	0.00221	96.78	0.00322	0.00282	97.43	0.99987	20.76	
	32	8	0.14229	0.12463	0.19145	0.16769	0.50372	0.00255	0.00223	98.21	0.00327	0.00287	98.29	0.99987	98.50	
		16	0.08492	0.07438	0.13353	0.11695	0.79384	0.00255	0.00222	97.02	0.00324	0.00284	97.57	0.99987	25.95	
		32	0.06203	0.05433	0.12037	0.10543	0.89556	0.00253	0.00222	95.92	0.00323	0.00283	97.32	0.99987	11.65	
	10	8	8	0.11973	0.10487	0.16464	0.14421	0.63091	0.00287	0.00252	97.60	0.00378	0.00331	97.70	0.99984	58.48
			16	0.07883	0.06905	0.12676	0.11103	0.82431	0.00284	0.00249	96.40	0.00366	0.00320	97.12	0.99985	21.30
			32	0.07797	0.06830	0.12619	0.11053	0.82679	0.00285	0.00250	96.34	0.00364	0.00319	97.11	0.99985	20.93
16		8	0.11374	0.09963	0.16935	0.14833	0.68524	0.00258	0.00226	97.73	0.00340	0.00298	97.99	0.99986	45.91	
		16	0.08951	0.07840	0.13492	0.11817	0.78426	0.00257	0.00225	97.13	0.00333	0.00292	97.53	0.99986	27.49	
		32	0.06875	0.06022	0.11509	0.10081	0.86713	0.00253	0.00221	96.32	0.00323	0.00283	97.19	0.99987	15.31	
32		8	0.12701	0.11125	0.17897	0.15675	0.61429	0.00263	0.00230	97.93	0.00345	0.00302	98.07	0.99986	62.77	
		16	0.08898	0.07793	0.13534	0.11854	0.79356	0.00256	0.00225	97.12	0.00330	0.00289	97.56	0.99987	26.00	
		32	0.06592	0.05774	0.11319	0.09914	0.86528	0.00259	0.00227	96.07	0.00333	0.00291	97.06	0.99987	15.55	

models produced using EWT-SLSTM and SWT-SLSTM approaches.

The S&P data is also used to evaluate the proposed architecture. We use the historical data for the period from 01/01/2010 to 20/02/2018 downloaded from Yahoo finance. The conducted experiments on S&P dataset composed of two parts. The first part of the experiments applies multiresolution analysis to data and performs the training process to do short-term forecasting, and the second part performs the training process to do long-term forecasting. The short-term forecasting aims at forecasting the close price one-day ahead in the future, whereas the long-term aims at forecasting the close price 30-days ahead in the future.

The short-term forecasting results generated from the proposed architecture for both EWT and SWT are shown in Table D.11. The results are classified based on number of neurons and slide window size (time steps). They are also classified based on type of multiresolution analysis method. The  $R^2$  is calculated for the whole dataset including training and testing parts. The best performance is achieved using EWT-SLSTM architecture. The best achieved RMSE, MAE, and  $R^2$  are 0.0019, 0.0014,

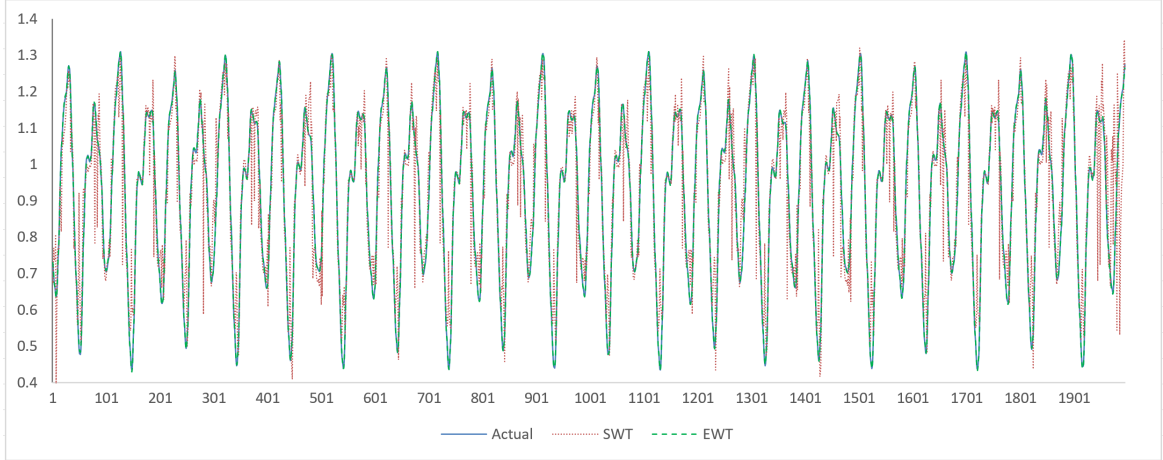


Figure D.7: Mackey-Glass data curves of actual values versus forecasts using best selected models of EWT-SLSTM and SWT-SLSTM architectures

and 0.99997, respectively. These results were produced using ten time steps input features to LSTM network architecture designed of 16 neurons in the intermediate network and 32 in the final network. For the SWT-SLSTM architecture, the best performance for short-term forecasting was achieved using ten time steps input features to a network architecture composed of 16 neurons in both intermediate and final LSTM networks. As we can see in Table D.11 that the best SWT-SLSTM model produced 0.0489 for RMSE, 0.0382 for MAE, and 0.97504 for  $R^2$ . Figure D.8 illustrates the difference between actual data and forecasts of the two best short-term forecasting models produced using EWT-SLSTM and SWT-SLSTM approaches.

For the long-term forecasting using S&P dataset, the conducted experiments aim at forecasting the exact close price 30-days ahead in the future. We allocate the last 30% of the whole data for the testing, while the remaining 70% is allocated for training and validation. The results of both stationary and empirical wavelet transforms are shown in Table D.12. The highest performance of the EWT analysis method is produced

Table D.11: Results of short-term forecasting using EWT-SLSTM and SWT-SLSTM for S&P normalized and original test data

Window	Network		Stationary wavelet transform					Empirical wavelet transform								
	1st stage	2nd stage	MAE		RMSE		$R^2$	MAE			RMSE		$R^2$			
			Norm.	Origin.	Norm.	Origin.	Value	Norm.	Origin.	%Improv	Norm.	Origin.	%Improv	Value	%Improv	
5	8	8	0.1299	240.32	0.2079	384.70	0.97445	0.0035	6.45	97.32	0.0052	9.66	97.49	0.99955	2.58	
		16	0.0763	141.25	0.1134	209.78	0.97501	0.0033	6.08	95.70	0.0041	7.64	96.36	0.99947	2.51	
		32	0.1031	190.76	0.1618	299.41	0.97426	0.0033	6.14	96.78	0.0042	7.81	97.39	0.99957	2.60	
	16	8	0.1448	268.01	0.2352	435.10	0.97415	0.0146	27.03	89.92	0.0243	44.87	89.69	0.99973	2.63	
		16	0.1385	256.23	0.2180	403.40	0.97461	0.0124	22.87	91.08	0.0206	38.08	90.56	0.99968	2.57	
		32	0.0944	174.59	0.1549	286.67	0.97365	0.0118	21.75	87.54	0.0199	36.78	87.17	0.99985	2.69	
	32	8	0.1083	200.38	0.1737	321.35	0.96946	0.0114	21.07	89.48	0.0183	33.83	89.47	0.99984	3.13	
		16	0.0536	99.14	0.0816	150.98	0.97461	0.0066	12.26	87.63	0.0106	19.69	86.96	0.99992	2.60	
		32	0.1204	222.80	0.1858	343.78	0.92290	0.0045	8.31	96.27	0.0071	13.08	96.20	0.99993	8.35	
	10	8	8	0.0438	81.03	0.0581	107.56	0.97584	0.0020	3.72	95.40	0.0028	5.27	95.10	0.99981	2.46
			16	0.0472	87.37	0.0627	115.95	0.97665	0.0018	3.37	96.14	0.0024	4.49	96.13	0.99969	2.36
			32	0.0437	80.87	0.0570	105.51	0.97555	0.0033	6.15	92.40	0.0045	8.27	92.17	0.99983	2.49
16		8	0.0412	76.23	0.0542	100.21	0.97592	0.0016	3.03	96.02	0.0022	4.10	95.91	0.99995	2.46	
		16	0.0382	70.69	0.0489	90.51	0.97504	0.0014	2.51	96.45	0.0020	3.78	95.82	0.99996	2.56	
		32	0.0401	74.17	0.0504	93.25	0.97525	0.0014	2.65	96.42	0.0019	3.52	96.22	0.99997	2.54	
32		8	0.0435	80.43	0.0564	104.28	0.96636	0.0108	19.92	75.23	0.0175	32.32	69.01	0.99987	3.47	
		16	0.0436	80.64	0.0559	103.50	0.97661	0.0049	9.06	88.77	0.0083	15.30	85.22	0.99997	2.39	
		32	0.0389	71.95	0.0501	92.78	0.97621	0.0043	7.97	88.93	0.0074	13.63	85.31	0.99997	2.43	

Table D.12: Results of long-term forecasting using EWT-SLSTM and SWT-SLSTM for S&P normalized and original test data

Window	Network		Stationary wavelet transform					Empirical wavelet transform								
	1st stage	2nd stage	MAE		RMSE		$R^2$	MAE			RMSE		$R^2$			
			Norm.	Origin.	Norm.	Origin.	Value	Norm.	Origin.	%Improv	Norm.	Origin.	%Improv	Value	%Improv	
5	8	8	0.1764	326.42	0.3037	562.01	0.84507	0.0629	116.32	64.36	0.0897	165.96	70.47	0.91717	8.53	
		16	0.1746	323.02	0.3032	560.95	0.87988	0.0625	115.69	64.19	0.0889	164.48	70.68	0.91727	4.25	
		32	0.1744	322.70	0.3033	561.10	0.87804	0.0620	114.79	64.43	0.0881	162.94	70.96	0.91746	4.49	
	16	8	0.1676	310.08	0.2933	542.77	0.86644	0.0650	120.34	61.19	0.0937	173.29	68.07	0.91609	5.73	
		16	0.1687	312.05	0.2972	549.88	0.86703	0.0648	119.86	61.59	0.0930	172.13	68.70	0.91719	5.79	
		32	0.1707	315.84	0.2987	552.72	0.84565	0.0646	119.55	62.15	0.0928	171.74	68.93	0.91759	8.51	
	32	8	0.1717	317.61	0.3014	557.65	0.86890	0.0678	125.36	60.53	0.0987	182.62	67.25	0.91565	5.38	
		16	0.1718	317.84	0.3005	555.99	0.86229	0.0665	123.03	61.29	0.0966	178.74	67.85	0.91623	6.25	
		32	0.1724	319.02	0.3028	560.33	0.87795	0.0664	122.93	61.47	0.0964	178.32	68.18	0.91766	4.52	
	10	8	8	0.1711	316.50	0.2945	544.96	0.83302	0.0665	123.00	61.14	0.0959	177.35	67.46	0.91575	9.93
			16	0.1705	315.39	0.2946	545.18	0.86787	0.0649	120.06	61.93	0.0931	172.22	68.41	0.91793	5.77
			32	0.1689	312.60	0.2948	545.39	0.83482	0.0646	119.56	61.75	0.0925	171.10	68.63	0.91776	9.93
16		8	0.1414	261.66	0.2395	443.10	0.86339	0.0657	121.49	53.57	0.0950	175.86	60.31	0.91863	6.40	
		16	0.1420	262.68	0.2406	445.23	0.85832	0.0654	120.96	53.95	0.0945	174.91	60.72	0.91978	7.16	
		32	0.1432	265.03	0.2428	449.34	0.85117	0.0650	120.28	54.62	0.0941	174.02	61.27	0.92030	8.12	
32		8	0.1714	317.12	0.3004	555.83	0.87440	0.0624	115.42	63.60	0.0894	165.38	70.25	0.92027	5.25	
		16	0.1698	314.15	0.2981	551.62	0.88114	0.0609	112.77	64.10	0.0868	160.55	70.89	0.92247	4.69	
		32	0.1701	314.77	0.2979	551.24	0.87725	0.0622	115.17	63.41	0.0886	163.84	70.28	0.92247	5.15	

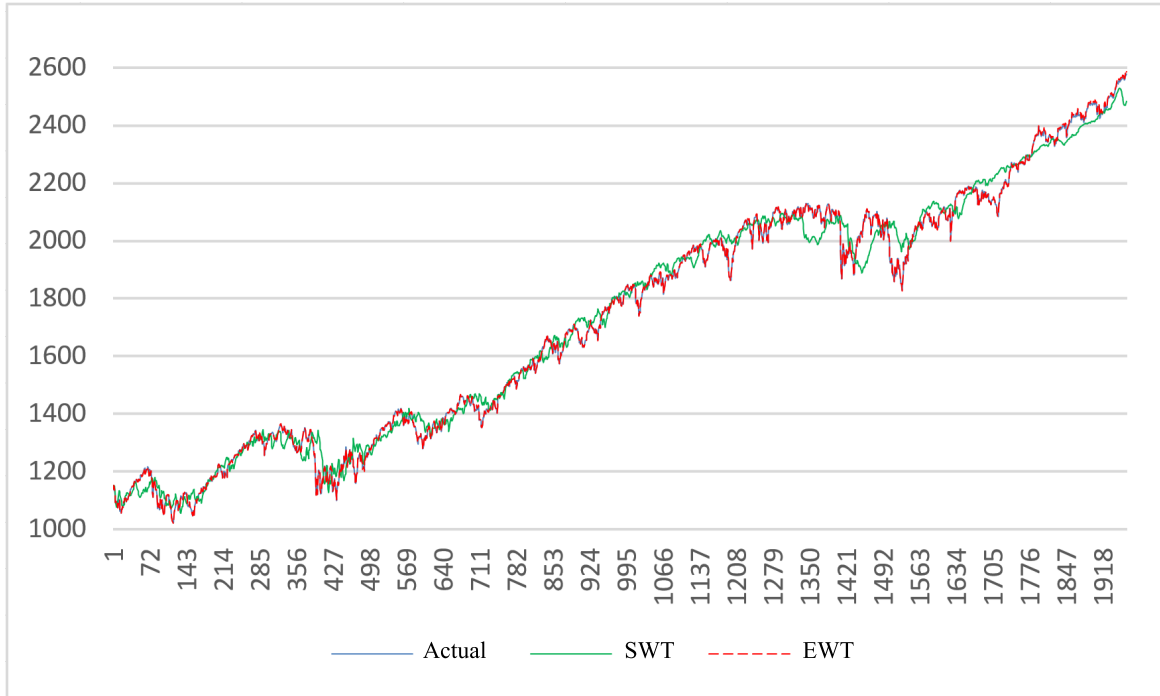


Figure D.8: S&P data curves of actual prices versus short-term forecasts using best selected models of EWT-SLSTM and SWT-SLSTM approaches

using ten time steps window size as input features to a network consisting of 32 neurons in the intermediate network and 16 neurons in the final network. The best achieved performance showed RMSE at 0.0868, MAE at 0.0609, and  $R^2$  at 0.92247. On the other hand, the best results of the SWT analysis method are attained using ten time steps features to a network architecture composed of 16 neurons in both intermediate and final LSTM networks. The produced results are 0.2395 for RMSE, 0.1414 for MAE, and 0.86339 for  $R^2$ . Comparing the results of the two approaches, we can notice the following:

- The results produced from the evaluation experiments conducted using Mackey-Glass time series and S&P stock index show that using empirical wavelet analysis attains superior performance compared to stationary wavelet analysis.

- Using an adaptive method to perform data decomposition and analysis helps in finding sparse representations of the data based on information included in the dataset.
- Stationary wavelet transform uses basis functions designed independently of the data representation which may result in insufficient multiresolution representation of the data. The wavelet coefficients include very large redundancy which increase computational complexity.
- The optimal number of resolution levels of the SWT is not easy to determine. On the other hand, EWT has the ability to design a basis wavelet function constructed based on the information involved in the data. The number of resolution levels depends on the Fourier spectrum of the data. The decomposition process is accomplished by segmenting the Fourier spectrum and applying filtering to separate each different mode.

## D.4 Deep Autoencoder and Technical Indicators

Performing data denoising and smoothing is a very important step in the preprocessing stage for many data science problems. In this section, two TIs are used as input features to the training network. Besides, a GRU autoencoder is adopted to extract smoother and denoised features from time series data. Generally, it consists of two GRU layers, the first one is the encoder and the other one is the decoder. The input to the GRU autoencoder is represented by sequence of vectors.

The autoencoder is a feature learning neural network designed to extract better

representation of data features. It is based on learning a function that is approximated to the identity function and aims at generating a set of new features  $\hat{x}$  that approximately matches original features  $x$  using backpropagation training algorithm, by setting the target value same as input.

$$h = f(x) \tag{D.2}$$

$$\hat{x} = f'(h) \tag{D.3}$$

$$\hat{x} \approx x \tag{D.4}$$

The objective of the training algorithm is to minimize the difference between input and target features. The learned features are subject to some constraints placed on the network by limiting the number of hidden states to extract extra patterns and constructs from original data. Original input can be reconstructed from noisy, compressed, or corrupted data [134].

Deep autoencoders are normally constructed using deep architectures, such as LSTM or GRU networks. Feature learning involves two phase operations; the first phase (encoding), performs data transformation into a smaller or larger dimension space, while the second phase (decoding), retransforms the data back to its approximated original representation. The compressed representation of the data in the intermediate layers defines the latent space representation. Deep autoencoders are commonly employed for image and signal compression and denoising. Particularly, for input features that exhibit random noisy patterns in its construct, deep autoencoders can be learned to extract noise-free meaningful data by subtracting the noise



and controlling the latent space [135].

Sequential data can be compressed and denoised using recurrent autoencoder model. The input data is encoded into a fixed-length vector using one or more layers of encoder and then reconstructed into its approximated original sequential form by one or more decoding layers. Decoder structure can be trained by controlling the size of the latent space to produce new transformed features that possess new meaningful properties contributing to fulfilling model development objectives. Relaxing the placed conditions on the decoder reconstruction training process could result in smoother and noise-free features which produce more efficient forecasting models.

After learning all the input sequences, an estimation of the target sequence is then generated. Partial reconstruction of input features helps improve the forecasting performance by subtracting noise and randomness from data. Relaxing autoencoder hidden layers conditions produces new data features characterized by additional properties and better data representation. The autoencoder network structure used by this work to extract denoised features is illustrated in Figure D.9. After data denoising and smoothing, we use a BGRU network to perform the forecast learning process.

The number of time steps used to form data is ten, fourteen, and twenty-four working days. Two technical indicator variables are added to the data to enhance and support feature smoothing and denoising. The employed variables are the simple and exponential moving averages discussed in appendix C. The technical indicator variables are computed based on two input sizes; the first one consists of  $n$  time steps, while the other input size consists of  $2n$  time steps where  $n$  represents size of slide window used to reshape the data into supervised learning. The process of

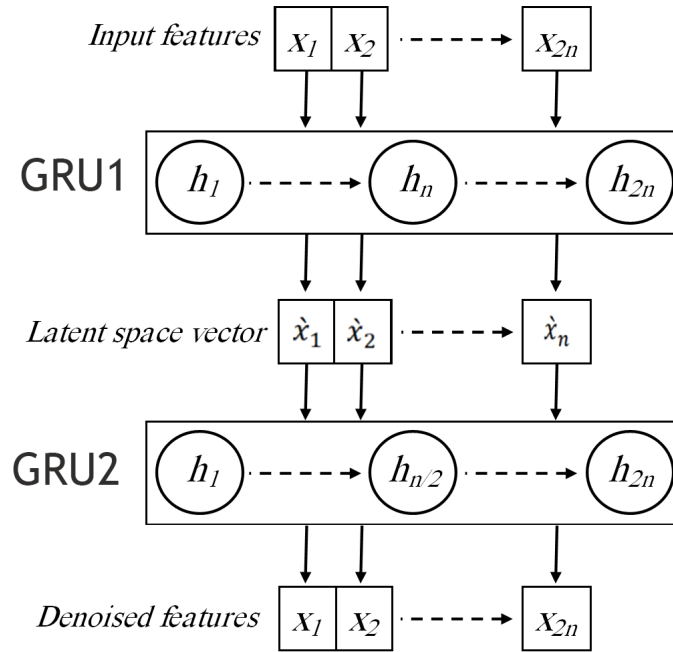


Figure D.9: Methodology used to train the GRU autoencoder

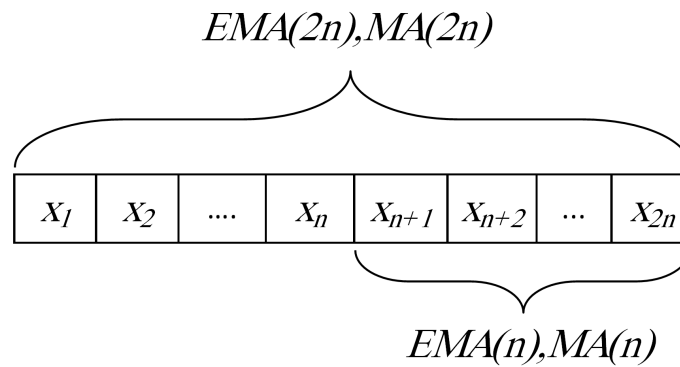


Figure D.10: Defining simple moving and exponential moving averages

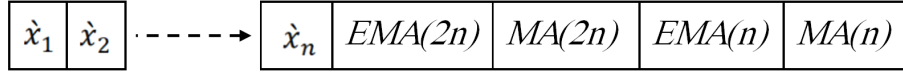


Figure D.11: Input data composed of denoised features and moving averages

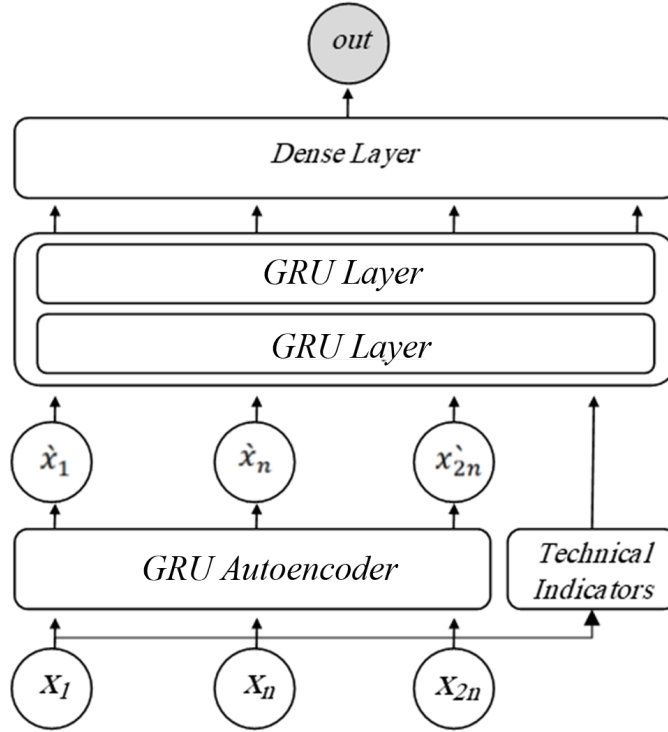


Figure D.12: Methodology used to train the model

defining the two TIs is illustrated in Figure D.10. The final input features used to train the network is composed of the denoised features and two TIs. Figure D.11 illustrates the input features structure. The methodology followed to construct the network architecture is illustrated in Figure D.12. Different structures of GRU layers developed by changing the number of memory cells (neurons) that determines the dimensionality of the output space.

To evaluate the methodology proposed in this section, we use the data of S&P and DJIA stock market indices data to perform the forecasting. The closing price

data was reshaped into a group of sequences, each consists of 10, 14, and 24 time steps to forecast the closing price one day ahead and twenty-two working days ahead (one month) in the future. All samples sequences are fed into the GRU autoencoder. Figure D.14 shows original and denoised data curves to express how new features align with data. Four input features is generated by computing the moving average and exponential moving average as illustrated in Figure D.11. Figure D.13 shows curves of the original data with the corresponding four generated technical features. The combination of the denoised features and technical indicator variables represents the input data to the deep BGRU.

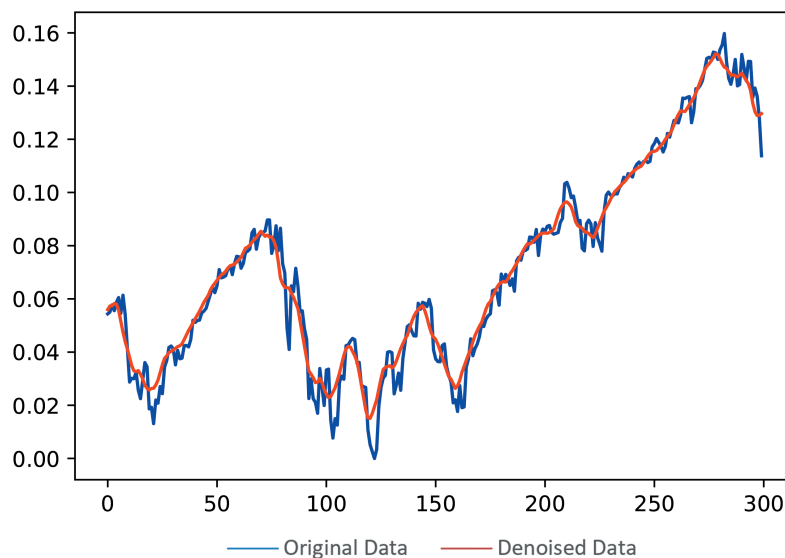


Figure D.13: The curves of the original and denoised data for part of the data

The averages results of the short and long-term forecast are shown in Table D.13. Results grouped by number of time lags (time steps). Best selected models based on performance on test data, classified by number of time steps are shown in Table D.14.

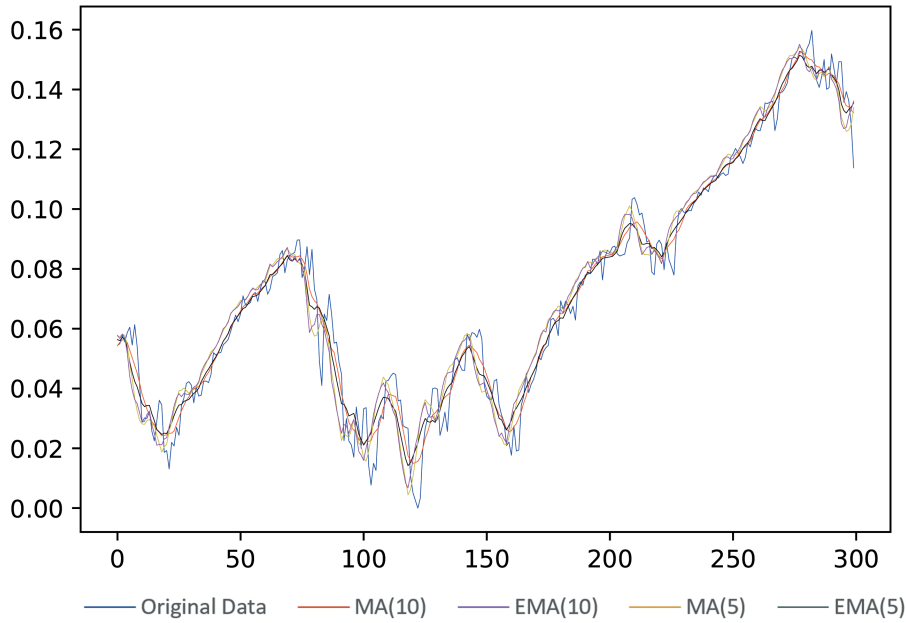


Figure D.14: Curves of part of the original data with corresponding generated technical features

Based on results, we may conclude the following:

- Comparing averages and minimum results in Tables D.13 and D.14, we notice that using window size of ten days produces the highest performance.
- It is clear that using larger window size for defining TIs may not increase the forecasting accuracy.
- The performance of the best models of both short and long-term time scales does not surpass that of the random walk model shown in Table 5.6. Consequently, the performance produced by BGRU model proposed in section 5.2 produces slightly higher accuracy. Therefore, using more TIs may improve the forecast accuracy.

Table D.13: Performance averages for both short- and long-term forecast of S&P and DJIA datasets

Dataset	Time scale	Time steps	RMSE	MAE	MAPE
DJIA	1	10	365.036	265.879	1.1363
		14	575.864	443.043	1.8823
		24	3310.135	2801.638	11.9529
	22	10	2731.921	2440.811	10.5153
		14	3377.711	2974.319	12.7082
		24	3588.998	3191.744	13.6700
S&P	1	10	19.453	12.816	0.5071
		14	23.351	15.079	0.5950
		24	23.211	16.518	0.6535
	22	10	196.378	179.503	6.9852
		14	182.298	164.968	6.4161
		24	251.568	231.503	8.9839

Table D.14: Best performance based on the RMSE results using test data for both short- and long-term forecast of S&P and DJIA datasets

Dataset	Time scale	Time steps	Neurons	BGRU Epochs	Autoencoder Epochs	RMSE	MAE	MAPE	$R^2$
DJIA	1	10	48	400	200	194.802	128.000	0.5616	0.9916
		14	16	400	100	211.764	138.447	0.6065	0.9900
		24	48	800	400	214.043	144.682	0.6367	0.9896
	22	10	32	400	800	1154.292	1019.787	4.5224	0.6541
		14	16	400	200	1682.222	1520.050	6.5785	0.2611
		24	48	400	800	1384.410	1234.506	5.4925	0.4939
S&P	1	10	48	400	100	19.453	12.815	0.5071	0.9891
		14	16	800	800	21.774	15.366	0.6101	0.9863
		24	32	400	800	23.211	16.518	0.6534	0.9843
	22	10	16	400	800	118.635	103.675	4.0555	0.5223
		14	32	800	800	91.486	77.084	3.0240	0.7142
		24	16	800	200	154.608	141.088	5.5009	0.1739

## APPENDIX E

# THESIS PUBLICATIONS

- Published Work:
  - K. A. Althelaya, E. S. M. El-Alfy, and S. Mohammed, “Evaluation of bidirectional lstm for short-and long-term stock market prediction,” in *Proceedings of the IEEE 9<sup>th</sup> International Conference on Information and Communication Systems (ICICS)*,2018, pp. 151–156.
  - K. A. Althelaya, E. S. M. El-Alfy, and S. Mohammed, “Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU),” in *Proceedings of the IEEE 21<sup>st</sup> Saudi Computer Society National Computer Conference*,2018.
- Under Review:
  - K. A. Althelaya, E. S. M. El-Alfy, and S. Mohammed, “Evaluation of bidirectional LSTM for short-and long-term stock market prediction,” in *Concurrency and Computation: Practice and Experience*,2018.
  - K. A. Althelaya, E. S. M. El-Alfy, and S. Mohammed, “Forecasting of Bahrain

Stock Market with Deep Learning: A Case Study,” in *Proceedings of the IEEE 8<sup>th</sup> International Conference on Modeling, Simulation and Applied Optimization*,2019.

- Under Preparation:
  - K. A. Althelaya, E. S. M. El-Alfy, and S. Mohammed, “Combining Multiresolution Analysis and Technical Indicators for Stock Market Forecasting Using Deep Learning,” ,2019.



# REFERENCES

- [1] Y.-J. Chen, Y.-M. Chen, S.-T. Tsao, and S.-F. Hsieh, “A novel technical analysis-based method for stock market forecasting,” *Soft Computing*, vol. 22, no. 4, pp. 1295–1312, 2018.
- [2] D. K. Kılıç and Ö. Uğur, “Multiresolution analysis of S&P500 time series,” *Annals of Operations Research*, vol. 260, no. 1-2, pp. 197–216, 2018.
- [3] P. Li, C. Jing, T. Liang, M. Liu, Z. Chen, and L. Guo, “Autoregressive moving average modeling in the financial sector,” in *Proceedings of the IEEE 2015 2<sup>nd</sup> International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*. IEEE, 2015, pp. 68–71.
- [4] G. Zhang, X. Zhang, and H. Feng, “Forecasting financial time series using a methodology based on autoregressive integrated moving average and Taylor expansion,” *Expert Systems*, vol. 33, no. 5, pp. 501–516, 2016.
- [5] M. Bildirici and Ö. Ersin, “Nonlinearity, volatility and fractional integration in daily oil prices: Smooth Transition Autoregressive ST-FI (AP) GARCH models,” *Romanian Journal of Economic Forecasting*, vol. 3, pp. 108–135, 2014.

- [6] M. Khashei, M. Bijari, and G. A. R. Ardali, “Hybridization of autoregressive integrated moving average (ARIMA) with probabilistic neural networks (PNNs),” *Computers & Industrial Engineering*, vol. 63, no. 1, pp. 37–45, 2012.
- [7] A. A. P. Santos and L. D. S. Coelho, “Neural networks, fuzzy system, and linear models in forecasting exchange rates: comparison and case studies,” in *Proceedings of the 2006 International Joint Conference on Neural Networks (IJCNN’06)*. IEEE, 2006, pp. 3094–3099.
- [8] E. Chong, C. Han, and F. C. Park, “Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies,” *Expert Systems with Applications*, vol. 83, pp. 187–205, 2017.
- [9] F. Shen, J. Chao, and J. Zhao, “Forecasting exchange rate using deep belief networks and conjugate gradient method,” *Neurocomputing*, vol. 167, pp. 243–253, 2015.
- [10] L. Di Persio and O. Honchar, “Artificial neural networks architectures for stock price prediction: Comparisons and applications,” *International Journal of Circuits, Systems and Signal Processing*, vol. 10, pp. 403–413, 2016.
- [11] R. Singh and S. Srivastava, “Stock prediction using deep learning,” *Multimedia Tools and Applications*, vol. 76, no. 18, pp. 18 569–18 584, 2017.
- [12] J. Li, H. Bu, and J. Wu, “Sentiment-aware stock market prediction: A deep learning method,” in *Proceedings of the IEEE 2017 International Conference on Service Systems and Service Management (ICSSSM)*. IEEE, 2017, pp. 1–6.

- [13] B.-L. Zhang, R. Coggins, M. A. Jabri, D. Dersch, and B. Flower, “Multiresolution forecasting for futures trading using wavelet decompositions,” *IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 765–775, 2001.
- [14] W. Bao, J. Yue, and Y. Rao, “A deep learning framework for financial time series using stacked autoencoders and long-short term memory,” *PloS ONE*, vol. 12, no. 7, 2017.
- [15] B. G. Malkiel and E. F. Fama, “Efficient capital markets: A review of theory and empirical work,” *The Journal of Finance*, vol. 25, no. 2, pp. 383–417, 1970.
- [16] K. Hamid, M. T. Suleman, S. Z. A. Shah, and R. S. I. Akash, “Testing the weak form of efficient market hypothesis: Empirical evidence from Asia-Pacific markets,” *International Research Journal of Finance and Economics*, no. 58, 2010.
- [17] S. Basu, “Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis,” *The Journal of Finance*, vol. 32, no. 3, pp. 663–682, 1977.
- [18] B. G. Malkiel, “The efficient market hypothesis and its critics,” *Journal of Economic Perspectives*, vol. 17, no. 1, pp. 59–82, 2003.
- [19] W. Ferson, “Market efficiency and forecasting,” in *Forecasting Expected Returns in the Financial Markets*, S. Satchell, Ed. Academic Press, Elsevier, 2007, pp. 1–15.

- [20] A. Timmermann and C. W. Granger, “Efficient market hypothesis and forecasting,” *International Journal of Forecasting*, vol. 20, no. 1, pp. 15–27, 2004.
- [21] D. Gholamiangonabadi, S. D. M. Taheri, A. Mohammadi, and M. B. Menhaj, “Investigating the performance of technical indicators in electrical industry in Tehran’s stock exchange using hybrid methods of SRA, PCA and neural networks,” in *Proceedings of the IEEE 5<sup>th</sup> Conference on Thermal Power Plants (CTPP)*, 2014, pp. 75–82.
- [22] A. Altunkaynak and T. A. Nigussie, “Monthly water consumption prediction using season algorithm and wavelet transform-based models,” *Journal of Water Resources Planning and Management*, vol. 143, no. 6, pp. 04 017 011–1 – 04 017 011–10, 2017.
- [23] H. Wang, G. Wang, G. Li, J. Peng, and Y. Liu, “Deep belief network based deterministic and probabilistic wind speed forecasting approach,” *Applied Energy*, vol. 182, pp. 80–93, 2016.
- [24] P. Liang, H.-D. Yang, W.-S. Chen, S.-Y. Xiao, and Z.-Z. Lan, “Transfer learning for aluminium extrusion electricity consumption anomaly detection via deep neural networks,” *International Journal of Computer Integrated Manufacturing*, vol. 31, no. 4-5, pp. 396–405, 2018.
- [25] I. Kaastra and M. Boyd, “Designing a neural network for forecasting financial and economic time series,” *Neurocomputing*, vol. 10, no. 3, pp. 215–236, 1996.
- [26] A. Lendasse, E. de Bodt, V. Wertz, and M. Verleysen, “Non-linear financial

- time series forecasting-application to the Bel 20 stock market index,” *European Journal of Economic and Social Systems*, vol. 14, no. 1, pp. 81–91, 2000.
- [27] N. Chen-xu and W. Jie-sheng, “Auto regressive moving average (ARMA) prediction method of bank cash flow time series,” in *Proceedings of the IEEE 2015 34<sup>th</sup> Chinese Control Conference (CCC)*. IEEE, 2015, pp. 4928–4933.
- [28] M. Kim, “Cost-sensitive estimation of ARMA models for financial asset return data,” *Mathematical Problems in Engineering*, vol. 2015, 2015.
- [29] D. Marcek, “Forecasting high frequency data: An ARMA-soft RBF network model for time series,” in *Applied Mechanics and Materials*, vol. 596. Trans Tech Publ, 2014, pp. 160–163.
- [30] J.-F. Chen, W.-M. Wang, and C.-M. Huang, “Analysis of an adaptive time-series autoregressive moving-average (ARMA) model for short-term load forecasting,” *Electric Power Systems Research*, vol. 34, no. 3, pp. 187–196, 1995.
- [31] D.-m. Xue and Z.-q. Hua, “ARIMA based time series forecasting model,” *Recent Advances in Electrical & Electronic Engineering (Formerly Recent Patents on Electrical & Electronic Engineering)*, vol. 9, no. 2, pp. 93–98, 2016.
- [32] D. Banerjee, “Forecasting of indian stock market using time-series ARIMA model,” in *Proceedings of the IEEE 2014 2<sup>nd</sup> International Conference on Business and Information Management (ICBIM)*. IEEE, 2014, pp. 131–135.
- [33] X. Wang and Y. Liu, “ARIMA time series application to employment forecast-

- ing,” in *Proceedings of the IEEE 2009 4<sup>th</sup> International Conference on Computer Science & Education (ICCSE'09)*. IEEE, 2009, pp. 1124–1127.
- [34] P. Newbold, “ARIMA model building and the time series analysis approach to forecasting,” *Journal of Forecasting*, vol. 2, no. 1, pp. 23–35, 1983.
- [35] D. Fay, J. Ringwood, M. Condon, and M. Kelly, “A comparison of linear and neural parallel time series models for short-term load forecasting in the Republic of Ireland,” in *Proceedings of the 3rd European IFS Workshop*. Shaker Verlag, 2000.
- [36] B. D. Baker and C. E. Richards, “A comparison of conventional linear regression methods and neural networks for forecasting educational spending,” *Economics of Education Review*, vol. 18, no. 4, pp. 405–415, 1999.
- [37] S. K. Sharma and S. Ghosh, “Short-term wind speed forecasting: Application of linear and non-linear time series models,” *International Journal of Green Energy*, vol. 13, no. 14, pp. 1490–1500, 2016.
- [38] H.-T. Pao and Y.-Y. Chih, “Comparison of linear and nonlinear models for panel data forecasting: Debt policy in Taiwan,” *Review of Pacific Basin Financial Markets and Policies*, vol. 8, no. 03, pp. 525–541, 2005.
- [39] B. Xiao-Ming and W. Cheng-Zhang, “Adaboost artificial neural network for stock market predicting,” *DEStech Transactions on Computer Science and Engineering*, no. aice-ncs, 2016.

- [40] Z. Guo, H. Wang, J. Yang, and D. J. Miller, “A stock market forecasting model combining two-directional two-dimensional principal component analysis and radial basis function neural network,” *PloS ONE*, vol. 10, no. 4, p. e0122385, 2015.
- [41] X. Zhong and D. Enke, “Forecasting daily stock market return using dimensionality reduction,” *Expert Systems with Applications*, vol. 67, pp. 126–139, 2017.
- [42] M. R. Hassan and B. Nath, “Stock market forecasting using hidden markov model: a new approach,” in *Proceedings of the IEEE 2005 5<sup>th</sup> International Conference on Intelligent Systems Design and Applications (ISDA ’05)*. IEEE, 2005, pp. 192–196.
- [43] W. Huang, Y. Nakamori, and S.-Y. Wang, “Forecasting stock market movement direction with support vector machine,” *Computers & Operations Research*, vol. 32, no. 10, pp. 2513–2522, 2005.
- [44] B. Majhi and C. Anish, “Multiobjective optimization based adaptive models with fuzzy decision making for stock market forecasting,” *Neurocomputing*, vol. 167, pp. 502–511, 2015.
- [45] W. Liu, T. Chen, and M. Y. Lee, “EPSO-GHSOM stock selecting and trading strategy on big data,” *Moon*, p. 7, 2007.
- [46] J. L. Ticknor, “A bayesian regularized artificial neural network for stock market

- forecasting,” *Expert Systems with Applications*, vol. 40, no. 14, pp. 5501–5506, 2013.
- [47] Y.-J. Chen, Y.-M. Chen, and C. L. Lu, “Enhancement of stock market forecasting using an improved fundamental analysis-based approach,” *Soft Computing*, vol. 21, no. 13, pp. 3735–3757, 2017.
- [48] A. Tsantekidis, N. Passalis, A. Tefas, J. Kannianen, M. Gabbouj, and A. Iosifidis, “Forecasting stock prices from the limit order book using convolutional neural networks,” in *Proceedings of the 2017 IEEE 19<sup>th</sup> Conference on Business Informatics (CBI)*, vol. 1. IEEE, 2017, pp. 7–12.
- [49] J.-F. Chen, W.-L. Chen, C.-P. Huang, S.-H. Huang, and A.-P. Chen, “Financial time-series data analysis using deep convolutional neural networks,” in *Proceedings of the 2016 IEEE 7<sup>th</sup> International Conference on Cloud Computing and Big Data (CCBD)*. IEEE, 2016, pp. 87–92.
- [50] A. Grossmann and J. Morlet, “Decomposition of Hardy functions into square integrable wavelet of constant shape,” *SIAM Journal of Mathematical Analysis*, vol. 15 (4), pp. 723–736, 1984.
- [51] S. G. Mallat, “A theory for multiresolution signal decomposition: the wavelet representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.
- [52] M. T. Ismail and A. A. A. Dghais, “Multiresolution analysis of bursa malaysia



- klci time series,” in *Proceedings of the American Institute of Physics Conference Series*, vol. 1847, no. 2. AIP Publishing, 2017, p. 020020.
- [53] S. Bekiros and M. Marcellino, “The multiscale causal dynamics of foreign exchange markets,” *Journal of International Money and Finance*, vol. 33, pp. 282–305, 2013.
- [54] H. Gunduz, Y. Yaslan, and Z. Cataltepe, “Intraday prediction of borsa istanbul using convolutional neural networks and feature correlations,” *Knowledge-Based Systems*, vol. 137, pp. 138–148, 2017.
- [55] M. R. Vargas, B. S. de Lima, and A. G. Evsukoff, “Deep learning for stock market prediction from financial news articles,” in *Proceedings of the IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, 2017, pp. 60–65.
- [56] B. Krollner, B. Vanstone, and G. Finnie, “Financial time series forecasting with machine learning techniques: A survey,” *Paper presented at the European Symposium on Artificial Neural Networks: Computational and Machine Learning, Bruges, Belgium*, 2010.
- [57] S. S. Hasan, R. Rahman, N. Mannan, H. Khan, J. N. Moni, and R. M. Rahman, “Improved stock price prediction by integrating data mining algorithms and technical indicators: A case study on Dhaka stock exchange,” in *Proceedings of the International Conference on Computational Collective Intelligence Technologies and Applications*. Springer, 2017, pp. 288–297.

- [58] F. B. Oriani and G. P. Coelho, "Evaluating the impact of technical indicators on stock forecasting," in *Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI)*, 2016, pp. 1–8.
- [59] Y. Shynkevich, T. McGinnity, S. A. Coleman, A. Belatreche, and Y. Li, "Forecasting price movements using technical indicators: Investigating the impact of varying input window length," *Neurocomputing*, vol. 264, pp. 71–88, 2017.
- [60] Y. L. Yong, D. C. Ngo, and Y. Lee, "Technical indicators for forex forecasting: A preliminary study," in *Proceedings of the International Conference in Swarm Intelligence*. Springer, 2015, pp. 87–97.
- [61] J. Chen, "SVM application of financial time series forecasting using empirical technical indicators," in *Proceedings of the IEEE International Conference on Information Networking and Automation (ICINA)*, vol. 1, 2010, pp. V1–77.
- [62] S. Rajab and V. Sharma, "An interpretable neuro-fuzzy approach to stock price forecasting," *Soft Computing*, pp. 1–16, 2017.
- [63] B. Weng, L. Lu, X. Wang, F. M. Megahed, and W. Martinez, "Predicting short-term stock prices using ensemble methods and online data sources," *Expert Systems with Applications*, vol. 112, pp. 258–273, 2018.
- [64] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach," *Applied Soft Computing*, vol. 70, pp. 525–538, 2018.

- [65] B. Weng, M. A. Ahmed, and F. M. Megahed, "Stock market one-day ahead movement prediction using disparate data sources," *Expert Systems with Applications*, vol. 79, pp. 153–163, 2017.
- [66] A. M. de Silva, R. I. A. Davis, S. A. Pasha, and P. H. W. Leong, "Forecasting Financial Time Series with Grammar-Guided Feature Generation," *Computational Intelligence*, vol. 33, no. 2, pp. 241–261, 2017.
- [67] R. H. Gálvez and A. Gravano, "Assessing the usefulness of online message board mining in automatic stock prediction systems," *Journal of Computational Science*, vol. 19, pp. 43–56, 2017.
- [68] J. Wang, R. Hou, C. Wang, and L. Shen, "Improved  $v$ -Support vector regression model based on variable selection and brain storm optimization for stock price forecasting," *Applied Soft Computing*, vol. 49, pp. 164–178, 2016.
- [69] C.-H. Su and C.-H. Cheng, "A hybrid fuzzy time series model based on ANFIS and integrated nonlinear feature selection method for forecasting stock," *Neurocomputing*, vol. 205, pp. 264–273, 2016.
- [70] Y.-S. Chen, C.-H. Cheng, C.-L. Chiu, and S.-T. Huang, "A study of ANFIS-based multi-factor time series models for forecasting stock index," *Applied Intelligence*, vol. 45, no. 2, pp. 277–292, 2016.
- [71] T.-L. Chen and F.-Y. Chen, "An intelligent pattern recognition model for supporting investment decisions in stock market," *Information Sciences*, vol. 346–347, pp. 261–274, 2016.

- [72] M. Thenmozhi and G. Sarath Chand, “Forecasting stock returns based on information transmission across global markets using support vector machines,” *Neural Computing and Applications*, vol. 27, no. 4, pp. 805–824, 2016.
- [73] D. Kumar, S. S. Meghwani, and M. Thakur, “Proximal support vector machine based hybrid prediction models for trend forecasting in financial markets,” *Journal of Computational Science*, vol. 17-part 1, pp. 1–13, 2016.
- [74] P.-C. Chang, J.-L. Wu, and J.-J. Lin, “A Takagi–Sugeno fuzzy model combined with a support vector regression for stock trading forecasting,” *Applied Soft Computing*, vol. 38, pp. 831–842, 2016.
- [75] M. Göçken, M. Özçalıcı, A. Boru, and A. T. Dosdoğru, “Integrating metaheuristics and Artificial Neural Networks for improved stock price prediction,” *Expert Systems with Applications*, vol. 44, pp. 320–331, 2016.
- [76] M. Maragoudakis and D. Serpanos, “Exploiting Financial News and Social Media Opinions for Stock Market Analysis using MCMC Bayesian Inference,” *Computational Economics*, vol. 47, no. 4, pp. 589–622, 2016.
- [77] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, “Predicting stock market index using fusion of machine learning techniques,” *Expert Systems with Applications*, vol. 42, no. 4, pp. 2162–2172, 2015.
- [78] C.-C. Chen, Y.-C. Kuo, C.-H. Huang, and A.-P. Chen, “Applying market profile theory to forecast Taiwan Index Futures market,” *Expert Systems with Applications*, vol. 41, no. 10, pp. 4617–4624, 2014.

- [79] R. Bisoi and P. Dash, “A hybrid evolutionary dynamic neural network for stock market trend analysis and prediction using unscented Kalman filter,” *Applied Soft Computing*, vol. 19, pp. 41–56, 2014.
- [80] Y.-S. Chen, C.-H. Cheng, and W.-L. Tsai, “Modeling fitting-function-based fuzzy time series patterns for evolving stock index forecasting,” *Applied Intelligence*, vol. 41, no. 2, pp. 327–347, 2014.
- [81] C. L. Dunis, R. Rosillo, D. de la Fuente, and R. Pino, “Forecasting IBEX-35 moves using support vector machines,” *Neural Computing and Applications*, vol. 23, no. 1, pp. 229–236, 2013.
- [82] L.-Y. Wei, T.-L. Chen, and T.-H. Ho, “A hybrid model based on adaptive-network-based fuzzy inference system to forecast Taiwan stock market,” *Expert Systems with Applications*, vol. 38, no. 11, pp. 13 625–13 631, 2011.
- [83] A. Rodríguez-González, Á. García-Crespo, R. Colomo-Palacios, F. Guldrís Iglesias, and J. M. Gómez-Berbís, “CAST: Using neural networks to improve trading systems based on technical analysis by means of the RSI financial indicator,” *Expert Systems with Applications*, vol. 38, no. 9, pp. 11 489–11 500, 2011.
- [84] Y. Kara, M. Acar Boyacioglu, and Ö. K. Baykan, “Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange,” *Expert Systems with Applications*, vol. 38, no. 5, pp. 5311–5319, 2011.
- [85] C.-H. Cheng, T.-L. Chen, and L.-Y. Wei, “A hybrid model based on rough sets

- theory and genetic algorithms for stock price forecasting,” *Information Sciences*, vol. 180, no. 9, pp. 1610–1629, 2010.
- [86] R. Majhi, G. Panda, and G. Sahoo, “Development and performance evaluation of FLANN based model for forecasting of stock markets,” *Expert Systems with Applications*, vol. 36, no. 3, pp. 6800–6808, 2009.
- [87] S. Mallat, *A wavelet tour of signal processing*. Academic press, 1999.
- [88] P.-H. Chiang, S. P. V. Chiluvuri, S. Dey, and T. Q. Nguyen, “Forecasting of solar photovoltaic system power generation using wavelet decomposition and bias-compensated random forest,” in *Proceedings of the IEEE 9<sup>th</sup> Annual Green Technologies Conference (GreenTech)*, 2017, pp. 260–266.
- [89] A. Aussem and F. Murtagh, “Combining neural network forecasts on wavelet-transformed time series,” *Connection Science*, vol. 9, no. 1, pp. 113–122, 1997.
- [90] A. R. Reis and A. A. Da Silva, “Feature extraction via multiresolution analysis for short-term load forecasting,” *IEEE Transactions on Power Systems*, vol. 20, no. 1, pp. 189–198, 2005.
- [91] J. Gilles, “Empirical wavelet transform,” *IEEE Transactions on Signal Processing*, vol. 61, no. 16, pp. 3999–4010, 2013.
- [92] P. Keerthy, P. Maya, and M. Sindhu, “An adaptive transient tracking harmonic detection method for power quality improvement,” in *Proceedings of the IEEE Region 10 Symposium (TENSymp) Conference*, 2017, pp. 1–6.

- [93] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015.
- [94] A. Graves, A. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6645–6649.
- [95] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [96] A. Graves and J. Schmidhuber, “Framewise phoneme classification with bidirectional LSTM and other neural network architectures,” *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.
- [97] R. Zhao, D. Wang, R. Yan, K. Mao, F. Shen, and J. Wang, “Machine health monitoring using local feature-based gated recurrent unit networks,” *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1539–1548, 2018.
- [98] P. Hillion and M. Suominen, “The manipulation of closing prices,” *Journal of Financial Markets*, vol. 7, no. 4, pp. 351–375, 2004.
- [99] T. Y. Hatiboglu, “Investigating the Predictability of Financial Time Series Through Bayesian Variable Selection Methods,” Master’s thesis, Vienna University of Economics and Business, Austria, 2016.
- [100] F. E. Tay and L. Cao, “Modified support vector machines in financial time series forecasting,” *Neurocomputing*, vol. 48, no. 1-4, pp. 847–861, 2002.

- [101] R. Majhi, G. Panda, G. Sahoo, P. K. Dash, and D. P. Das, “Stock market prediction of s&p 500 and djia using bacterial foraging optimization technique,” in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2007)*, 2007, pp. 2569–2575.
- [102] R. Palivonaite, K. Lukoseviciute, and M. Ragulskis, “Short-term time series algebraic forecasting with mixed smoothing,” *Neurocomputing*, vol. 171, pp. 854–865, 2016.
- [103] A. K. Rout, R. Bisoi, and P. Dash, “A low complexity evolutionary computation-ally efficient recurrent functional link neural network for time series forecasting,” in *Proceedings of the IEEE Power Communication and Information Technology Conference (PCITC)*, 2015, pp. 576–582.
- [104] A. K. Rout, P. Dash, R. Dash, and R. Bisoi, “Forecasting financial time series using a low complexity recurrent neural network and evolutionary learning approach,” *Journal of King Saud University-Computer and Information Sciences*, vol. 29, no. 4, 2015.
- [105] R. Adhikari and R. Agrawal, “A combination of artificial neural network and random walk models for financial time series forecasting,” *Neural Computing and Applications*, vol. 24, no. 6, pp. 1441–1449, 2014.
- [106] C. Croarkin, P. Tobias, C. Zey *et al.*, *Engineering statistics handbook*. NIST iTL, 2002.



- [107] H. Lütkepohl and F. Xu, “The role of the log transformation in forecasting economic variables,” *Empirical Economics*, vol. 42, no. 3, pp. 619–638, 2012.
- [108] C. R. Nelson and C. R. Plosser, “Trends and random walks in macroeconomic time series: some evidence and implications,” *Journal of Monetary Economics*, vol. 10, no. 2, pp. 139–162, 1982.
- [109] S. N. Durlauf and P. C. Phillips, “Trends versus random walks in time series analysis,” *Econometrica: Journal of the Econometric Society*, pp. 1333–1354, 1988.
- [110] A. Wald and J. Wolfowitz, “On a test whether two samples are from the same population,” *The Annals of Mathematical Statistics*, vol. 11, no. 2, pp. 147–162, 1940.
- [111] A. Wald and J. Wolfowitz, “An exact test for randomness in the non-parametric case based on serial correlation,” *The Annals of Mathematical Statistics*, vol. 14, no. 4, pp. 378–388, 1943.
- [112] Y.-T. Chen, “On the robustness of Ljung-Box and McLeod-Li Q tests: a simulation study,” *Economics Bulletin*, vol. 3, no. 17, pp. 1–10, 2002.
- [113] M. A. Arranz, “Portmanteau test statistics in time series,” *Time Orientated Language*, pp. 1–8, 2005.
- [114] G. E. Box and D. A. Pierce, “Distribution of residual autocorrelations in autoregressive-integrated moving average time series models,” *Journal of the American Statistical Association*, vol. 65, no. 332, pp. 1509–1526, 1970.

- [115] C.-M. KUAN, “Lecture on time series diagnostic tests,” *Institute of Economics Academia*, 2008.
- [116] G. M. Ljung and G. E. Box, “On a measure of lack of fit in time series models,” *Biometrika*, vol. 65, no. 2, pp. 297–303, 1978.
- [117] M. Duvinage, P. Mazza, and M. Petitjean, “The intra-day performance of market timing strategies and trading systems based on Japanese candlesticks,” *Quantitative Finance*, vol. 13, no. 7, pp. 1059–1070, 2013.
- [118] R. F. de Brito and A. L. Oliveira, “A foreign exchange market trading system by combining GHSOM and SVR,” in *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, 2012, pp. 1–7.
- [119] J. Orrantia, “DMForex: a data mining application to predict currency exchange rates and trends,” Ph.D. dissertation, California State University, Northridge, 2012.
- [120] X. Li, L. Yang, F. Xue, and H. Zhou, “Time series prediction of stock price using deep belief networks with intrinsic plasticity,” in *Proceedings of IEEE 29<sup>th</sup> Chinese Control And Decision Conference (CCDC)*, 2017, pp. 1237–1242.
- [121] R. Majhi, G. Panda, and B. Majhi, “Robust prediction of stock indices using PSO based adaptive linear combiner,” in *Proceedings of IEEE World Congress on Nature & Biologically Inspired Computing, (NaBIC)*, 2009, pp. 312–317.
- [122] M. Khashei and Z. Hajirahimi, “Performance evaluation of series and parallel

- strategies for financial time series forecasting,” *Financial Innovation*, vol. 3, no. 1, p. 24, 2017.
- [123] J. Wang and J. Wang, “Forecasting stock market indexes using principle component analysis and stochastic time effective neural networks,” *Neurocomputing*, vol. 156, pp. 68–78, 2015.
- [124] Z. Guo, H. Wang, Q. Liu, and J. Yang, “A feature fusion based forecasting model for financial time series,” *PloS ONE*, vol. 9, no. 6, p. e101113, 2014.
- [125] M.-W. Hsu, S. Lessmann, M.-C. Sung, T. Ma, and J. E. Johnson, “Bridging the divide in financial market forecasting: Machine learners vs. financial economists,” *Expert Systems with Applications*, vol. 61, pp. 215–234, 2016.
- [126] D. A. Lesmond, M. J. Schill, and C. Zhou, “The illusory nature of momentum profits,” *Journal of Financial Economics*, vol. 71, no. 2, pp. 349–380, 2004.
- [127] P. D. Yoo, M. H. Kim, and T. Jan, “Financial forecasting: Advanced machine learning techniques in stock market analysis,” in *Proceedings of the IEEE 9<sup>th</sup> International Multitopic Conference (INMIC)*, 2005, pp. 1–7.
- [128] E. F. Fama, “Market efficiency, long-term returns, and behavioral finance,” *Journal of Financial Economics*, vol. 49, no. 3, pp. 283–306, 1998.
- [129] D. A. Hsieh, “Chaos and nonlinear dynamics: Application to financial markets,” *The Journal of Finance*, vol. 46, no. 5, pp. 1839–1877, 1991.

- [130] S.-H. Poon and C. W. Granger, “Forecasting volatility in financial markets: A review,” *Journal of Economic Literature*, vol. 41, no. 2, pp. 478–539, 2003.
- [131] M. L. Mitchell and E. Stafford, “Managerial decisions and long-term stock price performance,” *The Journal of Business*, vol. 73, no. 3, pp. 287–329, 2000.
- [132] E. F. Fama, “Efficient capital markets: II,” *The Journal of Finance*, vol. 46, no. 5, pp. 1575–1617, 1991.
- [133] L. Glass and M. Mackey, “Mackey-glass equation,” *Scholarpedia*, vol. 5, no. 3, p. 6908, 2010.
- [134] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” *Journal of machine learning research*, vol. 11, no. Dec, pp. 3371–3408, 2010.
- [135] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, “A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional lstm neural networks,” in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 1996–2000.

# VITAE

- Name: Khaled Ahmed Lutf Al-Thelaya
- Nationality: Yemeni
- Date of Birth: 01/01/1984
- Email: *khaled.althelaya@hotmail.com*
- Permenant Address: Thamar, Yemen
- BSc in Computer Science from Thamar University, Yemen (September 2003 - July 2007).
- Research Interests:
  - Machine Learning Applications and Methodologies.
  - Deep Learning Techniques and Applications.
  - Time Series Analysis and Forecasting.
  - Computer vision and image processing.
  - Digital Signal processing.

- Academic Experience:
  - Teaching assistant, Thamar University, Yemen (October 2008 - August 2015).
  
- Conferences Attended:
  - Information and Communication Systems (ICICS), 2018 9<sup>th</sup> International Conference Apr 03-05, 2018.
  - The 21<sup>st</sup> National Computer Conference (NCC'2018) April 25-26, 2018.
  
- Presentations Delivered:
  - Evaluation of bidirectional LSTM for short-and long-term stock market prediction, paper presented at the Information and Communication Systems (ICICS), 2018 9<sup>th</sup> International Conference.
  - Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU), paper presented at the 21<sup>st</sup> National Computer Conference (NCC'2018).
  
- Professional Training :
  - A training program to develop the skills of faculty members sponsored by Thamar university authority lasted for one week.
  
- Skills :
  - Knowledge of research methodologies
  - Data and information collection
  - Academic writing and presentation skills
  - communication skills