2022

Time Series and Machine Learning Models for Financial Markets Forecast

Siqi Mao

FLORIDA STATE UNIVERSITY

COLLEGE OF ARTS AND SCIENCES

TIME SERIES AND MACHINE LEARNING MODELS FOR FINANCIAL MARKETS FORECAST

By

SIQI MAO

A Dissertation submitted to the
Department of Statistics
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

2022

Siqi Mao defended this dissertation on March 24, 2022.

The members of the supervisory committee were:

Xufeng Niu
Professor Directing Dissertation

Giray Ökten
University Representative

Adrian Barbu
Committee Member

Fred Huffer
Committee Member

Wei Wu
Committee Member

The Graduate School has verified and approved the above-named committee members, and certifies that the dissertation has been approved in accordance with university requirements.

This dissertation is dedicated to my family.

ACKNOWLEDGMENTS

The days of pursuing a doctorate are invaluable to me. There are so many amazingly wonderful people, and I am so grateful to have them in my life.

Firstly, I would like to express my sincere gratitude to my advisor Dr. Xufeng Niu for his inspirational ideas and patient guidance and for helping me become self-dependent in academic life. It is impossible for me to finish this dissertation without his continuous support.

My appreciation also goes to my other committee members: Dr. Adrian Barbu, Dr. Fred Huffer, Dr. Giray Ökten, and Dr. Wei Wu, for their constructive comments and insightful suggestions. Additionally, I am deeply grateful to all the faculty and staff in our statistics department.

Furthermore, I would like to thank all my friends, especially Dr. Yinpu Li and Shutong Ge. The informal academic discussions with Yinpu helped me understand some statistical concepts comprehensively. In addition, Shutong helped me fit in the new environment when I came to the United States the first time.

Finally, I would like to express my most profound appreciation to my parents for their unconditional love and support throughout my entire life. In addition, I would like to thank my wife, Yixin Kang, who is the light of my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

The following short list of symbols are used throughout the document. The symbols represent quantities that I tried to use consistently.

$P_{i,t}$	The price of asset i at time index t
$r_{i,t}$	The natural logarithm return of asset i at time index t
$D_{i,t}$	The direction of movement of the natural logarithm return of asset i at time index t
B	The backward shift operator
L_t	The linear portion in an ARIMA-LSTM model
N_t	The nonlinear portion in an ARIMA-LSTM model
ϵ_t	The error term in an ARIMA-LSTM model
$\boldsymbol{L_t}$	The linear portion in an VRIMA-LSTM model
$\boldsymbol{N_t}$	The nonlinear portion in an VRIMA-LSTM model
$\boldsymbol{\epsilon_t}$	The error term in an VRIMA-LSTM model
$\boldsymbol{i}_t$	The input gate (activation) vector in a LSTM unit at time index t
$\boldsymbol{f}_t$	The forget gate (activation) vector in a LSTM unit at time index t
$\boldsymbol{g}_t$	The cell input activation vector in a LSTM unit at time index t
$\boldsymbol{o}_t$	The output gate (activation) vector in a LSTM unit at time index t
$\boldsymbol{c}_t$	The cell state vector in a LSTM unit at time index t
$\boldsymbol{h}_t$	The hidden state vector in a LSTM unit at time index t
$\boldsymbol{S}_{i,t}$	The set of considered sentences for the asset i at time index t
$\boldsymbol{F_{S_{i,t}}}$	The polarities of $\boldsymbol{S}_{i,t}$ which are calculated by a BERT model.
$\boldsymbol{B_i}$	The financial sentiment index matrix for asset i
$\boldsymbol{A_i}$	The analyst recommendation index matrix for asset i

ABSTRACT

The prediction of financial time series is an essential topic in quantitative investment. In this dissertation, we proposed two types of new models. They are bidirectional encoder representations from Transformers-based financial sentiment index enhanced (BERTFSIE) models and analysts' recommendation index enhanced (ARIE) models. Compared to statistical time series models, BERTFSIE models and ARIE models involve not only the assets log-returns but also the financial sentiment of investors' reactions and analyst recommendations individually. Furthermore, an adaptive epoch size selection strategy, an early stopping for sequential data strategy, and a novel hyperparameters selection strategy (Score voting) are proposed to solve the over-fitting problem in the training process of a neural network model when its size is large, and the training data is limited.

An experimental study is conducted examining the BERTFSIE models and ARIE models. Additionally, they are compared with statistical time series models (e.g., Autoregressive Integrated Moving Average (ARIMA) model, Autoregressive Integrated Moving Average with Explanatory Variable (ARIMAX) model, and the Vector Autoregressive Moving Average (VARMA)), machine learning models (e.g., Long short-term memory (LSTM)), and hybrid models (e.g., ARIMA-LSTM, VARMA-LSTM) in terms of predicting the log-returns and direction movements of two stocks from four industries (Technology: Alphabet Inc., Verizon Communications Inc.; Finance: Citigroup Inc., The Goldman Sachs Group, Inc.; Pharmaceutical: Bristol-Myers Squibb Company, AbbVie Inc.; Retail: Costco Wholesale Corporation, Target Corporation) with three testing proportion: 10%, 20%, 30%. To be more specific, the forecastability of different models is evaluated in two separate ways: the prediction ability of the assets log-returns is assessed by root mean square error (RMSE) and mean absolute error (MAE), and the prediction ability in the direction of the assets log-returns are judged by weighted F_1 score and weighed AUC score. Meanwhile, one step ahead forecasting and rolling forward with expanding windows are selected in the experimental study.

The results of the experimental study indicate that BERTFSIE models perform almost uniformly better than the other models in terms of weighted F_1 score and weighed AUC score. In addition, with the increase of the validation size and testing size, the performance difference among these models is diminishing. At the same time, ARIE models have competitive performance with BERTFSIE models in some selected industries.

CHAPTER 1

INTRODUCTION AND MOTIVATION

Financial time series forecasting is challenging due to the noise and high dimensionality of financial market data. We observed only a limited part of a complex system in a financial market. It is most likely that there is not enough information available to understand the structure of a time series and predict its future values (e.g., Längkvist et al. 2014a).

Researchers are accustomed to handling the financial time series data by assuming the data had some properties. For example, the assumptions of statistical time series processes often include linearity and normality. Some famous models are widely applied in financial time series analysis with these assumptions. For example, Autoregressive Integrated Moving Average (ARIMA) models (uni-variate), Vector Autoregressive Moving-Average (VARMA) processes (multi-variate), and their many variations are often applied to financial time series. Furthermore, they are used to predict the asset prices or logarithmic returns of a firm and guide the process of investment decision-making and strategies (e.g., Huang and Hsu 2020).

In recent years, with the increase of computational power and the rapid development of novel techniques (e.g., graphics processing unit (GPU) accelerated computing, distributed computing, and parallel computing), more machine learning algorithms and approaches are employed in financial data analysis (e.g., Henrique et al. 2019). The most popular machine learning approaches are Multi-Layer Perceptron (MLP) networks and Recurrent Neural Networks (RNNs) in sequential data analysis. Specifically, Long Short-Term Memory (LSTM) networks are a particular type of RNN which was developed by Hochreiter and Schmidhuber (1997) to handle the gradient vanishing problem in sequential data prediction that occurs in training RNN networks with gradient-based learning methods. Additionally, LSTM is widely implemented in financial assets price or logarithmic returns prediction (e.g., Bao et al. 2017; Pawar et al. 2018).

Based on some comparisons between statistical time series models and machine learning methods, the LSTM models were indicated to be superior to the ARIMA models in predicting some monthly financial market indices adjusted closed prices (e.g., Siami-Namini and Namin 2018). However, the ARIMA method was stated to outperform the LSTM in the daily Bitcoin trading price forecasting (e.g., Yamak et al. 2019). This contradiction showed that each method has its pros and cons. To take advantage of both statistical models and machine learning methods, hybrid models (e.g., ARIMA-LSTM) were introduced and concluded to give better results than the single models (e.g., Choi 2018b; Temür et al. 2019).

The historical asset prices can only provide limited information when forecasting a company's stock price or log return. Daily market news and investors' reactions to some events often affect future asset prices. But, unlike the historical asset prices, the information and the investors' reactions are usually expressed and spread in unstructured data formats (e.g., text, photos, audio, video). Among these data formats, conveying information in the text is a typical one. To uncover the evidence of sentiment within the news and the investors' reactions expressed in text, the researchers used to employ the dictionary-based approach (also known as the Lexicon-based approach) (e.g., Yang et al. 2015; Park and Kim 2016). Moreover, financial sentiments could be analyzed by the text classification of the influential novel natural language processing (NLP) techniques (e.g., Embeddings from language models (ELMo) (e.g., Peters et al. 2018), Bidirectional encoder representations from Transformers (BERT) (e.g., Devlin et al. 2018)).

This dissertation is motivated by the following four main research questions. The first one is whether machine learning methods yield better results in financial market prediction than the statistical time series models. The second one is whether the hybrid models of statistical time series models and machine learning approaches show their superiority when compared with the single models in the forecast of the financial assets logarithmic returns and the direction of the assets logarithmic returns. Compared to analysts' recommendations, the third question is whether the NLP techniques in financial sentiment analysis (financial sentiments based on individual investors' altitude) reveal beneficial signals in predicting the financial market's upward or downward movements. The last question is whether the market indices are helpful regarding financial market assets prices prediction when it is compared to the models that do not involve the indices.

In this dissertation, the following strategies are applied to answer the four main research questions:

1. Statistical time series models (e.g., ARIMA, Autoregressive Integrated Moving Average with Explanatory Variable (ARIMAX), VARMA) and machine learning models (e.g., LSTM) will be investigated and compared in the forecast performance of some financial assets' logarithmic returns and their direction logarithmic returns.

2. Hybrid models of well-known statistical time series models and machine learning methods will be reviewed and compared to single models in financial assets' logarithmic returns and the direction of their logarithmic returns.

3. The effects of individuals' sentiment on the forecast accuracy of financial assets logarithmic returns and the direction of the assets logarithmic returns will be evaluated by using NLP techniques (e.g., Bidirectional Encoder Representations from Transformers (BERT)). In addition, the financial sentiments, which are generated by BERT, are considered as external information input to the models which have an LSTM component. Moreover, the effects of financial sentiments (based on individual investors) are compared with: (1) the original models and (2) the models that involved the analysts' suggestions (Analyst recommendation index enhanced (ARIE) models).

4. Multivariate models are considered to take market indices as covariates. Additionally, the performance of these market indices involving models is compared with the performance of their original models.

The rest of this dissertation is organized as follows:

- Chapter 2 is a literature review that discusses the research work in the development and application of statistical time series models, machine learning models, and hybrid models in financial time series prediction. After that, the evolution of applying natural language processing (NLP) techniques in financial sentiment analysis is described. Finally, because neural network models are widely employed in this dissertation, the development of some well-known optimization methods is described.

- Chapter 3 discusses the methodology of univariate and multivariate statistical time series models, including ARIMA, ARIMAX, and VARMA. After that, this chapter discusses mainly one machine learning method: LSTM. Furthermore, the structure of some hybrid models containing ARIMA-LSTM and VARMA-LSTM is reviewed. In addition, Bidirectional Encoder Representation from Transformers (BERT) is discussed.

  There are two new types of models proposed. They are BERT-based Financial Sentiment Index Enhanced (BERTFSIE) models and Analyst Recommendation Index Enhanced (ARIE) models. In the last part of this chapter, some popular iterative methods for optimizing neural network models are discussed.

- Chapter 4 provides an empirical analysis. The methods described in Chapter 3 are compared in terms of the prediction performance of the stock logarithmic returns and the movement direction of two selected stocks from the each four industries (Technology: Alphabet Inc., Verizon Communications Inc.; Finance: Citigroup Inc., The Goldman Sachs Group, Inc.; Pharmaceutical: Bristol-Myers Squibb Company, AbbVie Inc.; Retail: Costco Wholesale Corporation, Target Corporation), and the companies' details are shown in Table 4.1.

  For each industry, there are three testing and validation proportions (10%, 20%, 30%). At the end of this chapter, there are twelve tables provided to summarize the models' performance in forecasting the two stocks' price log-return and the movement direction in the four industries.

- Chapter 5 includes the conclusions are made based on the results shown in Chapter 4. To be more specific, conclusions are made by the individual four industries (Technology, finance, pharmaceutical, and retail industries). Furthermore, some existing challenges are stated (e.g., the neural network method's performance is sensitive to the predefined hyperparameters). Finally, there are some aspects that could be improved in this dissertation, and we provide a tentative plan to strengthen the current methods as future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Statistical Models in Financial Time Series Forecast

2.1.1 Univariate Cases

For univariate weakly stationary time series models, the embryonic of autoregressive (AR) models were first-mentioned in an application of analyzing sunspot periodicity (e.g., Yule 1927), and it started the modern age of time series analysis (e.g., Makridakis 1976). After that, the concept of AR models was enlarged from lower-order cases to general cases (e.g., Walker 1931). Several years later, the scheme of moving-average (MA) models, which were called "moving summation" at that time, were proposed by Slutzky (1937). Furthermore, autoregressive moving average (ARMA) models were presented to model the linear weakly stationary time series data, and autoregressive integrated moving average (ARIMA) models were developed to handle the linear non-stationary time series data, which can be processed to a weakly stationary time series after finite times forward difference. If the seasonal pattern existed in data, seasonal ARMA (SARMA) or seasonal ARIMA (SARIMA) models could be employed.

ARMA-type models were popularized because of the standard iterative stages modeling approach called the "Box–Jenkins method" (e.g., Box and Jenkins 1970). With the help of the standard modeling procedure, ARMA-type models were widely applied to do short-time forecasting in various sub-fields (e.g., real-estate prices, inflation, the stock exchanged indices, gold price) in finance (e.g., Tse 1997; Meyler et al. 1998; Kasap 1998; Guha and Bandyopadhyay 2016).

2.1.2 Multivariate Cases

Due to the deficient information provided by univariate financial time series, more than one time series would be considered sometimes. There is some empirical evidence showing the existence of a negative relationship between inflation and stock returns (e.g., James et al. 1985). When the external input time series are also quantitative and equally spaced, Autoregressive Integrated Moving Average with Explanatory Variable (ARIMAX) models, which are one variation of the ARIMA-type models, could be applied. In ARIMAX models, the letter "X" refers to "Exogenous variables". ARIMAX models are also called "Regression with ARIMA errors" (e.g., Hyndman and

Athanasopoulos 2018). But in the structure of ARIMAX models, there are only past terms of the exogenous variables (disturbances). Suppose the past terms of original time series are in the model too, the ARIMAX models need to be extended to "Dynamic Regression Models" (Pankratz, 1991) (It is also called "Transfer Function Models" according to Box (2016)). Compared to ARIMA models, ARIMAX models were shown to bring better results in predicting Slovstats' gross domestic product (GDP) per capita (e.g., Peter and Silvia 2012).

In ARIMAX or transfer function models, the exogenous time series are assumed to be unaffected by the output time series. However, if the feedback effect from the output time series to the exogenous time series exists, ARIMAX or transfer function models might not be valid anymore. Vector-type models could be applied to handle the feedback effects. For example, Vector Autoregression (VAR) models were famous for figuring out the dynamic relations among several macroeconomic variables (e.g., Sims 1980). Furthermore, VAR models could be extended to the Vector Autoregressive Moving-Average (VARMA) models, which are also called Multivariate Autoregressive Moving-Average (MARMA) models. Compared to the estimation for univariate statistical time series models, VARMA models' estimation and model identification are much more complicated. For instance, VARMA models may not be uniquely identified when modeling a stationary vector time series. To solve this problem, a necessary and sufficient condition was provided by (e.g., Hannan 1969). Later, a method called extended cross-correlation matrices (ECCM) was proposed to select the order (p, q) of a VARMA model (e.g., Tiao and Tsay 1983). There are more general discussions of VARMA in, e.g., Elliott et al. (2006); Tsay and Tsay (2013); Box et al. (2015).

The multivariate time series models had a wide application and showed their superiority over single variate models in financial data analysis. For example, the VARMA models made more accurate predictions than the ARMA models in the forecast of agricultural commodity and oil prices (e.g., Gülerce and Ünal 2017). Additionally, the VARMA models were proved to outperform the VAR models in the macroeconomic variables forecast (e.g., Athanasopoulos and Vahid 2008).

2.2 Machine Learning Approaches in Financial Time Series Prediction

In the last decades, machine learning models have competed with the established statistical time series models in financial time series prediction. For supervised learning algorithms, there are Random Forest, eXtreme Gradient Boosting (XGBoost), Support Vector Regression (SVR), Artificial

Neural Networks (ANNs), Recurrent neural network (RNN: LSTM/ GRU) (e.g., Shah 2019). The unsupervised learning methods include RBM-based methods (e.g., Restricted Boltzmann Machines (RBM), conditional RBM (cRBM), Temporal RBM), K-means (e.g., Längkvist et al. 2014b). In the literature review of machine learning approaches for financial market prediction (e.g., Henrique et al. 2019), artificial neural networks (ANNs) models were the most popular methods applied in the reviewed references. Therefore, in the following part of this section, the methods of artificial neural networks (ANNs) are mainly discussed.

McCulloch and Pitts (1943) started a new era of research on artificial neural networks (ANNs) by mathematizing them into models. In the next few decades, the theories of artificial neural networks were formulated speedily (For example, Hebbian theory (e.g., Hebb 1949), Perceptron (e.g., Rosenblatt 1958)). The research on artificial neural networks stagnated after Minsky and Papert (1969) emphasized that computers did not have sufficient computational power to process an extensive neural network at that time. However, the rapid growth of the processing power of computers and the reinvention of back-propagation (e.g., Rumelhart et al. 1986) made artificial neural networks possible in reality within the following decades. Additionally, there are many types of artificial neural networks introduced, for instance, Multi-Layer Perceptron (MLP) networks, Recurrent Neural Networks (RNNs), Recursive Neural Network (a generalization of recurrent neural networks), Convolutional Neural Networks (CNNs), Graph Neural Network (GNNs). Among these neural network methods, Multi-Layer Perceptron (MLP) networks and Recurrent Neural Networks (RNNs) are generally applied to the financial field.

In the application of MLPs in finance, the modular neural networks (3 layers forward fully connected) were used to predict the selling and buying timings of the Tokyo Stock Exchange Prices Indexes (TOPIX) (Kimoto et al., 1990). The MLPs were proved to be superior to the Atman's Z score models in the classification of the financial health of some firms (Lacher et al., 1995), and similar results were performed in some bankruptcy events (Leshno and Spector, 1996). The genetic algorithm method to feature discretization for MLPs was shown to be statistically better (in the McNemar tests) than backpropagation in training MLPs to predict stock price indices (Kim and Han, 2000). The MLPs were stated to outperform the linear models (e.g., capital asset pricing model (CAPM), French's 3-factor model) in predicting 367 public corporations traded on the Shanghai Stock Exchange (SHSE) (Cao et al., 2005). According to West et al. (2005), the ensemble models of MLPs (e.g., Cross-Validation, Bagging, Adaptive Boosting) were shown the superiority over the single MLPs in the financial decision. The MLPs were found to be better than support vector machines in predicting the direction of stock price index movement in the Istanbul Stock Exchange (Kara et al., 2011).

The application of MLPs mainly focuses on non-sequential financial data. However, the data in financial markets usually have a temporal-dependent structure. Therefore, compared with MLPs, RNNs are more suitable for financial time series forecasting. A recurrent neural networks approach was applied to recognize the stock price patterns in the first section of the Tokyo Stock Exchange, and the accuracy in the test set achieved 93.8% (Kamijo and Tanigawa, 1990). After LSTMs were built to overcome the gradient vanishing problem, the vanilla RNNs were generally replaced in financial data forecasts. The LSTMs with dropout were shown to be the best in comparison with vanilla RNNs and gated recurrent units (GRUs were introduced by Cho et al. (2014)) in the close stock prices of Google (Di Persio and Honchar, 2017). Some similar results can be found in, e.g., Zhao et al. 2017; Bao et al. 2017; Pawar et al. 2018.

2.3 Hybrid Models of Time Series Approaches and Machine Learning Methods in Financial Time Series Forecast

The classic statistical models for financial time series have strict assumptions, simple structures, fully developed estimation approaches, and stable performance. Compared with the statistical time series models, machine learning methods have a flexible model structure and potentially better performance. Still, they are time-consuming in the training process and pretend to be more sensitive to the different hyper-parameters combinations. Additionally, machine learning models need a large sample in the training process (especially for the ANNs). Machine learning models would suffer from the over-fitting problem (A model has a high performance in the training set but has poor performance in testing data). However, in financial time series analysis, the assumptions of classic statistical models are barely satisfied, and the sample size is also limited. Therefore, a natural idea was proposed to deal with such a dilemma. The idea is that the statistical time series models and machine learning methods could be combined to uncover the financial market behavior (e.g., stock prices or logarithmic returns trend).

The early-stage application of hybrid models in the financial market could be found in the 1990s. According to Wang and Leu (1996), predicting the mid-term price trend in the Taiwan stock exchange weighted stock index (TSEWSI) was made by an ARIMA-ANN model. Still, the design of the ANN structure depended on the order of ARIMA, and this strategy is rarely persuadable. A new combination method for hybrid models was proposed to extract the linear pattern by ARIMA models and then extract non-linear patterns by ANNs. The final prediction would be the summation of the linear and non-linear parts. The hybrid models were shown to achieve a higher forecasting accuracy in 3 empirical data sets, including the exchange rate (Zhang, 2003b). Similar results were obtained by the hybrid models of ARIMA and SVM in Pai and Lin (2005). According to Aladag et al. (2009), a new hybrid method combining Elman's Recurrent Neural Networks (ERNN) and ARIMA models was proposed to achieve higher accuracy than the hybrid strategy that was proposed by Zhang (2003b).

Moreover, in Kumar and M. (2014), there were three hybrid models discussed (ARIMA-SVM, ARIMA-ANN, and ARIMA-Random Forest), and the individual models were compared to predict the CNX Nifty (also known as S&P CNX Nifty) index returns. In the paper, the ARIMA-SVM was proven to be the best under various performance metrics. According to Khairalla and Ning (2017), an adaptive weights hybrid model of ARIMA and SVM was applied to the Sudanese Pound/EURO exchange rate and showed superiority to the hybrid method, which was proposed by Zhang (2003b).

Furthermore, according to Choi (2018b), the hybrid models combining ARIMA and LSTM were applied to predict stock price correlation. It turned out to be superior to all the mentioned rest financial models (e.g., full historical model, constant correlation model, single-index model, and multi-group model). A comprehensive review of hybrid models in time series modeling and forecast could be found in, e.g., Hajirahimi and Khashei 2019.

2.4 Natural Language Processing (NLP) Techniques in Financial Sentiment Analysis

Researchers have found that the markets might not be fully efficient. Investors' irrational behavior would have a remarkable impact on the financial market movement and potential risks. In addition, this kind of behavior could be reflected by investors' sentiments (e.g., Barberis et al. 1998; Baker and Wurgler 2006; Tetlock 2007; Nopp and Hanbury 2015). In January 2021, the "GameStop short squeeze" event was a typical example to support that financial sentiment could significantly impact the financial market.

A statistical method to get investors' opinions is to send questionnaires. However, this is a time and money consuming way to collect information. Compared with sending questionnaires to get investors' views directly, the implied sentiment could also be found on the Internet indirectly because the Internet has become a necessary part of our daily life and the easiness of accessibility to the Internet. It became the majority way of predicting the polarities of investors' reactions by their conversations or discussion on the Internet (e.g., Twitter, blogs) instead of through the questionnaires (e.g., Severyn and Moschitti 2015; Yang et al. 2015).

Similar to classification standards of sequential data prediction models, the sentiment analysis approaches could also be classified as classical methods, machine learning approaches, and hybrid models (e.g., Ghiassi et al. 2013; Atzeni et al. 2017).

The lexicon-based (dictionary-based) approach is a widely applied classical method for financial sentiment analysis. The main idea of lexicon-based (dictionary-based) approaches is to obtain the sentiment score of the target context by matching it with the pre-defined semantically labeled dictionaries while extracting the polarity (e.g., positive, negative) and strength of each word or phase in the target context (e.g., Taboada et al.). The two main critical concerns in the lexicon-based approach are the qualification of the pre-defined dictionaries and the weight calculation methods for polarity (e.g., Kearney and Liu 2014). Compared with the machine learning approaches, the main weakness of the lexicon-based (dictionary-based) approaches are that the pre-defined dictionaries were independently collected from the target context (e.g., Liu 2012). In other words, the dictionaries might not contain all the words or misinterpret the polarity of some phrases in the target context. Another problem may be caused by an inappropriate weighting mechanism when calculating the polarity of the target context (e.g., Kearney and Liu 2014).

Many standard machine learning (non-neural network machine learning models) are applied in financial sentiment analysis, such as Naive Bayes Classification, Decision Tree, Support Vector Machine Regression, and Random Forest (e.g., Wang et al. 2015; Atzeni et al. 2017). These classic machine learning methods in financial sentiment analysis would still need the manually designed features as the input. Nevertheless, the deep learning approaches (e.g., Convolutional Neural Networks (CNNs), Tree-Structured LSTM) would take word embeddings as input and be free from the requirements of classic machine learning methods (e.g., Kim 2014; Tai et al. 2015; Singhal and Bhattacharyya 2016; Pozzi et al. 2016).

The release of Transformer (a sequence-to-sequence model) is a significant breakthrough in the history of NLP (e.g., Vaswani et al. 2017). The Bidirectional Encoder Representation from Transformers (BERT) was proposed in the following year (Devlin et al., 2018). In short, the structure of BERT is the encoder of the Transformer, and it is based on attention mechanism instead of bi-direction LSTM (e.g., Embeddings from Language Models (ELMo) proposed by Peters et al. (2018)) . The BERT applies to various usage, and a typical one is the sentiment analysis. For instance, according to Sousa et al. (2019), the BERT was applied to analyze sentiments of financial news (e.g., CNBC, Forbes, New York Times, and others), and it yielded the highest accuracy among the other methods (e.g., text CNN, TF-IDF SVM, bow-SVM). If the BERT is going to be implemented in different fields, it could be trained by the specific domain corpus, for example, the Financial BERT (Araci, 2019; Hiew et al., 2019). Furthermore, the textual-based sentiment indices were created by BERT, and they involved stock return prediction by VAR and LSTM (Hiew et al., 2019). A more comprehensive review of financial sentiment analysis and NLP in FinTech applications could be found in, e.g., Liu 2012; Man et al. 2019; Chen et al. 2020.

2.5 Optimizers for Neural Network Models

A standard process for training a machine learning model (typically for neural network models) is to minimize a loss function (which is also called “cost function”) by adjusting the unknown parameters (which are also called ’weights’) in a supervised learning problem (Bishop, 2006). Therefore, training a machine learning model could be stated as an unconstrained optimization problem. Many iterative algorithms are designed to solve such an optimization problem, for example, Newton–Raphson method, which is the most famous algorithm in optimization problems (Galántai, 2000). However, calculating the Hessian matrix is impracticable when the parameter space dimension is large. The Quasi-Newton algorithms were proposed to solve this problem; for example, Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm (e.g., Broyden 1970; Fletcher 1970; Goldfarb 1970; Shanno 1970).

Compared with other machine learning models, the gradient methods, instead of Quasi-Newton methods, are more famous for the optimization problem in neural network models because of the success in NLP techniques, e.g., Transformer (Vaswani et al., 2017). In the gradient methods, there are many algorithms were proposed, for example, the stochastic gradient descent (SGD; Robbins and Monro, 1951), the stochastic gradient descent with Momentum (SGDM; The original idea is from Polyak (1964), A detailed discussion can be found in Liu et al., 2020), Nesterov accelerated gradient (NAG; Nesterov, 1983), Adaptive Gradient Algorithm (ADAGRAD; Duchi et al., 2011), Root Mean Square Propagation (RMSProp; Tieleman et al., 2012), Adaptive Moment Estimation (ADAM; Kingma and Ba, 2014), AdaBelief (Kingma and Ba, 2014). A comprehensive comparison and discussion of different gradient descent methods could be found in, e.g., Choi et al. (2019).

CHAPTER 3

METHODOLOGY IN FINANCIAL TIME SERIES FORECAST

The primary materials covered in this dissertation are about financial time series. Therefore, in this chapter, some notations need to be defined. Besides, all the time series discussed in this dissertation are assumed to be evenly spaced time series.

The natural logarithmic return of the asset i at time index t is defined as:

$$r_{i,t} = log\left(\frac{P_{i,t}}{P_{i,t-1}}\right), \tag{3.1}$$

where $P_{i,t}$ is the price of the asset i at time index t (Tsay, 2010, Chapter 1).

The direction of the movement of the natural logarithmic return of the asset i at time index t is defined as:

$$D_{i,t} = \begin{cases} 1 & r_{i,t} > 0 \\ 0 & r_{i,t} = 0 \\ -1 & r_{i,t} < 0 \end{cases}, \tag{3.2}$$

where $r_{i,t}$ is defined in Equation 3.1.

In this dissertation, univariate predefined target logarithmic returns (noted as r_{target}) are focused on. To be more specific, for the models included in the dissertation, the evaluation metrics are calculated only for the target logarithmic returns to make the performance of different models (univariate and multivariate models) comparable. After defining necessary notations, the rest of the chapter discusses the popular models applied in financial time series forecasts.

3.1 Statistical Time Series Models in Financial Time Series Forecast

There are three popular statistical time series models described in this sub-chapter. They are ARIMA models, Linear Regression Models with ARIMA errors (ARIMAX), and Vector Autoregressive Moving-Average (VARMA) models. Among these three models, only the ARIMA models are considered univariate models, and the rest are multivariate.

3.1.1 Univariate Time Series Models

Autoregressive Integrated Moving Average (ARIMA). A time series $\{r_t | r_t \in \mathbb{R}, t \in \mathbb{Z}\}$ generated by Equation 3.3 is called an auto-regressive integrated moving-average model with AR order p, trend-difference order d, MA order q. It is denoted as $ARIMA(p, d, q)$.

$$\phi(B)(1-B)^d r_t = \phi_0 + \theta(B)a_t, \tag{3.3}$$

where

$$\phi(B) = 1 - \phi_1 B - \cdots - \phi_p B^p,$$

$$\theta(B) = 1 - \theta_1 B \cdots - \theta_q B^q,$$

ϕ_0 is a constant, a_t is a white-noise process with mean zero and variance σ, and B is the backward shift operator ($B^k r_t = r_{t-k}$) (Tsay, 2010, Chapter 2).

In the modeling procedures of ARIMA, the Box–Jenkins method is the most popular and significant one. In the list of predefined order combinations of p and q, the information criterion could select the appropriate model (e.g., Akaike information criterion (AIC; Akaike, 1974), Bayesian information criterion (BIC), or Schwarz information criterion (SBC; Schwarz, 1978), Hannan–Quinn information criterion (HQ; Hannan and Quinn, 1979)).

3.1.2 Multivariate Time Series Models

Due to the complexity of the stock market, single time series may not provide the necessary information for prediction, so multivariate time series should be considered.

If there are two or more time series, there are basically two situations:

- Only one time series is mainly concerned, and all the rest time series affect the concerned time series with no feedback, the **linear regression model with ARIMA errors** would be applicable.
- If there exists a feedback effect between the concerned time series and other time series, the **Vector Autoregressive Moving-Average (VARMA)** models are better.

The linear regression model with ARIMA errors and Vector Autoregressive Moving-Average are introduced in the following paragraphs.

A Linear Regression Model With ARIMA Errors. A linear regression model with ARIMA errors is also called Autoregressive Integrated Moving Average with Explanatory Variable (ARIMAX).

Let $\{r_{target,t} | r_{target,t} \in \mathbb{R}, t \in \mathbb{Z}\}$ be the target time series and $\{r_{1,t} | r_{1,t} \in \mathbb{R}, t \in \mathbb{Z}\}, \cdots, \{r_{k,t} | r_{k,t} \in \mathbb{R}, t \in \mathbb{Z}\}$ be the other external k time series. A linear regression model with ARIMA errors can be defined as:

$$r_{target,t} = \beta_0 + \sum_{i=0}^{p_1} \beta_{1,i} \cdot r_{1,t-i} + \cdots + \sum_{i=0}^{p_k} \beta_{k,i} \cdot r_{k,t-i} + \eta_t$$

and η_t follows an ARIMA(p, d, q) process,

$$(1 - \phi_1 B - \cdots - \phi_p B)(1 - B)^d \eta_t = (1 - \theta_1 B \cdots - \theta_q B)\epsilon_t,$$

where ϵ_t is a white noise series (Hyndman and Athanasopoulos, 2018, Chapter 9).

Vector Autoregressive Moving-Average (VARMA). A k-dimensional time series $\{\boldsymbol{R_t} | \boldsymbol{R_t} \in \mathbb{R}^k, t \in \mathbb{Z}\}$ is a vector auto-regression moving-Average process, denoted as VARMA(p, q), if

$$\phi(\boldsymbol{B})\boldsymbol{R_t} = \boldsymbol{\phi}_0 + \boldsymbol{\theta}(\boldsymbol{B})\boldsymbol{a_t},$$

where

$$\phi(\boldsymbol{B}) = \boldsymbol{I}_k - \phi_1 \boldsymbol{B^1} - \cdots - \phi_p \boldsymbol{B}^p,$$

$$\theta(\boldsymbol{B}) = \boldsymbol{I}_k - \theta_1 \boldsymbol{B^1} \cdots - \theta_q \boldsymbol{B^q},$$

$\boldsymbol{I}_k$ is a k by k identity matrix, $\boldsymbol{\phi}_0$ is a constant vector, $\{\boldsymbol{a_t} | \boldsymbol{a_t} \in \mathbb{R}^k, t \in \mathbb{Z}\}$ is a sequence of independent and identically distributed random vectors with mean zero and a positive-definite covariance matrix $\boldsymbol{\Sigma}_a$, $\boldsymbol{B}$ is the element-wise backward shift operator for a vector ($\boldsymbol{B}^k \boldsymbol{R_t} = \boldsymbol{R_{t-k}}$) (Tsay, 2014, Chapter 3).

3.2 Machine Learning Methods in Financial Time Series Forecast

The statistical time series methods are discussed in the previous section. Compared with the time series models, machine learning approaches are more flexible in assumptions. Only the Long Short-Term Memory (LSTM) is discussed in this section. The LSTM is a particular recurrent neural network, and it was introduced to overcome the gradient vanishing problem in the training process of vanilla RNNs (Hochreiter and Schmidhuber, 1997). An LSTM cell with forget gate is proposed by Gers et al. (2000) and it is applied and proved to have great results in Sak et al. (2014) and Kong et al. (2019).

3.2.1 Long Short-term Memory Cell With a Forget Gate (LSTM)

Let $\boldsymbol{X_t} = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_p)'$ be the input matrix, where $\boldsymbol{x}_i = (x_{i,1}, \cdots, x_{i,t})', 1 \leq i \leq p$. Additionally, let $\boldsymbol{Y_t} = (\boldsymbol{y}_1, \cdots, \boldsymbol{y}_q)'$ be the output matrix, where $\boldsymbol{y}_i = (y_{i,1}, \cdots, y_{i,t})', 1 \leq i \leq q$. A LSTM cell is defined as the following equations:

$$\begin{aligned}
\boldsymbol{i}_t &= \sigma\left(\boldsymbol{W_{input,X}}\boldsymbol{X_t} + \boldsymbol{W_{input,h}}\boldsymbol{h}_{t-1} + \boldsymbol{b_{input}}\right), \\
\boldsymbol{f}_t &= \sigma\left(\boldsymbol{W_{f,X}}\boldsymbol{X_t} + \boldsymbol{W_{f,h}}\boldsymbol{h}_{t-1} + \boldsymbol{b_f}\right), \\
\boldsymbol{g}_t &= \tanh\left(\boldsymbol{W_{g,X}}\boldsymbol{X_t} + \boldsymbol{W_{g,h}}\boldsymbol{h}_{t-1} + \boldsymbol{b_g}\right), \\
\boldsymbol{o}_t &= \sigma\left(\boldsymbol{W_{o,X}}\boldsymbol{X_t} + \boldsymbol{W_{o,h}}\boldsymbol{h}_{t-1} + \boldsymbol{b_o}\right), \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t, \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh\left(\boldsymbol{c}_t\right),
\end{aligned} \tag{3.4}$$

where $\boldsymbol{W}$ type notations are wight matrices, $\boldsymbol{b}$ terms stand for bias vectors, $\odot$ is the Hadamard production (element-wise multiplication), $\sigma(\cdot)$ and $\tanh(\cdot)$ denote element-wise sigmoid function and element-wise hyperbolic tangent function, respectively. Additionally, $\boldsymbol{i}_t$ is the input gate (activation) vector, $\boldsymbol{f}_t$ is the forget gate (activation) vector, $\boldsymbol{g}_t$ is the cell input (activation) vector, $\boldsymbol{o}_t$ is the output gate (activation) vector, $\boldsymbol{c}_t$ is the cell state vector, $\boldsymbol{h}_t$ is the hidden state vector. Further more, $\boldsymbol{h}_t$ is the approximate of $\boldsymbol{Y_t}$ ($\boldsymbol{h}_t$ is also the output of the LSTM block). The $\sigma(\cdot)$ and $\tanh(\cdot)$ are defined by the following equations:

$$\sigma(\cdot) = \frac{1}{1 + e^{-X}}, \quad \tanh(\cdot) = \frac{e^X - e^{-X}}{e^X + e^{-X}}.$$

The $\boldsymbol{W}$ type wight matrices and $\boldsymbol{b}$ terms bias vectors can be learned in a problem with the specific input matrix, output matrix, and the predefined LSTM structure. Figure 3.1 can show the structure of a long short-term memory cell with a forget gate.

There is an advantage in applying a standard LSTM cell with a forget gate in the financial time series. The number of output features can differ from the number of input features. For example, tomorrow's stock closed price of Apple Inc. could be predicted by today's closed prices of the other three companies. However, the direct application of a standard LSTM cell has a problem too. It can only predict the exact time steps as the input series. Using the previous example, we can not use the past week's closed price from the other three companies to predict tomorrow's stock closed price of Apple Inc. by directly applying the LSTM block. To solve this problem, we need to add a few layers after the LSTM cell in order to reshape the output. For example, the additional layers could be linear layers (e.g., Sak et al. 2014).

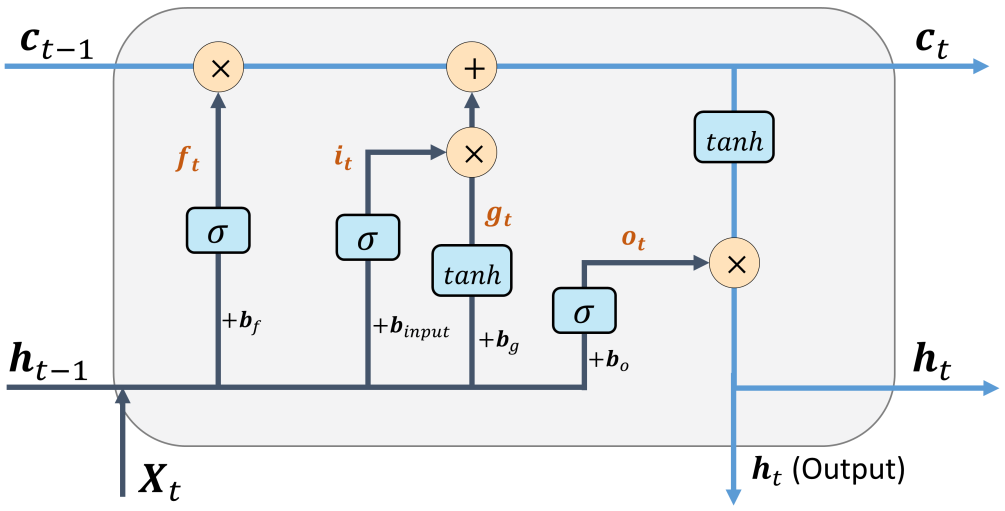


Figure 3.1: Long Short-term Memory With a Forget Gate[1].

As we mentioned at the beginning of this chapter, univariate target logarithmic returns r_{target} are focused. Additionally, to compare the performance of the introduced univariate and multivariate models, an additional linear layer should be employed in LSTM to make it focus on r_{target}. The linear layer is defined in Equation 3.5.

$$\begin{aligned} \boldsymbol{h}_t &= LSTM(\boldsymbol{X_t}), \\ \hat{r}_{target,t+1} &= \boldsymbol{W} \cdot vec(\boldsymbol{h}_t) + b, \end{aligned} \tag{3.5}$$

Sometimes, the single LSTM cell is not enough, and in this case, the stacked LSTM models can be applied. The stacked LSTM models assign the necessary numbers of LSTM blocks sequentially after the current LSTM cell, and the following LSTM cell takes the previous LSTM cell output as the input. For stacked LSTM cells, it can be illustrated by Figure 3.2.

The steps of the modeling process for LSTM are as follows:

- **Step 1:** Define the hyperparameters in LSTM: the size of the training window, the size of the prediction window, the epochs, the batch size, the number of hidden layers, the number of neurons in each layer.

[1]The plot is edited from https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

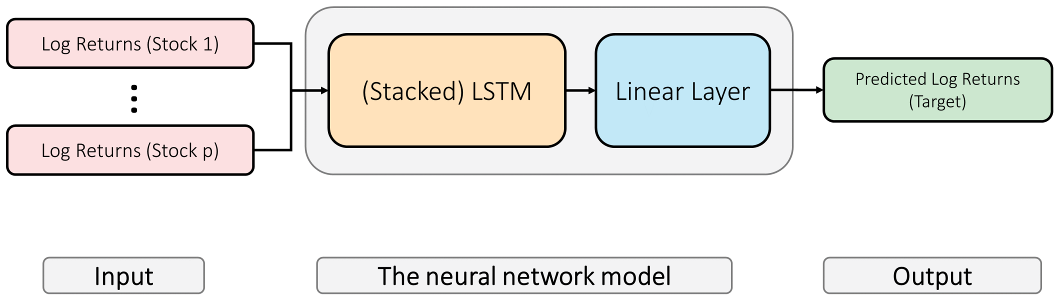


Figure 3.2: The Structure of the Edited Stacked LSTM Model.

- **Step 2:** Define the loss function and the optimizer (the gradient descent method).
- **Step 3:** Train the LSTM model by finding the values of the $\boldsymbol{W}$ type wight matrices and the $\boldsymbol{b}$ terms bias vectors that could make the loss function get a local minimum value.
- **Step 4:** Evaluate the trained LSTM models by cross-validation in terms of the predefined evaluation metrics and adjust the model structure if necessary.
- **Step 5:** Apply the well-defined LSTM models to do prediction.

3.3 Hybrid Models in Financial Time Series Forecast

In this section, the idea of hybrid models is restated, and two hybrid models (ARIMA-LSTM, VARMA-LSTM) are discussed.

3.3.1 ARIMA-LSTM

ARIMA-LSTM. A univariate time series $\{r_t | r_t \in \mathbb{R}, t \in \mathbb{Z}\}$ is assumed be consisted of three parts: a linear auto-correlation structure, a nonlinear part, and an error term (Zhang, 2003a). That is,

$$r_t = L_t + N_t + \epsilon_t, \tag{3.6}$$

where $\{L_t|L_t \in \mathbb{R}, t \in \mathbb{Z}\}$ is the linear portion, $\{N_t|N_t \in \mathbb{R}, t \in \mathbb{Z}\}$ is the nonlinear portion and $\{\epsilon_t|\epsilon_t \in \mathbb{R}, t \in \mathbb{Z}\}$ is the error term (e.g., Choi (2018a)). An ARIMA-LSTM model is consisted by estimating L_t by an ARIMA(p, d, q) and estimating N_t by an LSTM model. The steps of the modeling process for an ARIMA-LSTM model are as follows (e.g., Caliwag and Lim (2019)):

- **Step 1:** Estimate L_t by an ARIMA model from the time series $\{r_t|r_t \in \mathbb{R}, t \in \mathbb{Z}\}$ and get the residual η_t and
$$\eta_t = r_t - \hat{L}_t.$$
- **Step 2:** To estimate N_t based on η_t, train the LSTM model by finding the values of the $\boldsymbol{W}$ type wight matrices and the $\boldsymbol{b}$ terms bias vectors that could make the loss function get a local minimum value.
- **Step 3:** Obtain the estimation of r_t by
$$\hat{r}_t = \hat{L}_t + \hat{N}_t.$$
- **Step 4:** Evaluate the trained hybrid model by cross-validation with regard to the predefined evaluation metrics and adjust the model structure if necessary.
- **Step 5:** Apply the well-defined hybrid model to do prediction.

3.3.2 VARMA-LSTM

A k-dimensional time series $\{\boldsymbol{R}_t|\boldsymbol{R}_t \in \mathbb{R}^k, t \in \mathbb{Z}\}$ can be consisting of a linear auto-correlation structure, a nonlinear part, and an error term (Caliwag and Lim, 2019). That is,

$$\boldsymbol{R}_t = \boldsymbol{L}_t + \boldsymbol{N}_t + \boldsymbol{\epsilon}_t, \tag{3.7}$$

where $\{\boldsymbol{L}_t|\boldsymbol{L}_t \in \mathbb{R}^k, t \in \mathbb{Z}\}$ is the linear portion, $\{\boldsymbol{N}_t|\boldsymbol{N}_t \in \mathbb{R}^k, t \in \mathbb{Z}\}$ is the nonlinear portion and $\{\boldsymbol{\epsilon}_t|\boldsymbol{\epsilon}_t \in \mathbb{R}^k, t \in \mathbb{Z}\}$ is the error term.

VARMA-LSTM Model is consisted by estimating $\boldsymbol{L}_t$ by a VARMA(p, q) and estimating $\boldsymbol{N}_t$ by an LSTM model. Let $\boldsymbol{R}_t = \boldsymbol{L}_t + \boldsymbol{N}_t + \boldsymbol{\epsilon}_t$ be a k-dimensional time series data defined in Equation 3.7. Then The steps of the modeling process for a hybrid model are as follows (Caliwag and Lim, 2019):

- **Step 1:** Estimate $\boldsymbol{L}_t$ by a statistical time series method and get the residual $\boldsymbol{\eta}_t$ and
$$\boldsymbol{\eta}_t = \boldsymbol{R}_t - \hat{\boldsymbol{L}}_t.$$

- **Step 2:**To estimate $\boldsymbol{N}_t$ based on $\boldsymbol{\eta}_t$, train the LSTM model by finding the values of the $\boldsymbol{W}$ type wight matrices and the $\boldsymbol{b}$ terms bias vectors that could make the loss function get a local minimum value.
- **Step 3:** Obtain the estimation of $\boldsymbol{R}_t$ by

$$\hat{\boldsymbol{R}}_t = \hat{\boldsymbol{L}}_t + \hat{\boldsymbol{N}}_t.$$

- **Step 4:** For the target logarithmic returns of stock r_t, evaluate the trained hybrid model by cross-validation with respect to the evaluation metrics and adjust the model structure if necessary.
- **Step 5:** Apply the well-defined hybrid model to do prediction.

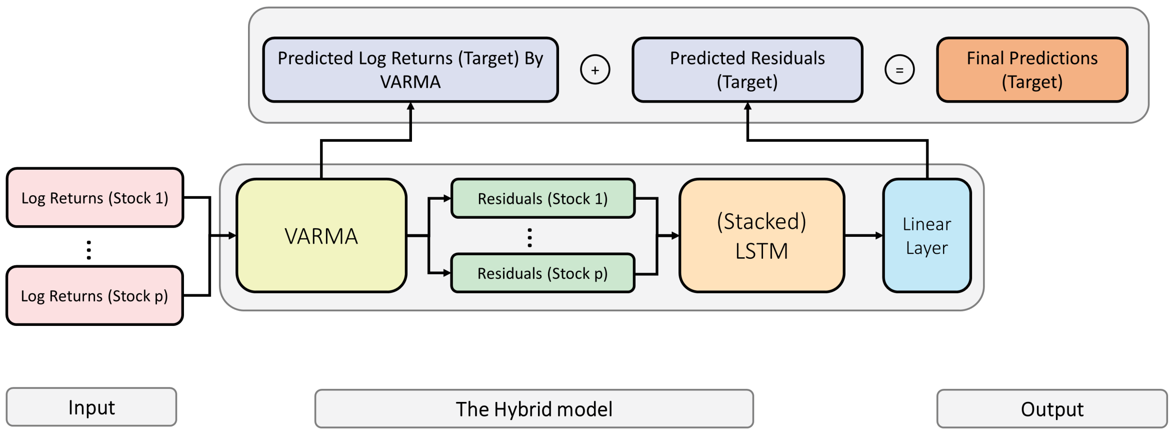


Figure 3.3: The Structure of the VARMA-LSTM Model.

3.4 Natural Language Processing (NLP) Techniques in Financial Sentiment Analysis

This chapter discusses one of the natural language processing (NLP) techniques, Bidirectional Encoder Representations From Transformers (BERT). The BERT could be applied to do text classification. Therefore, BERT could also be applied to do financial sentiment classification.

3.4.1 Bidirectional Encoder Representations From Transformers (BERT)

The Bidirectional Encoder Representations from Transformers (BERT) model was proposed by Devlin et al. (2018). It is an unsupervised language representation and the pre-trained transformers encoder. The BERT was trained by two tasks (Masked LM and Next Sentence Prediction (NSP)) on two data sets (the BooksCorpus (800M words) and English Wikipedia (2,500M words)). In the first task, "Masked LM", 15% of the tokens were masked in each input sequence at random, the BERT was trained by predicting the masked words. In the second task, "Next Sentence Prediction (NSP)", the BERT is trained by a binarized next sentence prediction task to identify the appropriate order between two sentences. Furthermore, The BERT is an open-source model [2].

A linear classifier needs to be added at the end of the BERT before employing the BERT to classify a sentence's polarity. Additionally, some sentences with labeled polarity are required to fine-tune the model. Furthermore, a linear classifier has a simple structure, and it is considered a weak leaner. Therefore, the BERT model needs to be decisive for making the linear weak learn to perform excellently.

There are four types of BERT:

- BERT-Base, Uncased: 12-layer, 768-hidden, 12-heads, 110M parameters.
- BERT-Large, Uncased: 24-layer, 1024-hidden, 16-heads, 340M parameters.
- BERT-Base, Cased: 12-layer, 768-hidden, 12-heads , 110M parameters.
- BERT-Large, Cased: 24-layer, 1024-hidden, 16-heads, 340M parameters.

An uncased BERT would convert all the input words into lowercase. On the other hand, a cased BERT would keep the form of input words.

3.5 BERT-based Financial Sentiment Index Enhanced (BERTFSIE) Models

Let $\boldsymbol{S}_{i,t}$ be the set of considered sentences for the asset i at time index t and $\boldsymbol{F}_{\boldsymbol{S}_{i,t}}$ be the polarities of $\boldsymbol{S}_{i,t}$ which are calculated by a BERT model. A financial sentiment index of the asset i at time t could be defined as follows:

$$b_{i,t} = f(\boldsymbol{F}_{\boldsymbol{S}_{i,t}}),$$

[2]`https://github.com/google-research/bert`

where $f(\cdot)$ is a real-valued function.

A financial time series model would be called the BERT-based financial sentiment index enhanced (BERTFSIE) version if it evolves the financial sentiment indices created by BERT only as of the input but not as the target output. For example, following the notation in Section 3.2.1, and let $\boldsymbol{B} = (\boldsymbol{b}_1, \cdots, \boldsymbol{b}_p)'$ be the financial sentiment index matrix, where $\boldsymbol{b}_i = (b_{i,1}, \cdots, b_{i,t})', 1 \leq i \leq p$. A BERT-based financial sentiment index enhanced LSTM models could be defined as follows:

$$LSTM(\boldsymbol{X}, \boldsymbol{B}) = \hat{\boldsymbol{Y}}.$$

In addition, there could be many BERT-based financial sentiment indices enhanced models, for instance, BERTFSIE-ARIMA-LSTM models, BERTFSIE-VARMA-LSTM models.

3.6 The Structure of the BERT-Based Financial Sentiment Index Enhanced (BERTFSIE) Multivariate LSTM

Let $\boldsymbol{X} = (\boldsymbol{x}_{1,t}, \ldots, \boldsymbol{x}_{p,t}, \boldsymbol{B}^{Bullish}_{1,t}, \boldsymbol{B}^{Bearish}_{1,t}, \ldots, \boldsymbol{B}^{Bullish}_{p,t}, \boldsymbol{B}^{Bearish}_{p,t})'$ be the input matrix, where where $\boldsymbol{x}_{i,t} = (x_{i,1}, \cdots, x_{i,t})'$, $\boldsymbol{B}^{Bullish}_{i,t} = (b^{Bullish}_{i,1}, \cdots, b^{Bullish}_{i,t})'$, $1 \leq i \leq p$, and $\boldsymbol{B}^{Bearish}_{i,t}$ has a similar structure with $\boldsymbol{B}^{Bullish}_{i,t}$. In the input matrix $\boldsymbol{X}$, each component would have the same dimension (i.e., in the same time length). Following the notation in Section 3.2.1 and the standard LSTM cell with a forget gate which is defined in Equations 3.4, the structure of the BERT-based financial sentiment index enhanced (BERTFSIE) multivariate LSTM could be defined as follows:

$$\begin{aligned} \boldsymbol{h}_t &= LSTM(\boldsymbol{X}), \\ \hat{x}_{target,t+1} &= \boldsymbol{W} \cdot vec(\boldsymbol{h}_t) + b, \end{aligned} \tag{3.8}$$

where $vec(\cdot)$ stands for the vectorization, $\boldsymbol{W}$, b are a weight matrix and a bias scalar, and $LSTM(\cdot)$ is the stacked LSTM model where each standard LSTM cell with a forget gate is defined in Equations 3.4.

The structure of the BERT-based financial sentiment index enhanced (BERTFSIE) multivariate LSTM could be illustrated by Figure 3.4, and the linear layer is formularized in Equation 3.8.

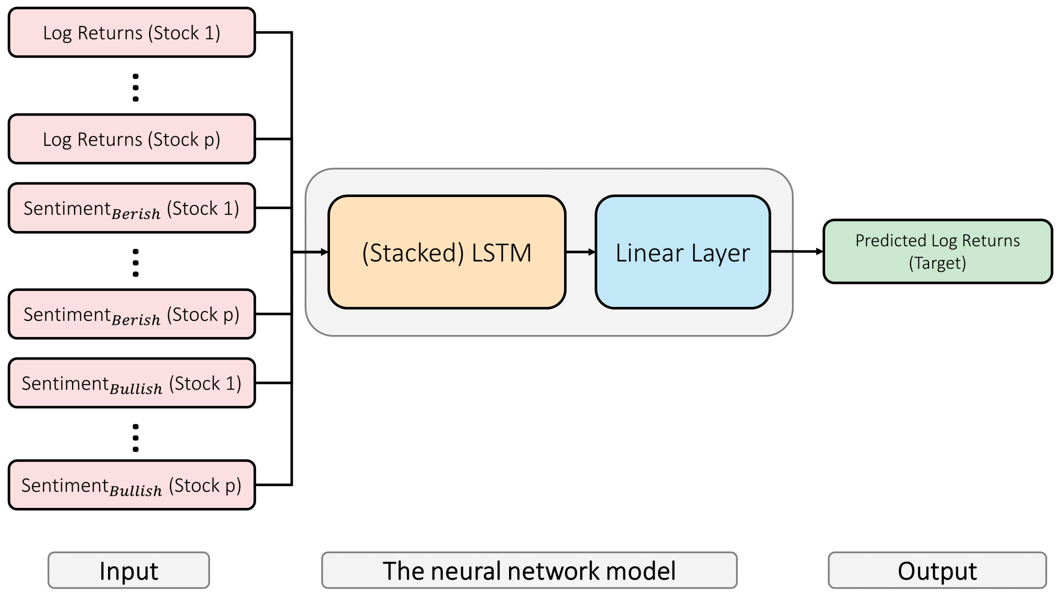

Log Returns (Stock 1)
Log Returns (Stock p)
Sentiment$_{Berish}$ (Stock 1)
Sentiment$_{Berish}$ (Stock p)
Sentiment$_{Bullish}$ (Stock 1)
Sentiment$_{Bullish}$ (Stock p)
(Stacked) LSTM
Linear Layer
Predicted Log Returns (Target)
Input
The neural network model
Output

Figure 3.4: The Structure of BERT-Based Financial Sentiment Index Enhanced (BERTFSIE) Multivariate LSTM Model.

3.7 The Structure of the BERT-Based Financial Sentiment Index Enhanced (BERTFSIE) VARMA-LSTM Model

Let $\boldsymbol{X}_t = (\boldsymbol{x}_{1,t}, \ldots, \boldsymbol{x}_{p,t})'$, $\boldsymbol{B}_t = (\boldsymbol{B}^{Bullish}_{1,t}, \boldsymbol{B}^{Bearish}_{1,t}, \ldots, \boldsymbol{B}^{Bullish}_{p,t}, \boldsymbol{B}^{Bearish}_{p,t})'$ be the input matrix and input financial sentiment index matrix, where each component in $\boldsymbol{X}, \boldsymbol{B}$ would have the same dimension (time length) that described in Section 3.6.

Following the notations in Section 3.3.2, the structure of the BERT-based financial sentiment index enhanced VARMA-LSTM Model could be described as:

$$\hat{\boldsymbol{L}}_{t+1} = VARMA(\boldsymbol{X}_t), \tag{3.9}$$

$$\hat{\boldsymbol{\epsilon}}_t = \boldsymbol{X}_t - \hat{\boldsymbol{L}}_t, \tag{3.10}$$

$$\boldsymbol{h}_{t+1} = LSTM(\hat{\boldsymbol{\epsilon}}_t, \boldsymbol{B}_t), \tag{3.11}$$

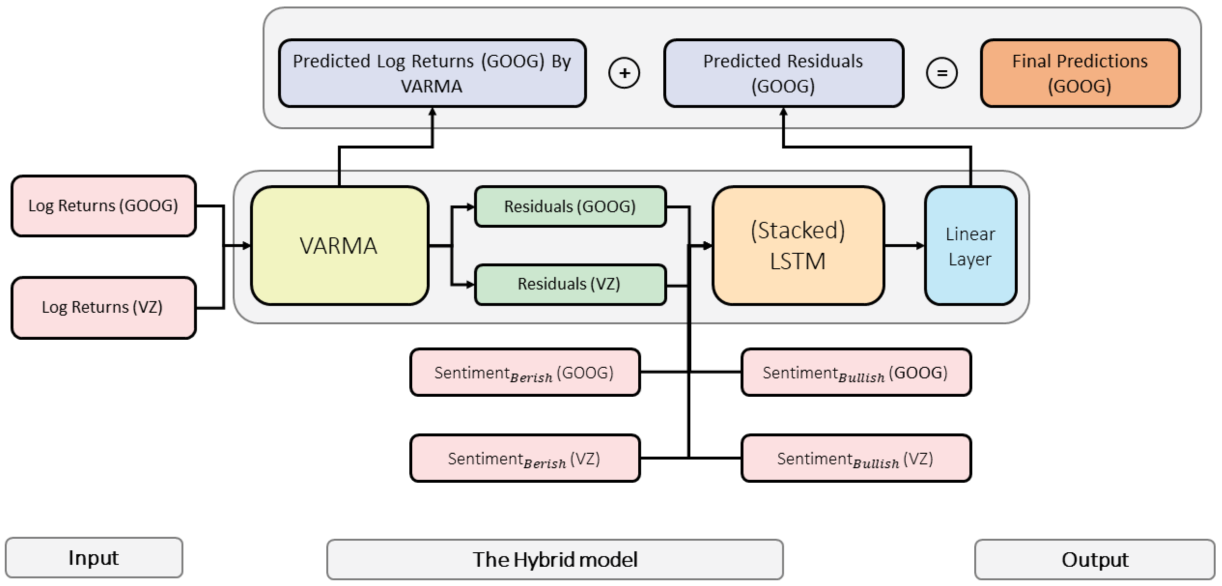
$$\hat{N}_{1,t+1} = \boldsymbol{W} \cdot vec(\boldsymbol{h}_{t+1}) + \boldsymbol{b}, \tag{3.12}$$

$$\hat{X}_{1,t+1} = \hat{L}_{1,t+1} + \hat{N}_{1,t+1}, \tag{3.13}$$

where $vec(\cdot)$ stands for the vectorization, $\boldsymbol{W}, b$ are a weight matrix and a bias.

The structure of the BERT-based financial sentiment index enhanced (BERTFSIE) VARMA-LSTM Model could be illustrated by Figure 3.5, and the linear layer is formularized Equation 3.12.

The BERT-based financial sentiment index enhanced (BERTFSIE) ARIMMA-LSTM model has the similar design as the BERTFSIE-VARMA-LSTM.



Predicted Log Returns (GOOG) By VARMA
+
Predicted Residuals (GOOG)
=
Final Predictions (GOOG)
Log Returns (GOOG)
Log Returns (VZ)
VARMA
Residuals (GOOG)
Residuals (VZ)
(Stacked) LSTM
Linear Layer
Sentiment$_{Berish}$ (GOOG)
Sentiment$_{Bullish}$ (GOOG)
Sentiment$_{Berish}$ (VZ)
Sentiment$_{Bullish}$ (VZ)
Input
The Hybrid model
Output

Figure 3.5: The Structure of the BERT-Based Financial Sentiment Index Enhanced VARMA-LSTM Model.

3.8 The Structure of Analyst Recommendation Index Enhanced (ARIE) Models

Analyst recommendation indices are created by analyst recommendation. The possible analyst recommendations are denoted as $A = \{c_1, \cdots, c_l\}$. The analyst recommendation polarity function is defined in Equation 3.14.

$$f(c_i) = \begin{cases} level_1 & \text{if } i = 1 \\ \vdots & \vdots \\ level_l & \text{if } i = l \end{cases}, \tag{3.14}$$

where $level_i$ is a predefined integer which is used to represent the polarity of the analyst recommendations, and $i \in \{1, 2, \cdots, l\}$.

For each time point, the available analyst recommendation polarities for stock i at time t could be converted into analyst recommendation index $a_{i,t}$ by some functions (e.g., arithmetic average, weighted average).

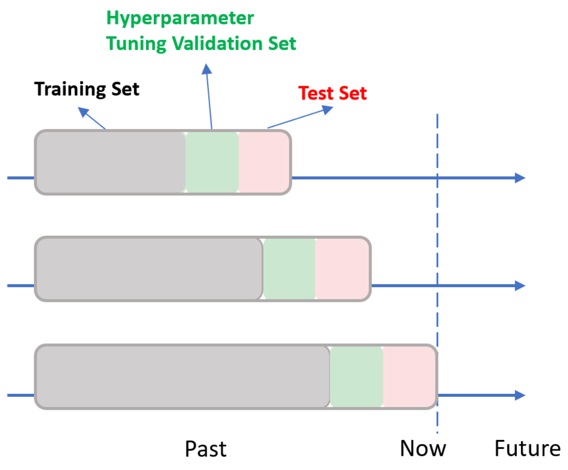
The analyst recommendations for p stocks are denoted as $\boldsymbol{A}_t = (\boldsymbol{a}_{1,t}, \cdots, \boldsymbol{a}_{p,t})'$. The analyst recommendation index enhanced (ARIE) models are replacing the BERT-based financial sentiment index $\boldsymbol{B}_t$ by $\boldsymbol{A}_t$ in the input data.

3.9 Cross-Validation for Sequential Data (Backtest)

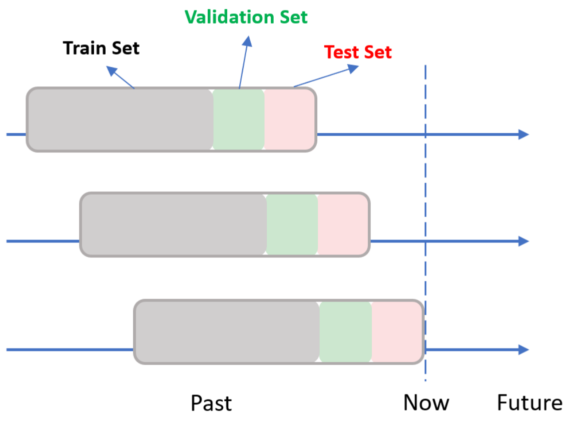
Compared with applying cross-validation to non-sequential data, the sequential data can not be re-sampled to reconstruct the new training set and the validation set during the cross-validation. The cross-validation for sequential data is also called "Backtest". There are two types of backtesting: the backtest with expanding window and the backtest with the sliding window, and they are shown in Figure 3.6. Unlike the statistical time series models, the neural network models have a bunch of hyper-parameters that need to be selected by backtests, for example, the learning rate of an optimizer, the number of epochs, the number of hidden layers, and the number of cells in each hidden layer. Moreover, these hyper-parameters could be selected by the model performance in the validation set. Figure 3.6 is used to illustrate how the cross-validation works with sequential data.

There are four types of walk-forward (rolling) forecast strategies could be used in the backtesting of financial time series analysis (e.g., Hyndman 2014; Hyndman and Athanasopoulos 2018):

- **One-step forecasts without re-estimation:** The model would be estimated by training data, and then one-step forecasts are computed on the testing data sets.
- **One-step forecasts with re-estimation:** The model would be re-fitted at each iteration before every forecast is performed.
- **Multi-step forecasts without re-estimation:** The model would be estimated by training data, and then multi-step forecasts are computed on the testing data sets.
- **Multi-step forecasts with re-estimation:** The model would be re-fitted at each iteration before every multi-step forecast is performed.



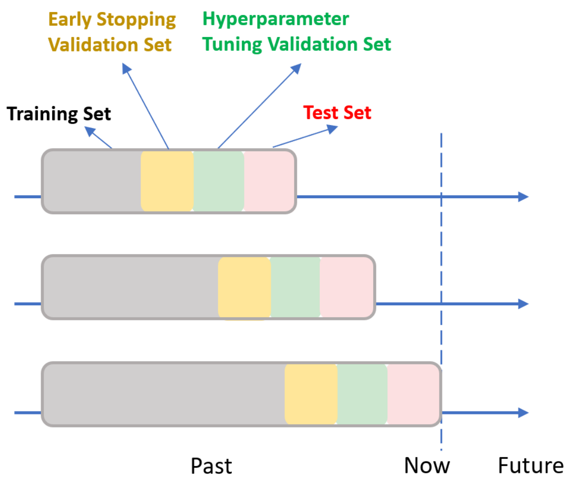
(a) The Backtest With Expanding Window.



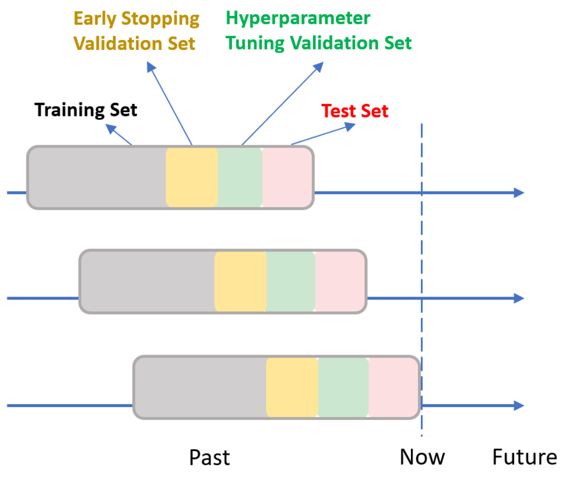
(b) The Backtest with Sliding Window.

Figure 3.6: The Backtest With Expanding Window and Sliding Window.

Different from the statistical sequential data modeling approaches, the application of neural network models in limited data size would suffer from the over-fitting problem. Therefore, the early stopping validation set for sequential data is proposed. It would be applied to all the models including a neural network component in this dissertation (e.g., Algorithm 5, Algorithm 6, Algorithm 7). In addition, the hyperparameters tuning validation set (shown in Figure 3.7) is used for score voting hyperparameters selection that is described in Algorithm 8.



(a) The Backtest With Expanding Window (Early Stopping).



(b) The Backtest With Sliding Window (Early Stopping).

Figure 3.7: The Backtest With Expanding Window And Sliding Window (Early Stopping).

3.10 Adaptive Epoch Selection Strategy

In the hyperparameter tuning process of a neural network model, the epoch size (how many rounds should we train the model based on the whole training data) is crucial. However, the epoch would be predetermined in practice instead of being involved in hyperparameter tuning.

If the hyperparameters searching space is ample, a preassigned epoch would not be suitable. Because if a tiny epoch is selected, most of the models would be insufficiently trained in the rolling forward forecast strategy. On the other hand, even though a large epoch could guarantee the models have been well trained, it would lead to an unbearable time cost when the validation data size or test data size is large. Consequently, an adaptive epoch selection strategy is proposed to solve this problem in Algorithm 1.

Algorithm 1: Pseudocode of Training Neural Network Models With an Adaptive Epoch Size.

```
Input: Epoch: Initial size (A small number).
Input: Other required parts for training a neural network model.
Output: The Model which has the best performace in the validation set.

1  $Validation_{loss} = [\,]$;
2  while $i < Epoch$ do
3      Train the Neural Network Model;
4      Add the validation loss $vali_{loss}$ into $Validation_{loss}$;
5      if $vali_{loss} == min(Validation_{loss})$ then
6          Save current model as $model_{best}$;
7          $Epoch+ = 1$;
8      end
9      $i+ = 1$;
10 end
```

3.11 Optimizers for Neural Network Models

Let us consider a differentiable loss function (also called an 'error function'):

$$f(\boldsymbol{\theta}) : \mathbb{R}^k \longrightarrow \mathbb{R},$$

where $\boldsymbol{\theta}$ is the parameter vector in $\mathbb{R}^k$.

Our goal is to find one of the parameter vectors $\{\boldsymbol{\theta}_{optimal}\}$ that minimize (locally) $f(\boldsymbol{\theta})$ in an iterative way, which is based on a predefined update rule $\mathcal{U}$, within finite steps. The update rule $\mathcal{U}$

can be defined as:

$$\boldsymbol{\theta}_{t+1} = \mathcal{U}(\mathcal{H}_t, \boldsymbol{\phi}_t),$$

where, $\mathcal{H}_t = \{\boldsymbol{\theta}_i, f(\boldsymbol{\theta}_i), \nabla f(\boldsymbol{\theta}_i)\}_{i=0}^{t}$ is the available iteration information before time t, $\boldsymbol{\phi}_t \in \mathbb{R}^l$ is a l dimensional hyper-parameters vector in time t (e.g., Choi et al. 2019). In the following part, some iterative optimization methods (Optimizers) for neural network models are discussed.

3.11.1 Gradient Descent (GD)

Gradient descent (GD) is also called "batch gradient descent". It applies whole training (full batch) data set to calculate and update the loss function $f(\boldsymbol{\theta})$. Additionally, the update rule is defined as:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \cdot \nabla f(\boldsymbol{\theta}_t),$$

where η is called "learning rate" and ∇ stands for the gradient of a vector function when $\boldsymbol{\theta} = \boldsymbol{\theta}_t$ (Ruder, 2016).

3.11.2 Stochastic Gradient Descent (SGD)

The basic idea of Stochastic Gradient Descent (SGD) is from Robbins and Monro (1951). Compared with gradient descent (GD), stochastic gradient descent uses each data point (a single data point) to calculate and update the loss function. It makes the loss function converge to a local minimum much faster but unstable than the gradient descent. To solve this problem, the mini-batch stochastic gradient descent is proposed. The difference between the stochastic gradient descent and the mini-batch stochastic gradient descent is that mini-batch SGD would calculate and update the loss function by a subset in the training data set instead of a single data point in the update step.

3.11.3 Stochastic Gradient Descent with Momentum (SGDM)

The stochastic gradient descent with momentum (SGDM) is an edited version of the stochastic gradient descent method. The original idea can be found in Polyak (1964). In the update step, SGDM considers not only the current information but also the historical updating information. The stochastic gradient descent with momentum (SGDM) can be defined as:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \cdot \boldsymbol{\nu}_t,$$

where $\boldsymbol{\nu}_t = \lambda \cdot \boldsymbol{\nu}_{t-1} + \nabla f(\boldsymbol{\theta}_t)$, $\lambda \in [0, 1]$, and $\boldsymbol{\nu}_{-1} = 0$.

When compared with SGD, it has been proved to converge faster and not easily trapped by a local minimum (Qian, 1999; Liu et al., 2020).

3.11.4 Adaptive Gradient Algorithm (Adagrad)

For some tasks, a fixed learning rate is not appropriate. Because when the value of the loss function in the current parameters vector is far from a local minimum value, the update step should be significant to converge faster. However, when the current parameters vector is close to a local minimum value, the update step should be small to avoid oscillation during the training process.

Adaptive Gradient Algorithm (Adagrad) is a method with adaptive learning rate for each component of the parameters (Duchi et al., 2011). The update process for $\boldsymbol{\theta}_{t,i}$ (the component i of $\boldsymbol{\theta}_t$) is described as follows:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,i}} + \epsilon} \nabla f(\boldsymbol{\theta}_{t,i}),$$

where $G_{t,i} = \sum_{j=0}^{t} (\nabla f(\boldsymbol{\theta}_{j,i}))^2$, $G_{0,i} = 0$, and ϵ is a small positive number which can make the fraction valid.

3.11.5 Root Mean Square Propagation (RMSProp)

Root Mean Square Propagation (RMSProp) is another well-know but unpublished method (Tieleman et al., 2012). In this method, the learning rate during the updating process is adaptive instead of a predefined fixed value. The update process is described as follows:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\eta}{\sqrt{\boldsymbol{\nu}_t} + \epsilon} \nabla f(\boldsymbol{\theta}_t),$$

where $\boldsymbol{\nu}_t = \lambda \cdot \boldsymbol{\nu}_{t-1} + (1-\lambda) \cdot (\nabla f(\boldsymbol{\theta}_t))^2$, $\lambda \in [0, 1]$, $\boldsymbol{\nu}_{-1} = 0$, and ϵ is a small positive number which can make the fraction valid.

3.11.6 Adaptive Moment Estimation (ADAM)

Adaptive Moment Estimation (ADAM) is another adaptive learning rates method which considers the advantage of both RMSProp and Adagrad (Kingma and Ba, 2014). The update process is defined as follows:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\eta}{\sqrt{\hat{\boldsymbol{\nu}}_t} + \epsilon} \hat{\boldsymbol{m}}_t,$$

where

$$\begin{aligned}
\hat{\boldsymbol{m}}_t &= \frac{\boldsymbol{m}_t}{1-\beta_1^t}, \\
\hat{\boldsymbol{\nu}}_t &= \frac{\boldsymbol{\nu}_t}{1-\beta_2^t}, \\
\boldsymbol{m}_t &= \beta_1 \cdot \boldsymbol{m}_{t-1} + (1-\beta_1)\cdot \nabla f(\boldsymbol{\theta}_t), \\
\boldsymbol{\nu}_t &= \beta_2 \cdot \boldsymbol{\nu}_{t-1} + (1-\beta_2)\cdot (\nabla f(\boldsymbol{\theta}_t))^2,
\end{aligned}$$

and $\beta_1, \beta_2 \in [0, 1)$, ϵ is a small positive number which can make the fraction valid.

The adaptive moment estimation method has been proved to be powerful in deep learning models. For example, the following models are trained by ADAM: Transformer (Vaswani et al., 2017), BERT (Devlin et al., 2018).

3.11.7 Adapting Stepsizes by the Belief in Observed Gradients (AdaBelief)

Adapting Stepsizes by the Belief in Observed Gradients (AdaBelief) is proposed to be a improved version of ADAM, and it edited the biased second raw moment estimate (Kingma and Ba, 2014).

The update process is defined as follows:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \frac{\eta}{\sqrt{\hat{\boldsymbol{s}}_t} + \epsilon}\hat{\boldsymbol{m}}_t,$$

where

$$\begin{aligned}
\hat{\boldsymbol{m}}_t &= \frac{\boldsymbol{m}_t}{1-\beta_1^t}, \\
\hat{\boldsymbol{\nu}}_t &= \frac{\boldsymbol{\nu}_t}{1-\beta_2^t}, \\
\boldsymbol{m}_t &= \beta_1 \cdot \boldsymbol{m}_{t-1} + (1-\beta_1)\cdot \nabla f(\boldsymbol{\theta}_t), \\
\boldsymbol{s}_t &= \beta_2 \cdot \boldsymbol{s}_{t-1} + (1-\beta_2)\cdot (\nabla f(\boldsymbol{\theta}_t) - \boldsymbol{m}_t)^2 + \epsilon,
\end{aligned}$$

and $\beta_1, \beta_2 \in [0, 1)$, ϵ is a small positive number which can make the fraction valid.

3.12 Evaluation Metrics

In the following section, some popular evaluation matrices for continuous and discrete data are briefly discussed.

3.12.1 Evaluation Metrics (Continuous Data)

The Mean Squared Error (MSE), the Root Mean Squared Error (RMSE), and the Mean Absolute Error (MAE) were discussed in Torgo (2017), and they are restated in this section.

The **mean squared error (MSE)** is defined as:

$$MSE = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} (y_i - \hat{y}_i)^2,$$

where $\hat{y}_i$ is the i-th prediction of the model and N_{test} is the sample size of the test data. Moreover, the **root mean squared error (RMSE)** refers to the square root of MSE.

The **mean absolute error (MAE)** is defined as:

$$MAE = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} |y_i - \hat{y}_i|,$$

where $\hat{y}_i$ is the i-th prediction of the model and N_{test} is the sample size of the test data.

3.12.2 Evaluation Metrics (Discrete Data)

According to Fawcett (2006), there are some widely accepted evaluation metrics for discrete data, and they are rephrased in this section. Before the introduction of the evaluation matrices, a confusion matrix is presented in Figure 3.8, and all the rest evaluation matrices are based on the confusion matrix.

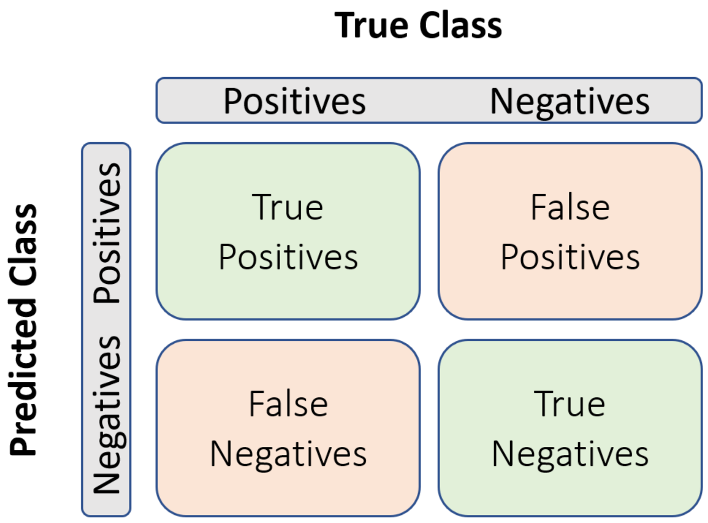


Figure 3.8: Confusion Matrix.

The **sensitivity** (also known as **true positive rate, hit rate or recall**) is defined as:

$$\text{sensitivity} = \frac{\text{Positives correctly classified}}{\text{Total positives}}.$$

The **false positive rate** (also known as **false alarm rate**) is defined as:

$$\text{False Positive Rate} = \frac{\text{Negatives incorrectly classified}}{\text{Total negatives}}.$$

The **precision** is defined as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}.$$

The F_1 **Score** is defined as:

$$F_1\ Score = \frac{2}{\text{Precision}^{-1} + \text{Recall}^{-1}}.$$

If there are more than two labels in the data, a **weighted F_1 score** could be applied. Let N be the number of classes in the date set, N_{test} be the total number of points in the testing data set, F_1 score$_i$ be the F_1 score for class i, and w_i be the number of true instances in class i.. Then the **weighted F_1 score** is defined as:

$$\text{Weighted } F_1 \text{ score} = \frac{1}{N_{test}} \sum_{i=1}^{N} (F_1 \text{ score}_i \times w_i).$$

The advantage of weighted F_1 score is that it can take label imbalance into account.

The **receiver operating characteristic (ROC) Graph** is defined as: A graph shows the discrimination ability of a classifier by plotting the true positive rate against the false positive rate in a two-dimensional space.

The **area under a receiver operating characteristic curve (AUC)** is defined as the area under a receiver operating characteristic curve (AUC), which is a scalar between 0 and 1. The higher the AUC, the better performance of a binary classifier. When calculating the AUC, the input must be a probability or a score from a classifier. The sigmoid function $S(x)$ could be applied to convert the model output to a valid score. For a data set with imbalance labels, a **weighted AUC** could be defined in a similar way as the **weighted F_1 score**.

CHAPTER 4

EMPIRICAL ANALYSIS

4.1 Experimental Study

The experiment aims to compare the performance of the statistical time series models (ARIMA, ARIMAX, VARMA), the machine learning models (LSTM), the hybrid models (ARIMA-LSTM, VARMA-LSTM), the BERTFSIE models (BERTFSIE-LSTM, BERTFSIE-VARMA-LSTM). The analyst recommendation index enhanced (ARIE) models (ARIE-Multi LSTM, ARIE-VARMA-LSTM) in terms of prediction of the stocks' logarithmic returns and their direction in different four sections (Finance, pharmaceutical, retail, and technology industry).

In the models' performance comparison, three market indices (Dow Jones Industrial Average Index, the Standard and Poor's 500 Index, the Nasdaq-100 Index) are also considered covariates in some multivariate models.

To make all the models' performance comparable in this dissertation, we only focus on one stock log return and their directions in each section. For example, there are two stocks (stock A and stock B) are included in the finance section. An ARIMA model is employed, and the predictions have been performed for stock A, and the model performance has been evaluated by a predefined metric M. When the VARMA model is applied in this section, the two stocks (stock A and stock B) are contained, even though the model would predict for both stock A and stock B. Nevertheless, when the performance of VARMA is evaluated by metric M, only the prediction of the stock A is considered. Through this process, these two models' performances are comparable.

To test the effect of the individuals' reactions to the stock logarithmic returns, the performance of BERTFSIE type models would be compared with their original models.

To test the effect of the analysts' recommendations on the stock logarithmic returns, the suggestion would be first converted to analysts' recommendations indices. Then these indices are also considered as covariates in the models' performance comparison for some multivariate models. Additionally, The performance of these analysts' recommendations indices enhanced models would also be compared with BERTFSIE type models.

In summary, the following comparisons are performed:

- The forecast ability of univariate and multivariate models is compared.
- The impact of considering market indices as covariates would be tested by comparing it with the models without considering these market indices.
- The influence of the individual investors' reactions to the stocks' logarithmic returns would be tested by comparing BERTFSIE type models with the original models.
- The effect of the analysts' recommendations on the stocks' logarithmic returns would also be tested by comparing BERTFSIE type models with the models which take the analysts' recommendations indices as covariates (ARIE-models).

4.2 Experimental Setup

The experimental setup is as follows:

- **Time Duration**: The time duration is from 7/14/2016 to 8/24/2021 (A total of valid days are 1288).
- **Backtesting Strategy**: One-step forward forecasts with re-estimation and the backtesting with expanding window are picked for statistical time series models. Furthermore, one-step forward forecasts with re-estimation and the backtesting with expanding window (early stopping) are chosen for the models which includes an LSTM component. The criteria for early stopping are the same as the one used for gradient descent.
- **Variables**: The stocks' logarithmic returns, the BERT-based financial sentiment indices (BERTFSI), the analyst recommendation indices (ARI), and three market indices (Dow Jones Industrial Average Index, The Standard and Poor's 500 Index, The Nasdaq-100 Index) are considered in the data set.
- **Evaluation Metrics**: RMSE, MAE, Weighted F_1 Score, and Weighted AUC Score are chosen as the evaluation metrics.
- **Test Data Proportion**: Training data proportion (90%) and testing data proportion (10%); Training data proportion (80%) and testing data proportion (20%); Training data proportion (70%) and testing data proportion (30%).
- **Standardization**: Standardization is applied to all the models, which include an LSTM component. Only the early stopping validation set is standardized by the mean and slandered deviation of the training data. When it comes to the hyper-parameter tuning validation set and the testing set, they would not be changed, and the predictions generated by standardized data would be inverted by the mean and slandered deviation of the training data.

The time duration is for the logarithmic returns, which are calculated by Equation 3.1. For the original stock prices, the time duration is from 7/13/2016 to 8/24/2021 (Totally data size is 1289). Moreover, there are three test data proportion strategies, and they are designed to test the sensitivity of the model's performance to the test data proportion.

There are four sections in the empirical analysis: finance, pharmaceutical, retail, and technology industry. For each section, there are two stocks included, and they are shown in Table 4.1.

The criteria for considering a company in this experiment is that the counts of associated individual investors' reactions to this company should be more than 7000 (This is just a predefined number to ensure the counts are sufficient) in the conversation section from Yahoo finance. The inadequate individuals' reactions to stock will lead to an unsatisfied forecast-ability of the BERTFSIE type models.

Table 4.1: The Description of Selected Companies in the Empirical Analysis.

Section	Company	Company Stock Code
Technology	Alphabet Inc.	GOOG
Technology	Verizon Communications Inc.	VZ
Finance	Citigroup Inc.	C
Finance	The Goldman Sachs Group, Inc.	GS
Pharmaceutical	Bristol-Myers Squibb Company	BMY
Pharmaceutical	AbbVie Inc.	ABBV
Retail	Costco Wholesale Corporation	COST
Retail	Target Corporation	TGT

For the experimental study in this dissertation, the data preparation process of the selected four industries is similar. Therefore, the technology industry is picked for illustration. Additionally, for simplicity purposes, we use "Verizon (VZ)" as the abbreviation of "Verizon Communications Inc.", and "Alphabet (GOOG)" as the abbreviation of "Alphabet Inc."

4.3 Data Collection

For each selected company in one of the picked industries, the raw data set consists of four parts:

- The historical stock prices,
- Individual investors' reactions to the stock,

- The analysts' recommendations,
- The market indices.

Again, to illustrate the structure of the collected data set for each stock, the data set of Verizon Communications Inc. is selected as an example. The four parts in the example are:

- The historical stock prices of Verizon Communications Inc. from Yahoo finance[1],
- Analysts' recommendations for Verizon Communications Inc. from Yahoo finance[1],
- Three market indices (Dow Jones Industrial Average Index, The Standard and Poor's 500 Index, The Nasdaq-100 Index)[1],
- The individuals' reactions to Verizon Communications Inc. in the conversation section from Yahoo finance[2].

The first three parts could be directly downloaded by a module "Yahoo! Finance market data downloader"[1] in Python. However, the last part is not downloadable. Therefore, some web scraping techniques are required to solve this problem. After checking the source code of the webpage, not all the reactions were loaded simultaneously, and there were only 20 reactions initially. The button "Show More" needs to be clicked to get more reactions, and the link to the webpage would not change during this process. This technique for webpage design is called "Asynchronous JavaScript and XML (AJAX)". Furthermore, even if the API of the JSON file of reactions could be identified, they are not directly accessible due to the encryption.

The selenium web-driver is a web framework for automatically testing web applications. One of the advantages of the selenium web driver is that it can simulate user actions. The selenium web driver is selected and applied to solve this problem. Before extracting the reactions, the robots exclusion protocol[3] of the website "Yahoo! Finance" has been checked. There are no limitations for robots to access the following web address: `https://finance.yahoo.com/quote/VC/community?p=VC`.

4.3.1 The First Component of the Data Set

For the first component of the data set of each considered stock, Table 4.2 shows the historical stock information raw data format of Verizon from 7/13/2016 to 8/24/2021. The rest raw data

[1]`https://pypi.org/project/yfinance/`
[2]`https://finance.yahoo.com/quote/VZ/community?p=VZ`
[3]`https://finance.yahoo.com/robots.txt`

format for the considered stock prices are similar. In this experiment, the adjusted close prices are selected for the stocks instead of the raw closed prices.

Table 4.2: The Historical Stock Information of Verizon Communications Inc. From 7/13/2016 to 8/24/2021.

Date	High	Low	Open	Close	Volume	Adj Close[4]
7/13/2016	67.34	66.60	67.18	66.98	365000	66.98
7/14/2016	68.53	67.34	67.53	67.41	386000	67.41
⋮	⋮	⋮	⋮	⋮	⋮	⋮
8/23/2021	108.84	106.52	107.59	107.82	156900	107.82
8/24/2021	110.68	108.54	108.54	110.03	98200	110.03

Note. The unit of High (Price), Low (Price), Open (Price), Close (Price), and Adj Close (Price) is dollars. The close price is adjusted for splits, and the Adjusted close price is adjusted for both dividends and splits.

4.3.2 The Second Part of the Data Set

The second part of the data set is the analysts' recommendations indices. The analysts' recommendations of a stock could be downloaded from Yahoo finance[5]. The raw data format of analysts' recommendations can be shown in Table 4.3. There are 68 analysts' recommendations in Table 4.3, and they are too sparser when compared with the number of historical stocks prices. It needs to be converted into the analysts' recommendations indices before being input into the models. The method of how to convert the analysts' recommendations to indices is described in Section 4.4.

Table 4.3: The Historical Analysts' Recommendations of Verizon Communications Inc. From 7/13/2016 to 8/24/2021.

Date	Firm	To Grade	From Grade	Action
7/20/2016 (7:42:35 AM UTC)	Oppenheimer	Perform	Outperform	Down
7/27/2016 (11:58:25 AM UTC)	Hilliard Lyons	Neutral	Buy	Down
7/27/2016 (2:01:53 PM UTC)	JP Morgan	Neutral		main
⋮	⋮	⋮	⋮	⋮
6/25/2021 (11:55:01 AM UTC)	Redburn Partners	Sell	(Null)	Init
7/22/2021 (12:00:59 PM UTC)	Credit Suisse	Neutral	(Null)	Main

Note. "Init" means "Initial Rating"; "Main" means "Maintain Current Positions" in the column "Action".

[5] `https://pypi.org/project/yfinance/`

4.3.3 The Third Piece of the Data Set

For the data of each section, some market indices should be considered as covariates. Therefore, Dow Jones industrial average index, the Standard and Poor's 500 indices, the Nasdaq-100 index are considered in this experiment. The raw data format of these three market indices can be shown in Table 4.4.

Table 4.4: The Historical Values of Selected Market Indices From 7/13/2016 to 8/24/2021.

Date	Dow Jones Industrial Average (DJI)	NASDAQ 100 (NDX)	S&P 500 ($GSPC$)
7/13/2016	18372.12	4565.77	2152.43
7/14/2016	18506.41	4596.49	2163.75
7/15/2016	18516.55	4589.83	2161.74
⋮	⋮	⋮	⋮
8/23/2021	35335.71	15312.82	4479.53
8/24/2021	35366.26	15357.68	4486.23

Note. All the values are adjusted close values which are adjusted for both dividends and splits.

4.3.4 The Fourth (Last) Portion of the Data Set

The last portion of the data set is the BERT-based financial indices of the stock. Furthermore, the stock of Verizon is still selected as an example. The raw data format of individual investors' reactions to Verizon is a JSON file. Each JSON file contains at most 20 comments, and an example is illustrated by List 4.1

Listing 4.1: Example JSON file format of individual investors' reactions to Verizon communications inc. in Yahoo finance.

```
{"contextId":"finmb_415798",
 "messageId":"46cdcd45-ccd8-40ee-9f8f-b3204eabac11",
 "namespace":"yahoo_finance",
 "meta":{...},
 "details":{{"userText":"I understand 53bn is alot of debt to take
    on.  However interest coverage is currently over 6x. If we
    factor in their payments on the 53bn it wont really affect our
    coverage by all that much. Maybe by a point or 2.  Dividend
    should be able to remain intact."}},
 "tags":[],
 "userLabels":["NEUTRAL"],
 "reactionStats":{"upVoteCount":2,
                  "downVoteCount":0,
                  "abuseVoteCount":0,
                  "replyCount":1},
"lastActivity":{"activityAt":1616489101,
                  "activityAuthor":{
                           "guid":"DXV2DOSRK...N3TPE","userType":"
                              YAHOO_USER",
                           "nickname":"Longs10ga",
                           "image":{"url":"https://s.y...sq.jpg",
                                   "height":192,
                                   "width":192},
                           "userCategory":"REGULAR_USER"}},
"index":"v=1:s=time:t=1629913289:off=1160"}
```

The raw data of the individual investors' reactions cannot be input into the model directly because they are not numerical data. The comments need to be first extracted out of the JSON file, and then they would be treated as the input of a BERT model and converted to the financial sentiments. Finally, the financial sentiments are converted into financial sentiment indices and inputted into the BERTFSIE models. In addition, the preparation process of individual investors' reactions to Verizon is described in Section 4.4.

4.4 Data Preparation

The raw data collected in Section 4.3 need to be prepared before feeding into the models. As we mentioned before, Verizon has been selected as the example company to show the data preparation process by following the structure of Section 4.3.

4.4.1 The First Component of the Data Set

The historical stock logarithmic returns and their direction movements from Verizon are calculated by Equation 3.1 and 3.2 based on their adjusted close prices.

Table 4.5: The Logarithmic Returns (the Original Stock Logarithmic Returns $\times$ 1000) and Their Movement Direction of Verizon From 7/14/2016 to 8/24/2021.

Time	Log Return (Verizon)	Direction (Verizon)
7/14/2016	-2.861	-1
⋮	⋮	
8/23/2021	-1.803	-1
8/24/2021	-7.426	-1

4.4.2 The Second Part of the Data Set

The raw data format of analysts' recommendations is not numerical. They need to be converted into analysts' recommendations indices before being input into the model. However, different stock-research rating providers (recommendation providers) have other descriptions for the same grade (e.g., some sample descriptions shown in Table 4.3). The analysts' recommendations indices are designed with five grades, and the different terms for the same grades have been matched into these five grades based on the conversion table shown in Figure 4.1 (The description in the plot is created based on the website[6]). There are some converted analysts' recommendations is shown in Table 4.6.

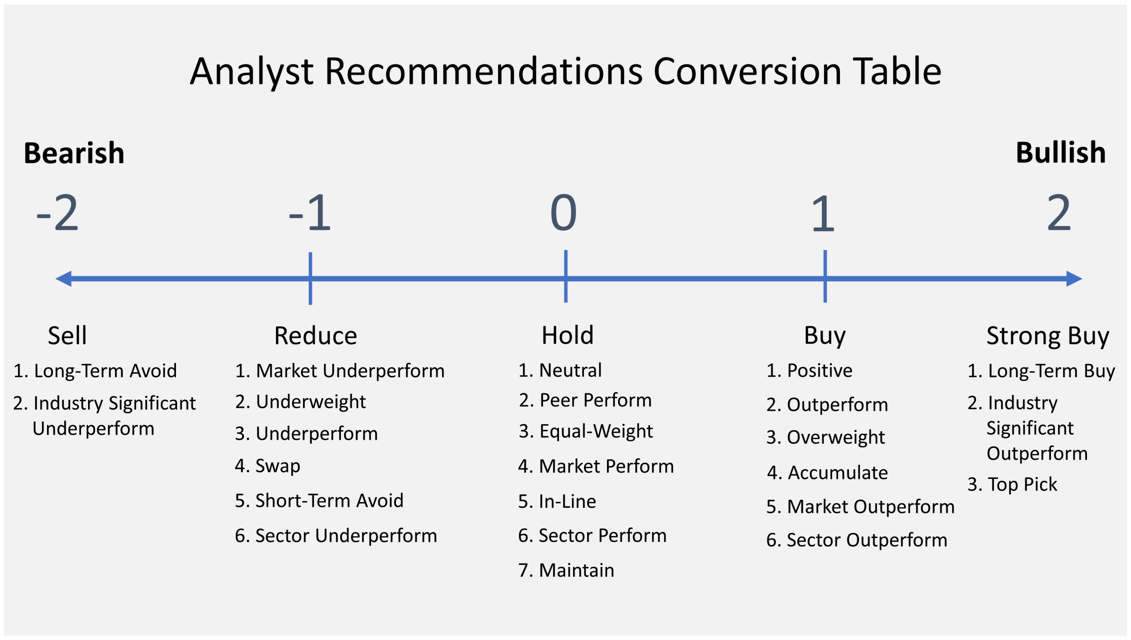


Figure 4.1: Analysts' Recommendations.

The date shown in Table 4.6 has a detailed available time for each analysts' recommendation. These time points should be matched to the adjusted closed price. Moreover, there are multiple analysts' recommendations available within one trading date. The analysts' recommendations indices are designed as follows:

1. The date of each analysts' recommendation is converted from Coordinated Universal Time (UTC) to Eastern Standard Time (EST).

[6]`https://www.marketwatch.com/tools/guide.asp`

Table 4.6: The Converted Historical Analysts' Recommendations of Verizon Communications Inc. From 7/13/2016 to 8/24/2021.

Date	Firm	Numerical Scaled Grade
7/20/2016 (7:42:35 AM UTC)	Oppenheimer	0
7/27/2016 (11:58:25 AM UTC)	Hilliard Lyons	0
7/27/2016 (2:01:53 PM UTC)	JP Morgan	0
⋮	⋮	⋮
6/25/2021 (11:55:01 AM UTC)	Redburn Partners	-2
7/22/2021 (12:00:59 PM UTC)	Credit Suisse	0

2. If the detailed time point of each analysts' recommendation is before 4 PM on a specific day, we considered that it happened on that day. Otherwise, it is considered that happened on the next day.

3. An analysts' recommendation index on a day is constructed by the arithmetic mean value from all the analysts' recommendations on that day.

4. If there are no analysts' recommendations available on 7/13/2016, the analyst recommendation index is set as 0.

5. Due to the sparsity of the analyst recommendations, when there are no recommendations, the indices are considered to have the same value as the previous index, which has non-zero values.

After the conversion process, the prepared historical analyst recommendation indices from 7/14/2016 to 8/24/2021 of Verizon are shown in Table 4.7.

Table 4.7: The Historical Analysts' Recommendations Indices of Verizon Communications Inc. From 7/14/2016 to 8/24/2021.

Date	Analyst Recommendation Index
7/14/2016	0
⋮	⋮
7/27/2020	0.667
⋮	⋮
8/24/2021	0

4.4.3 The Third Piece of the Data Set

The logarithmic returns of the three market indices are calculated by Equation 3.1. Additionally, the logarithmic returns of these three market indices are shown in Table 4.8.

Table 4.8: The Logarithmic Returns (the Original Logarithmic Returns $\times$ 1000) of the Three Market Indices Historical Values From 7/13/2016 to 8/24/2021.

Date	Dow Jones Industrial Average (DJI)	NASDAQ 100 (NDX)	S&P 500 ($GSPC$)
7/14/2016	5.245	7.283	6.706
7/15/2016	-0.929	0.548	-1.450
⋮	⋮	⋮	⋮
8/23/2021	8.488	6.121	14.488
8/24/2021	1.495	0.864	2.925

Note. All the values are adjusted close values which are adjusted for both dividends and splits.

4.4.4 The Fourth (Last) Portion of the Data Set

The last component of the data set is the BERT-based financial indices of the stock. The individual investors' reactions to Verizon Communications Inc. stock prices needed to be cleaned and then fed into the BERT model and converted to sentiment indices. The reactions preparation strategies are as follows:

1. Removing all Emojis (e.g., ☺, ☹, ☻.) in the reactions.
2. Extracting necessary information by regular expressions (a sequence of characters that define a search pattern).
3. Discarding the reactions which contain only figures.
4. Converting Unix timestamps to traditional dates format (EST). For example, "1602423801" is equivalent to "10/11/2020 @ 8:43 pm (EST)".

Every trading day, the adjusted close prices are available at 4:00 PM (EST). However, the reactions have a more exact time. To match the reactions' available time to the time of daily logarithmic returns, we need to adjust the time of a reaction. For example, a reaction would be classified as the same day if it is created before 4:00 PM (EST) on that day, and a reaction would be classified as the next day if it is created after 4:00 PM (EST) on that day. For example, the reaction created at 3:30 PM (EST) on 10/11/2020 is classified as 10/11/2020, and the reaction created at 4:30 PM (EST) on 10/11/2020 is classified as 10/12/2020. On the non-trading days, all the adjusted closed stocks prices are assumed to be unchanged and as same as the adjusted closed stock price on the previous trading day. Furthermore, the early-closure days are considered the same as the regular trading days. Finally, the polarities of reactions (comments) are assumed to be unchanged in the selected time duration.

Table 4.9: The Individual Investors' Reactions to Verizon.

Date	Name	Content	User Label	Up Votes	Down Votes	Abuse Votes	Replies
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
10/9/2020	Kenlong	Looking ⋯	BULLISH	2	0	0	3
10/9/2020	rlee	Hey⋯	()	2	0	2	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
10/7/2020	Daniel	I ⋯	BEARISH	2	6	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

4.5 Financial Sentiment (BERTs) and Financial Sentiment Index

After the preprocessing, Table 4.9 shows the data format of the prepared individual investors' reactions. The financial sentiment of individual investors is required to calculate the financial sentiment index. Based on Table 4.9, the feature "User Label" is a straightforward source to get the financial sentiment, and it could show the investors' attitude toward the performance of Verizon Communications Inc. and its stock. Because the "User Label" is created by who left the comment, all the reaction contents may not necessarily have a "User Label". To get enough financial sentiments, the BERT could be employed to classify the reaction contents, which are without "User Label", into suitable classes (e.g., Bullish, Bearish, Neutral).

Due to the anonymous comments mechanism, there might be many irrelevant or inappropriate comments in the data sets. To remove these comments, we applied Tukey's fences (Tukey, 1977). There are a total of 3901 of reactions after removing "abused reactions" by the Tukey's fences (with a coefficient value of 1.5). Among all the responses, there are only 509 reactions having labels. The splitting strategy of the training set and the testing set in here needs to match the split way of how it is done in the data set of the logarithmic returns (90% training and 10% testing). Finally, there are 198 reactions for training and 311 reactions for testing. Figure 4.2 shows the distribution of the reactions' labels in the training data.

There are four types of BERT available. To pick the best one for prediction purposes, 20% of the training data is selected to create a validation set. The model will be saved during the training process if it has the highest weighted F_1-score in the validation set. Cross-validation is not employed because we consider the reactions are time-dependent.

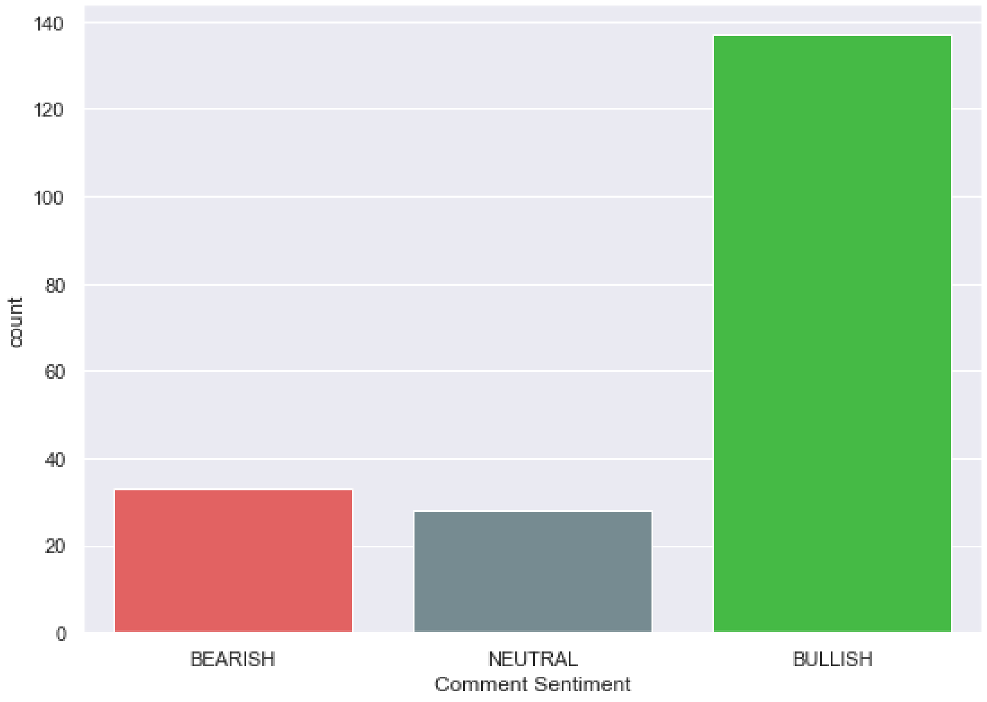


Figure 4.2: The Distribution of Reactions' Label in the Training Data.

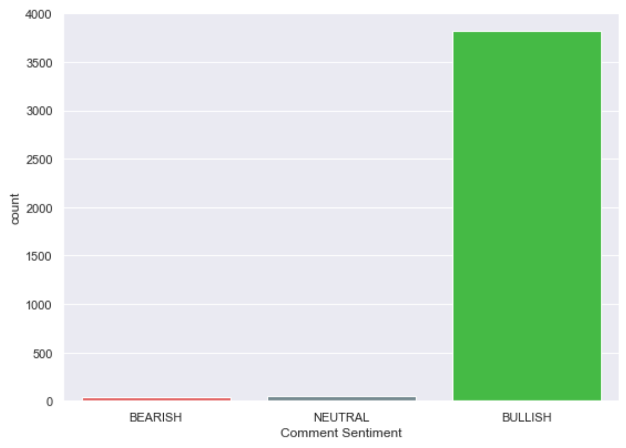
From Table 4.10, the BERT large model (uncased) and BERT large model (cased) have the same best weighted F_1-Score in the validation set.

Table 4.10: The Performance of BERTs in the Validation Set.

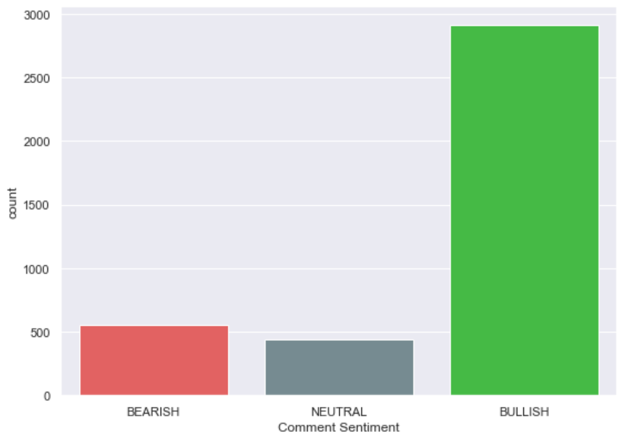
Name	Weighted F_1 Score	Time
BERT base model (cased)	0.763	151.06s
BERT large model (cased)	0.815	327.05s
BERT base model (uncased)	0.708	160.48s
BERT large model (uncased)	0.815	340.93s

Note. The number in red color stands for the best performance under the specific metric in the table.

After comparing the distribution of the predicted sentiments by the BERT large model (uncased) and the BERT large model (cased) in Figure 4.3, the BERT large model (uncased) is the more



(a) BERT Large Model (Cased).



(b) BERT Large Model (Uncased).

Figure 4.3: The Distribution of Predicted Reactions Label by BERTs.

appropriate, and it is selected to do the financial sentiment estimation and prediction.

By checking Table 4.11, the BERT large model (uncased) has the best performance in the testing data set. It means the model selection strategy worked well. Additionally, Table 4.12 gives some example labels which are predicted by the BERT large model (uncased) in the test data.

Table 4.11: The Performance of BERTs in the Test Set.

Name	Weighted F_1 Score
BERT base model (cased)	0.726
BERT large model (cased)	0.547
BERT base model (uncased)	0.575
BERT large model (uncased)	0.764

Note. The number in red color stands for the best performance under the specific metric in the table.

The reactions labels construct the sentiment indices. The reactions with the label “NEUTRAL” are assumed to contribute nothing to the logarithmic returns. Further, there is no obvious evidence to show that the negative reaction and the positive reaction have the same effect on the log return of a stock, so the sentiment indices are considered separately as two components (the bullish sentiments and the bearish sentiments) at a specific time. For one reaction, the up-vote counts are the times of agreement, and the down-vote counts are considered the times of disagreement.

The sentiment indices construction strategy on a specific date are as follows:

Table 4.12: Examples of the Predicted Labels by the BERT Large Model (Uncased) in the Test Data.

Content	Predicted Label By BERT
poor vz	BEARISH
This stock is going down a lot more. Yahoo move is desperate and a bad idea.	BEARISH
VZ PE ratio under 7. This baby is going to \$55+.	BULLISH
Verizon coming back	BULLISH
Anybody here?	NEUTRAL
Maybe this stock needs to be discussed on Reddit rather than Yahoo???	NEUTRAL

- For all reactions has label "BULLISH" on that date,
  $B_{Bullish} = \#\ of\ upvote_{Bullish} - \#\ of\ downvote_{Bullish} + \#\ of\ reactions_{Bullish}.$
- For all reactions has label "BEARISH" on that date,
  $B_{Bearish} = \#\ of\ upvote_{Bearish} - \#\ of\ downvote_{Bearish} + \#\ of\ reactions_{Bearish}.$

After the conversion process, a part of the financial sentiment indices (Verizon) are shown in Table 4.13.

Table 4.13: The Financial Sentiment Index of Verizon From 7/13/2016 to 8/24/2021.

Time	$B_{Bullish}$(VZ)	$B_{Bearish}$(VZ)
7/14/2016	0	0
7/15/2016	0	0
$\vdots$	$\vdots$	$\vdots$
8/23/2021	29	6
8/24/2021	51	5

4.6 Prepared Data Illustration

To construct the data set for the selected technology companies, we apply similar processes from Section 4.3 to Section 4.5 to Alphabet Inc. The prepared data of Verizon and Alphabet could be combined in an extensive data set to be the data set of the technology industry. The example of the technology industry can be shown in Table 4.14. Furthermore, the other three sectors (Finance, Pharmaceutical, Retail) are considered in this experimental study. Additionally, the data for these industries are also prepared, and the structure of these data sets are similar to Table 4.14. Therefore, this dissertation does not show the redundant visualization of the data collection and process parts for these three data sets.

4.7 Data Application and Model Training Strategy

There are four industries considered in this dissertation, and the technology section (the data set is shown in Table 4.14) is chosen as the example to explain the data application and model training strategy for training proportion 90% (1160 days) and testing proportion 10% (128 days).

4.7.1 Null Model

A null model is provided in each section to test if building a model is necessary. The null model for the example industry is designed in Algorithm 2.

Algorithm 2: The training process of Null Model.

```
1  Set Predictions =[ ]; Set $Performance\_Test$ = [ ];
2  For $k$ in $range(0, ..., 127)$ do
3      Potential_Orders = [ ];
4      Take the simple arithematic average of the training data set as one-step forward
         prediction, and the training data set has the indices from $[1, 1160 + k]$;
5      Add the prediction into Predictions;
6  end
7  Evaluate the predictions performance by the predefined metrics;
8  For $criterion \in Evaluation\_Metrics\_Set\_Testing$ do
9      Calculate $criterion\_test_{current}$;
10     Append $criterion\_test_{current}$ to Performance_Test;
11 end
12 return Predictions; Performance_Test;
```

4.7.2 ARIMA Models

The training process of ARIMA models for the predefined testing proportion 10% (128 days) can be described in Algorithm 3. Besides, the target data is "Log Return (GOOG)" in Table 4.14.

Table 4.14: Prepared Data for the Technology Industry From 7/14/2016 to 8/24/2021 (the Logarithmic Returns Are the Original Logarithmic Returns × 1000).

Date	Log Return (VZ)	$B_{Bullish}$ (VZ)	$B_{Bearish}$ (VZ)	Analyst Recommendation (VZ)	Log Return (GOOG)	$B_{Bullish}$ (GOOG)	$B_{Bearish}$ (GOOG)	Analyst Recommendation (GOOG)	Dow Jones Industrial Average	NASDAQ 100	S&P 500
7/14/2016	-2.861	0	0	0	5.522	0	0	0	5.245	7.283	6.706
7/15/2016	0	0	0	0	-1.527	7	0	0	-9.294	5.478	-1.450
7/18/2016	1.611	5	7	0	1.917	6	0	0	2.379	8.907	6.482
7/19/2016	-4.121	22	10	0	4.324	0	0	0	-1.436	1.400	-3.556
7/20/2016	-1.437	0	0	0	5.723	1	0	0	4.261	1.939	1.165
7/21/2016	-4.505	1	0	0	-3.460	1	0	0	-3.619	-4.193	-2.188
7/22/2016	1.310	16	9	0	5.549	2	0	0	4.544	2.891	4.091
7/25/2016	-4.108	81	14	0	-4.007	2	0	0	-3.016	-4.198	-2.993
7/26/2016	-1.915	5	3	0	-1.827	1	0	0	3.227	-1.045	1.324
7/27/2016	9.262	1	2	0	4.527	0	1	0	-1.199	-8.553	6.564
7/28/2016	-8.350	6	0	1	5.566	15	1	0	1.605	-8.568	3.932
7/29/2016	9.976	13	0	1	3.021	11	1	1	1.630	-1.307	1.866
8/1/2016	-16.560	16	10	1	5.306	4	1	1	-1.271	-1.506	5.442
8/2/2016	-9.217	1	0	1	-2.345	3	0	1	-6.382	-4.943	-7.774
8/3/2016	-1.854	0	0	1	2.733	2	0	1	3.129	2.249	3.199
8/4/2016	0.556	2	2	1	-2.033	8	0	1	0.213	-0.161	2.000
8/5/2016	-5.392	0	0	1	13.661	2	2	1	8.567	10.387	9.942
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
8/2/2021	-0.359	25	-5	0	5.667	0	0	1	-1.845	-2.789	0.249
8/3/2021	-0.538	44	-2	0	2.134	0	0	1	8.170	7.955	6.515
8/4/2021	-6.481	59	-3	0	-1.847	0	0	1	-4.643	-9.261	1.458
8/5/2021	-1.265	61	0	0	6.678	0	0	1	5.987	7.775	6.493
8/6/2021	-1.448	53	-4	0	0.701	0	0	1	1.674	4.106	-4.772
8/9/2021	-1.813	32	0	0	7.025	0	0	1	-0.940	-3.034	1.571
8/10/2021	6.510	13	6	0	0.685	0	0	1	0.992	4.628	-5.269
8/11/2021	1.621	31	10	0	-2.952	0	0	1	1.275	6.228	-1.717
8/12/2021	1.259	38	0	0	5.071	0	0	1	4.138	0.419	4.066
8/13/2021	4.125	11	0	0	0.119	0	0	1	1.606	0.437	3.156
8/16/2021	1.967	15	0	0	3.678	0	0	1	1.105	3.093	0.270
8/17/2021	3.566	3	0	0	-11.703	0	0	1	-5.573	-7.951	-9.152
8/18/2021	-12.722	10	4	0	-5.335	0	0	1	-10.814	-10.882	-9.706
8/19/2021	-1.804	11	0	0	2.512	0	0	1	1.256	-1.906	5.103
8/20/2021	2.705	4	0	0	11.076	0	0	1	8.109	6.455	10.571
8/23/2021	-1.803	29	6	0	19.057	0	0	1	8.488	6.121	14.496
8/24/2021	-7.426	51	5	0	9.164	0	0	1	1.495	0.864	2.925

Note. All the values are adjusted close values which are adjusted for both dividends and splits.

Algorithm 3: The Training Process of ARIMA Models.

```
Input: The entire column (“Log Return (GOOG)”) of Table 4.14.
Input: Testing proportion 10% (128 data points), maximum AR order = 7, maximum
       MA order = 7.
Input: $criterion\_test_{current}$.
Output: Predictions & Evaluation metrics.

1  Set Predictions =[ ]; Set $Performance\_Test$ = [ ];
2  For $k$ in $range(0, ..., 127)$ do
3      Potential_Orders = [ ];
4      For criterion ∈ [AIC, BIC, HQIC, AICC] do
5          Train the ARIMA model with maximum AR order = 7, maximum MA order = 7
            with the training data set where the indices are from $[1, 1160 + k]$;
6          Select the best AR order $p'$ and MA order $q'$ by criterion;
7          Add $(p', q')$ into Potential_Orders;
8      end
9      Select the majority orders $(p_{(best)}, q_{(best)})$ form Potential_Orders;
10     Refit a ARIMA model with $(p_{(best)}, q_{(best)})$ and the training data is the training data
        set where the indices are from $[1, 1160 + k]$;
11     Make one-step forward prediction by the refit model;
12     Add the prediction into Predictions;
13 end
14 Evaluate the predictions performance by the predefined metrics;
15 For criterion ∈ Evaluation_Metrics_Set_Testing do
16     Calculate $criterion\_test_{current}$;
17     Append $criterion\_test_{current}$ to Performance_Test;
18 end
19 return Predictions; Performance_Test;
```

4.7.3 ARIMAX Models

In the ARIMAX models, the two columns “Log Return (GOOG)” and “Log Return (VZ)” of Table 4.14 are enrolled. The performance evaluation is similar to the process of the ARIMA models, and the predicted values with “Log Return (GOOG)” is the target values. Further, the log return (VZ) is considered an external variable.

4.7.4 VARMA Models

Compared with the ARIMA models, VARMA models consider two columns "Log Return (GOOG)" and "Log Return (VZ)" of Table 4.14. Although VARMA models would output a two-dimensional vector with names "Log Return (GOOG)" and "Log Return (VZ)". But only the predicted values with "Log Return (GOOG)" are considered in the performance evaluation to make different models comparable. In addition, the detailed evaluation process could be found in Algorithm 4.

4.7.5 Univariate LSTM Models

The target data is the column "Log Return (GOOG)" in Table 4.14. The training and performance evaluation process for univariate LSTM models is the concrete of Algorithm 5 with some additional evaluation steps with following hyper-parameters setup. In the first part of the, the predefined hyper-parameters are as follows:

- The optimizer is "AdaBelief" (Adapting Stepsizes by the Belief in Observed Gradients) with $\epsilon = 10^{-12}$, *weight decay* $= 1.2 \times 10^{-6}$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$.
- The learning rate is 0.01.
- The validation size is 128 (same as the test size).
- The initial train epochs size is 100.
- The initial fine tune epochs size is 10.
- The $val\_stop_{criterion}$ is RMSE.

In the second part, the grid search is applied to select the most appropriate neural networks structure from the following combinations:

- The input window size $\in \{10, 20, ..., 90, 100\}$.
- The number of hidden layers $\in \{1, 2, 3\}$.
- The number of cells in each hidden layer $\in \{1, 2, 3, 4\}$.

In addition, all the models which contain an LSTM component will have the same hyper-parameters setup.

Algorithm 4: The Training Process of VARMA Models

```
Input: The entire columns “Log Return (GOOG)” and “Log Return (VZ)” of Table 4.14.
Input: Testing proportion 10% (128 data points), maximum AR order = 3, maximum
       MA order = 2.
Input: $criterion\_test_{current}$.
Output: Predictions & Evaluation metrics.

1  Predictions =[ ];
2  For $k$ in $range(0, ..., 127)$ do
3      Potential_Orders = [ ];
4      For criterion ∈ [AIC, BIC, AICC] do
5          Train the VARMA model with maximum AR order = 3, maximum MA order = 2
             with the training data set where the indices are from $[1, 1160 + k]$;
6          Select the best AR order $p'$ and MA order $q'$ by criterion;
7          Add $(p', q')$ into Potential_Orders;
8      end
9      Select the majority orders $(p_{(best)}, q_{(best)})$ form Potential_Orders;
10     Refit a VARMA model with $(p_{(best)}, q_{(best)})$ and the training data is the training data
         set where the indices are from $[1, 1160 + k]$;
11     Make one-step forward prediction by the refit model;
12     Add the prediction with label (“Log Return (GOOG)”) into Predictions;
13 end
14 Evaluate the predictions performance by the predefined metrics;
15 For criterion ∈ Evaluation_Metrics_Set_Testing do
16     Calculate $criterion\_test_{current}$;
17     Append $criterion\_test_{current}$ to Performance_Test;
18 end
19 return Predictions; Performance_Test;
```

4.7.6 ARIMA-LSTM Model

The training and performance evaluation process for ARIMA-LSTM Model is the concrete of Algorithm 6. In addition, the setup for the hyperparameters is the same as the one in the subsection Univariate LSTM Models. Only the column “Log Return (GOOG)” of Table 4.14 is considered.

4.7.7 Multivariate LSTM Model

The training and performance evaluation process for multivariate LSTM models is the concrete of Algorithm 5 with some additional evaluation steps with the hyperparameters setup in the subsection Univariate LSTM Models. The two columns "Log Return (GOOG)" and "Log Return (VZ)" of Table 4.14 are considered. The "Log Return (GOOG)" is the target column. For **multivariate LSTM models with the market indices**, the other three columns of "Dow Jones Industrial Average", "NASDAQ 100", and "S&P 500" of Table 4.14 are also contained, the target column is still "Log Return (GOOG)".

4.7.8 VARMA-LSTM Model

The training and performance evaluation process for VARMA-LSTM Model is the concrete of Algorithm 6. In addition, the setup for the hyperparameters is the same as the subsection Univariate LSTM Models. The two columns "Log Return (GOOG)" and "Log Return (VZ)" of Table 4.14 are contained. For **VARMA-LSTM models with the market indices**, the other three columns of "Dow Jones Industrial Average", "NASDAQ 100", and "S&P 500" of Table 4.14 are contained as the input with the residual of the VARMA model. Also, the target column is still "Log Return (GOOG)".

4.7.9 BERTFSIE-Multivariate-LSTM Models

The training and performance evaluation process for BERTFSIE-Multivariate-LSTM Models is the concrete of Algorithm 5. In addition, the setup for the hyperparameters is the same as the one in the subsection Univariate LSTM Models. The two columns "Log Return (GOOG)" and "Log Return (VZ)" of Table 4.14 are contained. The other four columns "$B_{Bullish}$ (VZ)", "$B_{Bearish}$ (VZ)", "$B_{Bullish}$ (GOOG)" and "$B_{Bearish}$ (GOOG)" are combined with being inputted into the neural network part. The structure of this realized BERTFSIE-Multivariate-LSTM Models could be shown in Figure 4.4.

For **BERTFSIE-Multivariate-LSTM Model with the market indices**, the other three columns of "Dow Jones Industrial Average", "NASDAQ 100", and "S&P 500" of Table 4.14 are contained. Also, the target column is still "Log Return (GOOG)".

4.7.10 BERTFSIE-VARMA-LSTM Models

The training and performance evaluation process for BERTFSIE-VARMA-LSTM Model is the concrete of Algorithm 6. In addition, the setup for the hyperparameters is the same as the one in the

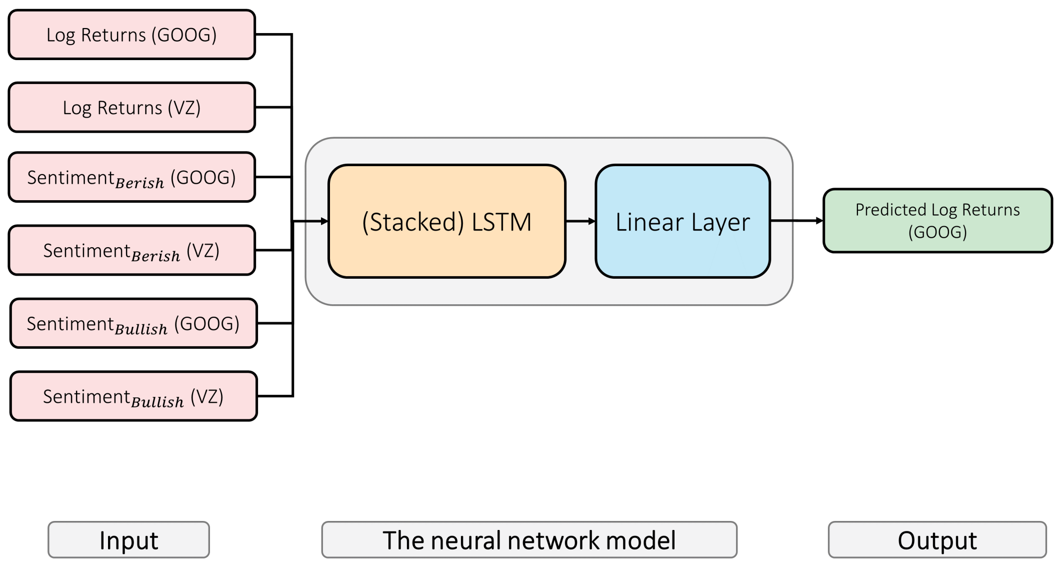

Log Returns (GOOG)
Log Returns (VZ)
Sentiment$_{Berish}$ (GOOG)
Sentiment$_{Berish}$ (VZ)
Sentiment$_{Bullish}$ (GOOG)
Sentiment$_{Bullish}$ (VZ)
(Stacked) LSTM
Linear Layer
Predicted Log Returns (GOOG)
Input
The neural network model
Output

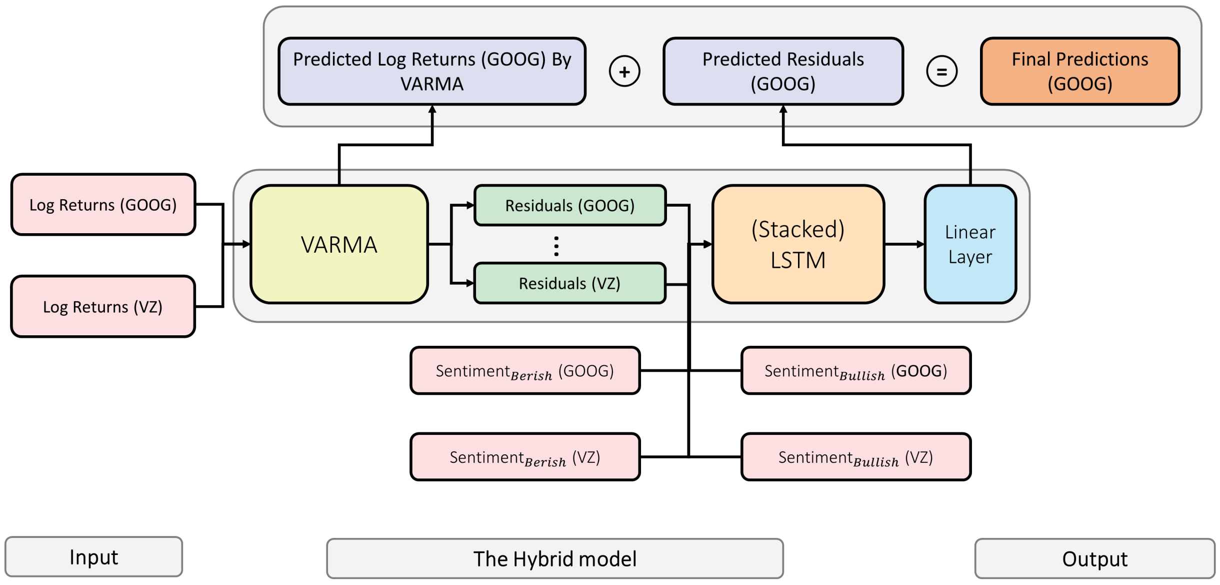
Figure 4.4: The Structure of BERT-based Financial Sentiment Index Enhanced (BERTFSIE) multivariate LSTM Model.

subsection Univariate LSTM Models. The two columns "Log Return (GOOG)" and "Log Return (VZ)" of Table 4.14 are contained. The other four columns "$B_{Bullish}$ (VZ)", "$B_{Bearish}$ (VZ)", "$B_{Bullish}$ (GOOG)" and "$B_{Bearish}$ (GOOG)" are combined with the residual of the VARMA model, and they are inputted into the neural network part. The structure of this realized BERTFSIE-VARMA-LSTM model could be shown in Figure 4.5.

For **BERTFSIE-VARMA-LSTM models with the market indices**, the other three columns of "Dow Jones Industrial Average", "NASDAQ 100", and "S&P 500" of Table 4.14 are contained as the input with the residual of the VARMA model. Also, the target column is still "Log Return (GOOG)".

4.7.11 ARIE-Multivariate-LSTM Models & ARIE-VARMA-LSTM Models

The process of ARIE-Multivariate-LSTM models (ARIE-VARMA-LSTM Models) and ARIE-Multivariate-LSTM models with the market indices (ARIE-VARMA-LSTM models with the market indices) are similar to the process in BERTFSIE-Multivariate-LSTM Models (BERTFSIE-VARMA-LSTM). The only difference is that the BERT-based financial sentiment indices are replaced by the analyst recommendation indices ("Analyst Recommendation (GOOG)" and "Analyst Recommendation (VZ)" in Table 4.14).



Predicted Log Returns (GOOG) By VARMA
+
Predicted Residuals (GOOG)
=
Final Predictions (GOOG)
Log Returns (GOOG)
Log Returns (VZ)
VARMA
Residuals (GOOG)
Residuals (VZ)
(Stacked) LSTM
Linear Layer
Sentiment$_{Berish}$ (GOOG)
Sentiment$_{Bullish}$ (GOOG)
Sentiment$_{Berish}$ (VZ)
Sentiment$_{Bullish}$ (VZ)
Input
The Hybrid model
Output

Figure 4.5: The Structure of the BERT-Based Financial Sentiment Index Enhanced VARMA-LSTM Model.

4.8 Results

The results of the model performance comparison for the different test proportions are displayed and discussed in this chapter. There are three test proportions (10%, 20%, and 30%), and for each test proportion, there are four industries covered (Finance, Pharmaceutical, Retail, Technology).

4.8.1 Finance Industry

Testing Proportion 10%**.** Two companies are mainly considered (Citigroup Inc. and The Goldman Sachs Group, Inc.). The target variable is the log-returns of "Citigroup". Furthermore, there are six blocks of the models' performance comparison shown in Table 4.15.

The first block is the performance of the null model. We can see that advanced models are necessary because the null model has a poor performance in all the considered metrics.

In the second block, the performance of the four models is compared. They are all statistical time series models. The ARIMA model has the best RMSE, the VARMA (Market) has the most petite MAE, and the ARIMAX model outperforms the other three models in Weighted F_1-Score and Weighted AUC. In addition, VARMA (Market) stands for the VARMA model that considers the two financial companies and the three market indices.

In the third block, the performance of three LSTM models is described. Multi-LSTM (Market) stands for the multivariate LSTM model that considers the two financial companies and market indices. Furthermore, Multi-LSTM (Market) model has the smallest RMSE and MAE in the block. Uni-LSTM, standing for the uni-variate LSTM model, has the best score among the models in the second block in terms of Weighted F_1-Score, and Weighted AUC.

Hybrid models' performance is in comparison in the fourth block. VARMA-LSTM has the best performances in all considered evaluation metrics except Weighted F_1-Score in the block. VARMA-LSTM (Market) beats the other models in Weighted F_1-Score in the fourth block. Besides, VARMA-LSTM (Market) is a short form of a VARMA-LSTM model with market indices.

The fifth block discusses the empirical forecast ability of BERTFSIE types of models. Further, in the block, the BERTFSIE-VARMA-LSTM model's performance surpasses the other models in Weighted F_1-Score, and Weighted AUC. ERTFSIE-VARMA-LSTM (Market) is the best model in terms of RMSE and MAE.

In the last block, the performance of ARIE types of models is shown. ARIE-Multi LSTM (Market), representing the ARIE-VARMA-LSTM model with the market indices, beats the other three models with respect to Weighted F_1-Score. ARIE-VARMA-LSTM is the top when comparing the prediction ability in RMSE, MAE, and Weighted AUC.

When it comes to the entire Table 4.15, VARMA-LSTM has the best score in RMSE and MAE when it is compared to all the other models in the table. Additionally, BERTFSIE-VARMA-LSTM beats the rest models in Weighted F_1-Score, and Weighted AUC.

Testing Proportion 20%. All setups for this section are the same as the section with testing proportion 10%, except for the testing proportion and validation proportion are changed to 20%.

Same as what we find in Table 4.15, we can see that advanced models are helpful. The model which has the best performance in the second block of Table 4.16 is similar to the models in Table 4.15. However, in the third block of Table 4.16, the best performance models are inverse. The Uni-LSTM has the best performance in RMSE and MAE. The VARMA-LSTM model is still the best in terms of RMSE, MAE, and Weighted AUC in the fourth block. But, the performance of selected VARMA-LSTM (Market) in Table 4.16 does not have a competitive performance to VARMA-LSTM anymore. The BERTFSIE-type models in the fifth block show that the BERTFSIE-VARMA-LSTM

Table 4.15: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Citigroup Inc. (Finance Industry; Testing Proportion 10%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	15.612	12.498	0.333	0.367
ARIMA	**15.902**	12.722	0.453	0.485
ARIMAX	21.658	16.857	**0.562**	**0.555**
VARMA	16.067	12.969	0.421	0.409
VARMA (Market)	16.061	**12.678**	0.469	0.460
Uni-LSTM	15.496	12.251	**0.594**	**0.595**
Multi-LSTM	14.904	12.348	0.469	0.482
Multi-LSTM (Market)	**14.848**	**11.929**	0.499	0.510
ARIMA-LSTM	16.381	12.831	0.543	0.554
VARMA-LSTM	**13.809**	**11.240**	0.577	**0.592**
VARMA-LSTM (Market)	13.916	11.360	**0.586**	0.583
BERTFSIE-Multi LSTM	14.838	11.745	0.549	0.542
BERTFSIE-Multi LSTM (Market)	19.699	16.080	0.493	0.508
BERTFSIE-VARMA-LSTM	16.402	12.412	**0.629**	**0.619**
BERTFSIE-VARMA-LSTM (Market)	**14.162**	**11.349**	0.488	0.612
ARIE-Multi LSTM	16.030	13.113	0.429	0.426
ARIE-Multi LSTM (Market)	17.096	13.246	**0.551**	0.574
ARIE-VARMA-LSTM	**14.229**	**11.250**	0.495	**0.599**
ARIE-VARMA-LSTM (Market)	17.078	13.302	0.494	0.517

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values × 1000.

(Market) model performance is still the best in RMSE and MAE. But the BERTFSIE-Multi LSTM model outperforms the other models in the block in terms of the other selected metrics. For the model performances in the final block of Table 4.16, it is different from the one in Table 4.15. The ARIE-Multi LSTM (Market) model has the lowest value in RMSE and MAE. Moreover, the ARIE-VARMA-LSTM (Market) model has the best performance in Weighted F_1-Score, and the ARIE-Multi LSTM model has the highest value in Weighted AUC.

For the overall performances in Table 4.16, the ARIE-Multi LSTM (Market) model is the best in RMSE and MAE. The BERTFSIE-Multi LSTM model has the best performance in Weighted F_1-Score and Weighted AUC.

Testing Proportion 30%. In the current section, testing and validation proportions have been increased to 30%.

Table 4.16: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Citigroup Inc. (Finance Industry; Testing Proportion 20%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	20.300	15.142	0.357	0.431
ARIMA	20.413	15.170	0.485	**0.528**
ARIMAX	26.976	20.670	**0.515**	0.517
VARMA	**20.280**	15.175	0.503	0.505
VARMA (Market)	20.541	**15.076**	0.493	0.509
Uni-LSTM	**20.820**	**15.380**	0.476	0.456
Multi-LSTM	22.291	16.668	0.505	0.542
Multi-LSTM (Market)	21.315	16.154	**0.535**	**0.544**
ARIMA-LSTM	23.434	18.018	**0.520**	0.491
VARMA-LSTM	**21.751**	**16.394**	0.510	**0.534**
VARMA-LSTM (Market)	21.975	16.801	0.485	0.450
BERTFSIE-Multi LSTM	23.940	18.220	**0.603**	**0.679**
BERTFSIE-Multi LSTM (Market)	21.635	16.249	0.515	0.531
BERTFSIE-VARMA-LSTM	22.102	16.670	0.587	0.615
BERTFSIE-VARMA-LSTM (Market)	**21.341**	**15.732**	0.538	0.543
ARIE-Multi LSTM	21.554	15.757	0.523	**0.553**
ARIE-Multi LSTM (Market)	**20.063**	**14.848**	0.556	0.539
ARIE-VARMA-LSTM	21.427	16.052	0.521	0.521
ARIE-VARMA-LSTM (Market)	21.490	16.281	**0.562**	0.521

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values × 1000.

In the first block of Table 4.17, null model does not perform well. The comparison of the employed advanced models is meaningful.

In the second block of Table 4.17, under the selected metrics, the best model among the four statistical time series models are almost identical to the situation in the first block of Table 4.16, except for the VARMA model outperforms the others in MAE.

The uni-variate LSTM model has the best RMSE and MAE compared with the other models in the third block. Furthermore, the Multivariate LSTM model has the best performance with regard to the other two selected metrics.

For the hybrid models in the fourth block, the ARIMA-LSTM model beat the other two models in terms of RMSE, MAE, and Weighted F_1-Score. The VARMA-LSTM (Market) model has the highest Weighted AUC.

The BERTFSIE-VARMA-LSTM model is the best in the fifth block in all the selected metrics. Finally, for the entire table, ARIMA-LSTM has the lowest RMSE. In addition, BERTFSIE-VARMA-LSTM has the best score in MAE, Weighted F_1-Score and Weighted AUC.

Table 4.17: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Citigroup Inc. (Finance Industry; Testing Proportion 30%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	36.158	23.516	0.343	0.439
ARIMA	37.357	24.100	0.482	**0.509**
ARIMAX	47.922	30.570	**0.499**	0.497
VARMA	**36.925**	**24.046**	0.496	0.489
VARMA (Market)	37.654	24.361	0.476	0.473
Uni-LSTM	**21.579**	**16.271**	0.492	0.474
Multi-LSTM	22.385	16.514	**0.500**	**0.494**
Multi-LSTM (Market)	21.864	16.442	0.461	0.444
ARIMA-LSTM	**21.017**	**15.883**	**0.507**	0.499
VARMA-LSTM	23.644	18.396	0.444	0.431
VARMA-LSTM (Market)	21.320	15.956	0.495	**0.510**
BERTFSIE-Multi LSTM	21.305	16.022	0.518	0.507
BERTFSIE-Multi LSTM (Market)	21.831	16.524	0.511	0.506
BERTFSIE-VARMA-LSTM	**21.074**	**15.368**	**0.558**	**0.575**
BERTFSIE-VARMA-LSTM (Market)	23.919	18.044	0.520	0.520
ARIE-Multi LSTM	22.431	17.370	0.527	**0.533**
ARIE-Multi LSTM (Market)	**21.560**	16.281	0.469	0.475
ARIE-VARMA-LSTM	21.983	16.914	**0.528**	0.527
ARIE-VARMA-LSTM (Market)	21.580	**16.070**	0.469	0.464

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values × 1000.

4.8.2 Pharmaceutical Industry

Testing Proportion 10%. Bristol-Myers Squibb Company and AbbVie Inc. are mainly considered in the pharmaceutical industry. The target variable is the log-returns of AbbVie Inc. In addition, six blocks of the models' performance comparison are shown in Table 4.18.

The performance of the null model is the same as the performance of the ARIMA model. That is, ARIMA fails to model this data set. However, applying advanced models is still helpful when comparing the null mode with other models' performance.

In the second block, the ARIMA model has the best RMSE and MAE, the VARMA (Market) has the most significant Weighted F_1-Score, and Weighted AUC scores.

The performance of LSTM models is discussed in the third block. Multivariate LSTM (Multi-LSTM) has the best Weighted F_1-Score and Weighted AUC scores. Further, uni-variate LSTM (uni-variate) outperforms the other LSTM model in RMSE and MAE.

In the fourth block (Hybrid models), VARMA-LSTM has the smallest values in RMSE and MAE. For the other three evaluation metrics, the ARIMA-LSTM model is the best.

In the next block (fourth block), the BERTFSIE-VARMA-LSTM model achieves the top performance in terms of RMSE, MAE, Weighted F_1-Score, and Weighted AUC.

In the last block, ARIE-VARMA-LSTM (Market) has the top performance in RMSE and MAE. The ARIE-VARMA-LSTM has the most considerable Weighted F_1-Score compared with the other three ARIE-type models. The ARIE-Multi LSTM model outperforms the rest models concerning the Weighted AUC score.

When it comes to the entire Table 4.18, BERTFSIE-VARMA-LSTM has the best RMSE, MAE, Weighted F_1-Score and Weighted AUC.

Testing Proportion 20%. The results shown in the first block of Table 4.19 are still the same as the performance of the ARIMA model, and we have the same conclusion that the ARIMA model fails but considering other models are still meaningful.

In the second block of Table 4.19, the best performing model agrees with the situation in the same block of Table 4.18, that is, ARIMA has the best RMSE and MAE. In addition, VARMA (Market) has the highest Weighted F_1-Score and Weighted AUC. However, in the third block of Table 4.19, the Multi-LSTM model is the best model concerning all the selected metrics. The VARMA-LSTM (Market) model outperforms the other two hybrid models in the fourth block in Weighted F_1-Score, and Weighted AUC. Furthermore, the BERTFSIE-VARMA-LSTM model has the highest value in Weighted F_1-Score, and the BERTFSIE-Multi LSTM (Market) model beats the other three models in the fifth part of Table 4.19. In the final block of Table 4.19, the ARIE-VARMA-LSTM model has the best performance in all selected metrics except for the metric RMSE.

For the overall performances in Table 4.19, ARIE type models have the best RMSE and MAE. In addition, BERTFSIE type models have the most significant value in Weighted F_1-Score, Weighted AUC.

Table 4.18: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Bristol-Myers Squibb Company. (Pharmaceutical Industry; Testing Proportion 10%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	10.482	7.728	0.389	0.380
ARIMA	**10.482**	**7.728**	0.389	0.380
ARIMAX	11.559	8.784	0.449	0.444
VARMA	10.615	7.885	0.447	0.420
VARMA (Market)	10.599	7.827	**0.517**	**0.520**
Uni-LSTM	**11.729**	**8.967**	0.491	0.434
Multi-LSTM	15.065	10.923	**0.502**	**0.505**
Multi-LSTM (Market)	12.204	9.435	0.431	0.437
ARIMA-LSTM	11.129	8.762	**0.537**	**0.527**
VARMA-LSTM	**10.984**	**8.142**	0.516	0.506
VARMA-LSTM (Market)	25.284	13.949	0.466	0.434
BERTFSIE-Multi LSTM	10.865	7.916	0.464	0.457
BERTFSIE-Multi LSTM (Market)	10.661	8.220	0.528	0.476
BERTFSIE-VARMA-LSTM	**10.461**	**7.696**	**0.578**	**0.611**
BERTFSIE-VARMA-LSTM (Market)	10.595	7.918	0.511	0.511
ARIE-Multi LSTM	11.156	8.580	0.511	**0.540**
ARIE-Multi LSTM (Market)	13.080	9.858	0.537	0.517
ARIE-VARMA-LSTM	16.066	9.370	**0.554**	0.515
ARIE-VARMA-LSTM (Market)	**11.005**	**8.370**	0.494	0.465

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values × 1000.

Testing Proportion 30%. The same results and conclusions are made in this part after we checked the values in the first block of Table 4.20. The ARIMA fails to handle the model, but the other models are still functional.

The best performing models in the second and the third block of Table 4.20 is the same as the models in Table 4.19. The ARIMA model has the best performance in RMSE and MAE. Additionally, the VARMA (Market) model outperforms the other two metrics: Weighted F_1-Score, Weighted AUC in the second block. In the third block, the multivariate LSTM model is the best.

The ARIMA-LSTM beats all the other models in the four metrics in the fourth block of Table 4.20.

For the models' performance comparison in the fifth block, the BERTFSIE-Multi LSTM model has the lowest value in RMSE, and the BERTFSIE-VARMA-LSTM model has the largest Weighted

Table 4.19: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Bristol-Myers Squibb Company. (Pharmaceutical Industry; Testing Proportion 20%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	12.874	9.218	0.394	0.477
ARIMA	**12.874**	**9.218**	0.394	0.477
ARIMAX	14.720	10.660	0.452	0.425
VARMA	12.899	9.331	0.502	0.484
VARMA (Market)	13.096	9.406	**0.509**	**0.505**
Uni-LSTM	16.338	12.250	0.486	0.450
Multi-LSTM	**13.231**	**9.367**	**0.516**	**0.499**
Multi-LSTM (Market)	14.654	11.269	0.505	0.497
ARIMA-LSTM	**13.860**	**10.189**	0.404	0.378
VARMA-LSTM	15.887	12.101	0.473	0.475
VARMA-LSTM (Market)	14.157	10.664	**0.550**	**0.552**
BERTFSIE-Multi LSTM	17.473	13.824	0.511	0.532
BERTFSIE-Multi LSTM (Market)	**14.183**	**10.688**	0.554	**0.558**
BERTFSIE-VARMA-LSTM	16.695	11.767	0.539	0.536
BERTFSIE-VARMA-LSTM (Market)	16.537	12.475	**0.557**	0.546
ARIE-Multi LSTM	**12.807**	9.253	0.462	0.503
ARIE-Multi LSTM (Market)	16.161	11.428	0.498	0.517
ARIE-VARMA-LSTM	13.196	**9.184**	**0.510**	**0.538**
ARIE-VARMA-LSTM (Market)	16.026	12.114	0.487	0.492

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values × 1000.

AUC score. In addition, the BERTFSIE-VARMA-LSTM (Market) model outperforms the other three models in MAE, Weighted F_1-Score.

In the last block, the ARIE-Multi LSTM (Market) model is the best one in terms of all the evaluation metrics.

For the overall performance in the table, the ARIE-Multi LSTM (Market) model has the best RMSE and MAE. The BERTFSIE-VARMA-LSTM (Market) model has the largest Weighted F_1-Score, and the BERTFSIE-VARMA-LSTM model has the best Weighted AUC score.

4.8.3 Retail Industry

Testing Proportion 10%**.** For the retail industry, the stock price log-returns of Costco Wholesale Corporation and Target Corporation are included, and the focused variable is the log-

Table 4.20: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Bristol-Myers Squibb Company. (Pharmaceutical Industry; Testing Proportion 30%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	16.403	11.237	0.373	0.460
ARIMA	**16.403**	**11.237**	0.373	0.460
ARIMAX	19.017	12.974	0.459	0.436
VARMA	16.578	11.430	0.461	0.444
VARMA (Market)	16.922	11.645	**0.494**	**0.474**
Uni-LSTM	17.412	13.366	0.473	0.472
Multi-LSTM	**15.667**	**11.203**	**0.540**	**0.541**
Multi-LSTM (Market)	20.284	13.882	0.514	0.538
ARIMA-LSTM	**16.555**	**11.806**	**0.509**	**0.522**
VARMA-LSTM	17.334	12.755	0.470	0.486
VARMA-LSTM (Market)	19.582	15.293	0.508	0.510
BERTFSIE-Multi LSTM	**16.263**	12.293	0.520	0.494
BERTFSIE-Multi LSTM (Market)	20.207	15.679	0.446	0.468
BERTFSIE-VARMA-LSTM	17.304	13.533	0.496	**0.543**
BERTFSIE-VARMA-LSTM (Market)	17.563	**11.747**	**0.545**	0.511
ARIE-Multi LSTM	15.873	12.114	0.515	0.504
ARIE-Multi LSTM (Market)	**14.417**	**10.331**	**0.530**	**0.535**
ARIE-VARMA-LSTM	14.706	10.757	0.455	0.438
ARIE-VARMA-LSTM (Market)	18.191	13.910	0.511	0.513

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values × 1000.

returns of "Target". Similarly, there are six blocks of the models' performance shown in Table 4.21.

By comparing the other models' performance with the performance of the null model, advanced models are applicable.

In the second section, the VARMA model has the highest Weighted F_1-Score when compared with the other three statistical models. Also, VARMA (Market) model performance is the best with regard to RMSE, MAE, and Weighted AUC.

In the third block, the performance of three LSTM models is shown. Uni-LSTM model has the lowest value in RMSE and MAE in the union. Moreover, the Multi-LSTM model is the top one when the evaluation metrics are Weighted F_1-Score, or Weighted AUC.

Hybrid models' performance is compared in the fourth section. VARMA-LSTM has the best performance in all considered evaluation metrics except for Weighted AUC in this part, and ARIMA-LSTM outperforms all the other two hybrid models in Weighted AUC score.

BERTFSIE-Multi LSTM (Market) has the smallest RMSE and MAE values in the following block (fifth block). Besides, BERTFSIE-VARMA-LSTM beats the rest of the models in terms of Weighted F_1-Score. In addition, BERTFSIE-VARMA-LSTM (Market) is the best in Weighted AUC compared with the rest models in the block.

The ARIE-VARMA-LSTM model has the best performance in RMSE and MAE in the final block. On the other hand, ARIE-VARMA-LSTM (Market) model has the highest score in Weighted F_1-Score, and Weighted AUC.

When we are talking about the overall performance in Table 4.21, the BERTFSIE-Multi LSTM (Market) model outperforms all the other models in RMSE and MAE. Moreover, ARIE-VARMA-LSTM (Market) model has the largest value in Weighted F_1-Score, and BERTFSIE-VARMA-LSTM (Market) has the largest Weighted AUC.

Testing Proportion 20%**.** The performance of the null model shows that the application of the other advanced models is meaningful.

In the second block of Table 4.22, the VARMA model has the smallest value in RMSE, and the ARIMAX model has the highest Weighted F_1-Score. The top performance model in the second block of Table 4.22 has a significant difference from the leading model in the same block of Table 4.21 Additionally, in the third block of Table 4.22, the Multi-LSTM (Market) model has the best score in RMSE, MAE, and Weighted AUC. The VARMA-LSTM (Market) model is the best in the fourth block in all the selected metrics. Furthermore, the BERTFSIE-VARMA-LSTM (Market) model outperforms all the models in the fifth block of Table 4.22. In the last part of Table 4.22, the ARIE-Multi LSTM (Market) model has the most considerable value in the metrics: Weighted F_1-Score and Weighted AUC. Additionally, the ARIE-VARMA-LSTM model has the lowest value in RMSE and MAE.

For the overall performances in Table 4.22, ARIE type models have the highest Weighted F_1-Score and Weighted AUC. In addition, BERTFSIE type models have the best RMSE and MAE.

Testing Proportion 30%**.** The advanced models are concluded to be functional when their performances are compared with the null model in the first block of Table 4.23.

Table 4.21: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Costco Wholesale Corporation. (Retail Industry; Testing Proportion 10%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	10.646	8.689	0.414	0.494
ARIMA	10.728	8.790	0.500	0.484
ARIMAX	10.878	8.706	0.513	0.492
VARMA	10.741	8.736	**0.550**	0.516
VARMA (Market)	**10.666**	**8.698**	0.533	**0.527**
Uni-LSTM	**10.632**	**8.858**	0.488	0.461
Multi-LSTM	11.568	9.306	**0.540**	**0.563**
Multi-LSTM (Market)	11.311	9.152	0.461	0.482
ARIMA-LSTM	11.052	9.122	0.531	**0.549**
VARMA-LSTM	**10.711**	**8.640**	**0.559**	0.538
VARMA-LSTM (Market)	11.486	9.370	0.481	0.435
BERTFSIE-Multi LSTM	11.197	8.817	0.530	0.513
BERTFSIE-Multi LSTM (Market)	**10.262**	**8.368**	0.530	0.557
BERTFSIE-VARMA-LSTM	10.633	8.541	**0.542**	0.522
BERTFSIE-VARMA-LSTM (Market)	11.448	8.841	0.537	**0.581**
ARIE-Multi LSTM	10.849	8.745	0.470	0.452
ARIE-Multi LSTM (Market)	11.321	8.977	0.520	0.486
ARIE-VARMA-LSTM	**10.767**	**8.600**	0.535	0.545
ARIE-VARMA-LSTM (Market)	11.103	8.912	**0.578**	**0.554**

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values × 1000.

In the second block of Table 4.23, the VARMA (Market) model's forecasting performance is the top in all the selected metrics, except for the RMSE. In addition, the ARIMA model has the best performance in RMSE.

For the model performance comparison of univariate and multivariate LSTM models, the Multi-LSTM (Market) model has the best Weighted F_1-Score and Weighted AUC. Additionally, the Multi-LSTM model has the smallest RMSE and MAE.

The VARMA-LSTM model beats the other two models in RMSE and MAE for the hybrid models. Moreover, the VARMA-LSTM (Market) model outperforms the other models in Weighted F_1-Score, and Weighted AUC.

For the BERTFSIE type of models, BERTFSIE-VARMA-LSTM (Market) has the most significant Weighted F_1-Score and Weighted AUC. Besides, BERTFSIE-Multi LSTM has the lowest

Table 4.22: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Costco Wholesale Corporation. (Retail Industry; Testing Proportion 20%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	11.253	8.754	0.398	0.487
ARIMA	11.330	8.880	0.495	0.472
ARIMAX	12.122	9.295	**0.537**	0.505
VARMA	**11.313**	8.847	0.512	0.491
VARMA (Market)	11.330	**8.843**	0.502	**0.521**
Uni-LSTM	14.180	11.117	**0.526**	0.493
Multi-LSTM	13.979	10.769	0.498	0.505
Multi-LSTM (Market)	**11.982**	**9.177**	0.489	**0.512**
ARIMA-LSTM	12.188	9.575	0.517	0.504
VARMA-LSTM	13.915	10.974	0.503	0.462
VARMA-LSTM (Market)	**12.037**	**9.155**	**0.540**	**0.526**
BERTFSIE-Multi LSTM	13.750	11.036	0.511	0.520
BERTFSIE-Multi LSTM (Market)	11.299	8.732	0.529	0.555
BERTFSIE-VARMA-LSTM	12.155	9.329	0.518	0.520
BERTFSIE-VARMA-LSTM (Market)	**11.156**	**8.518**	**0.536**	**0.574**
ARIE-Multi LSTM	13.294	10.412	0.540	0.519
ARIE-Multi LSTM (Market)	14.118	10.978	**0.584**	**0.608**
ARIE-VARMA-LSTM	**11.434**	**8.788**	0.452	0.536
ARIE-VARMA-LSTM (Market)	12.818	9.731	0.568	0.551

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values $\times$ 1000.

RMSE and MAE compared with the other models.

The ARIE-VARMA-LSTM (Market) model has the best performance in RMSE and MAE in the final block. Furthermore, the ARIE-Multi LSTM (Market) model has the most outstanding Weighted F_1-Score. The ARIE-Multi LSTM model has the highest value for the last metric weighted AUC.

For the models' performance across the block in Table 4.23, the multivariate LSTM model has the best RMSE, and the BERTFSIE-Multi LSTM model has the lowest MAE. For the models' performance in Weighted F_1-Score and Weighted AUC, the BERTFSIE-VARMA-LSTM (Market) model outperforms the rest models in Table 4.23.

Table 4.23: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Costco Wholesale Corporation. (Retail Industry; Testing Proportion 30%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	16.045	10.769	0.377	0.491
ARIMA	**16.042**	10.819	0.486	0.488
ARIMAX	18.584	11.987	0.499	0.469
VARMA	16.046	10.807	0.505	0.502
VARMA (Market)	16.101	**10.780**	**0.534**	**0.548**
Uni-LSTM	13.880	10.831	0.498	0.482
Multi-LSTM	**12.073**	**9.398**	0.475	0.472
Multi-LSTM (Market)	13.380	10.746	**0.516**	**0.506**
ARIMA-LSTM	13.276	10.221	0.442	0.422
VARMA-LSTM	**12.726**	**9.969**	0.476	0.453
VARMA-LSTM (Market)	13.264	10.427	**0.520**	**0.509**
BERTFSIE-Multi LSTM	**12.241**	**9.346**	0.550	0.530
BERTFSIE-Multi LSTM (Market)	12.206	9.436	0.464	0.453
BERTFSIE-VARMA-LSTM	16.337	12.887	0.494	0.472
BERTFSIE-VARMA-LSTM (Market)	13.863	10.871	**0.575**	**0.580**
ARIE-Multi LSTM	13.753	10.802	0.508	**0.515**
ARIE-Multi LSTM (Market)	13.568	10.408	**0.518**	0.508
ARIE-VARMA-LSTM	13.049	10.144	0.453	0.437
ARIE-VARMA-LSTM (Market)	**12.884**	**10.133**	0.459	0.478

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values × 1000.

4.8.4 Technology Industry

Testing Proportion 10%**.** The log-returns of Alphabet Inc. and Technology Verizon Communications Inc. are mainly involved in the technology industry, and the target variable is the log-returns of Alphabet Inc. In addition, six blocks of the models' performance comparison are shown in Table 4.24.

The null model's performance shown in Table 4.24 shows that all the considered models are functional.

In the second part of Table 4.24, the VARMA (Market) model has the best RMSE and Weighted F_1-Score. Furthermore, the VARMA has the best MAE and Weighted AUC.

In the third block, the Uni-LSTM model has the best performance in terms of Weighted F_1-Score, and Weighted AUC scores. In addition, the Multi-LSTM (Market) has the best RMSE and

MAE. Besides, the VARMA-LSTM model beats all the other models in the evaluation metrics in the fourth block.

In the block of BERTFSIE-type models (fifth block), the BERTFSIE-VARMA-LSTM model has the smallest RMSE and MAE. Additionally, the BERTFSIE-VARMA-LSTM (Market) model has the best Weighted F_1-Score and Weighted AUC.

At the bottom of Table 4.24, the ARIE-VARMA-LSTM (Market) is the top model when compared with the other models in terms of RMSE, MAE, and Weighted AUC. Besides, the ARIE-Multi LSTM model is the best in Weighted F_1-Score.

When it comes to the whole Table 4.24, the BERTFSIE-VARMA-LSTM model has the best overall performance in RMSE and MAE. Moreover, the BERTFSIE-VARMA-LSTM (Market) model beats all the other models in Weighted F_1-Score and Weighted AUC.

Table 4.24: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Alphabet Inc. (Technology Industry; Testing Proportion 10%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	13.203	9.563	0.471	0.504
ARIMA	13.260	9.580	0.552	0.565
ARIMAX	13.799	10.078	0.515	0.443
VARMA	13.256	**9.542**	0.523	**0.560**
VARMA (Market)	**13.157**	9.557	**0.579**	0.553
Uni-LSTM	14.584	11.298	**0.566**	**0.542**
Multi-LSTM	13.283	10.222	0.497	0.489
Multi-LSTM (Market)	**12.888**	**9.614**	0.433	0.422
ARIMA-LSTM	14.319	11.314	0.444	0.420
VARMA-LSTM	**12.435**	**9.142**	**0.602**	**0.584**
VARMA-LSTM (Market)	13.227	10.178	0.447	0.387
BERTFSIE-Multi LSTM	13.538	10.243	0.531	0.493
BERTFSIE-Multi LSTM (Market)	15.226	11.577	0.557	0.583
BERTFSIE-VARMA-LSTM	**12.344**	**9.037**	0.579	0.560
BERTFSIE-VARMA-LSTM (Market)	14.120	10.698	**0.616**	**0.609**
ARIE-Multi LSTM	15.163	11.182	**0.590**	0.548
ARIE-Multi LSTM (Market)	**12.883**	**9.647**	0.557	**0.594**
ARIE-VARMA-LSTM	13.536	10.214	0.502	0.479
ARIE-VARMA-LSTM (Market)	13.135	9.756	0.544	0.559

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values $\times$ 1000.

Testing Proportion 20%. When the validation and the testing proportion increased from 10% to 20%, the top performance models in the second block of Table 4.25 are different from those in Table 4.24. Besides, the ARIMA model has the best score in RMSE, Weighted F_1-Score, and Weighted AUC in the second block of Table 4.25. In the third block of Table 4.25, the Multi-LSTM (Market) mod has the best performance in all the selected metrics.

The VARMA-LSTM model achieves the best Weighted F_1-Score when compared with the other two hybrid models in the fourth block of Table 4.25.

For the BERTFSIE-type models, the BERTFSIE-VARMA-LSTM (Market) model performance is the best when compared in RMSE and MAE in the fifth block of Table 4.25. However, the BERTFSIE-VARMA-LSTM model outperforms the other models in Weighted F_1-Score and Weighted AUC in the fifth block of Table 4.25.

For the performances rank in the final block of Table 4.25, the ARIE-VARMA-LSTM (Market) models have the best performance uniformly.

For the overall performances in the entire Table 4.25, the BERTFSIE-VARMA-LSTM (Market) model has the best performance in RMSE, MAE. In addition, BERTFSIE-VARMA-LSTM has the highest Weighted F_1-Score and Weighted AUC.

Testing Proportion 30%. In the second block of Table 4.26, the VARMA model has the best RMSE, MAE, and Weighted AUC. The univariate model (ARIMA) has the most considerable Weighted F_1-Score.

For the performance of LSTM models in the third block, each model has been ranked top under different evaluation metrics. The univariate LSTM model has the lowest MAE, and the Multi-LSTM (Market) beats the other two models in RMSE and Weighted AUC score. Furthermore, the Multi-LSTM has the best Weighted F_1-Score.

In the fourth block of Table 4.26, the VARMA-LSTM (Market) model outperforms the other two models in all the selected metrics except for the Weighted F_1-Score.

For the performance of BERTFSIE-type models, the BERTFSIE-VARMA-LSTM (Market) model beats all the models in the fifth block under all selected evaluation metrics.

The ARIE-VARMA-LSTM (Market) is the best under MAE, Weighted F_1-Score, and Weighted AUC in the last block. In addition, the ARIE-Multi LSTM (Market) model has the smallest RMSE.

For the entire Table 4.26, the BERTFSIE-VARMA-LSTM (Market) model is the best one under RMSE, MAE, Weighted F_1-Score and Weighted AUC when compared with all the other models.

Table 4.25: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Alphabet Inc. (Technology Industry; Testing Proportion 20%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	16.619	11.924	0.435	0.506
ARIMA	**16.702**	11.973	**0.544**	**0.550**
ARIMAX	17.214	12.452	0.521	0.463
VARMA	16.757	**11.902**	0.504	0.523
VARMA (Market)	16.912	12.111	0.494	0.518
Uni-LSTM	19.549	14.312	0.485	0.467
Multi-LSTM	19.322	13.544	0.467	0.447
Multi-LSTM (Market)	**17.401**	**12.880**	**0.514**	**0.528**
ARIMA-LSTM	17.757	12.970	0.441	0.448
VARMA-LSTM	**17.313**	**12.963**	0.522	**0.529**
VARMA-LSTM (Market)	17.957	13.180	**0.538**	0.502
BERTFSIE-Multi LSTM	17.259	12.772	0.513	0.518
BERTFSIE-Multi LSTM (Market)	17.734	13.385	0.509	0.511
BERTFSIE-VARMA-LSTM	16.628	11.388	**0.558**	**0.552**
BERTFSIE-VARMA-LSTM (Market)	**16.053**	**11.226**	0.543	0.546
ARIE-Multi LSTM	17.479	12.983	0.493	0.482
ARIE-Multi LSTM (Market)	19.808	15.181	0.508	0.518
ARIE-VARMA-LSTM	20.757	15.784	0.509	0.501
ARIE-VARMA-LSTM (Market)	**16.145**	**11.637**	**0.556**	**0.535**

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values $\times$ 1000.

Table 4.26: Models Performance in Forecasting the Stock Price Log-Returns and the Movement Directions of Alphabet Inc. (Technology Industry; Testing Proportion 30%)

Model	RMSE	MAE	Weighted F_1-Score	Weighted AUC
Null Model	21.606	14.550	0.429	0.485
ARIMA	21.557	14.684	**0.538**	0.542
ARIMAX	23.199	15.463	0.479	0.449
VARMA	**21.240**	**14.320**	0.527	**0.567**
VARMA (Market)	21.289	14.447	0.531	0.556
Uni-LSTM	17.229	**12.654**	0.496	0.504
Multi-LSTM	18.202	13.512	**0.522**	0.507
Multi-LSTM (Market)	**17.044**	12.712	0.511	**0.532**
ARIMA-LSTM	17.821	13.395	0.511	0.523
VARMA-LSTM	17.418	13.377	**0.540**	0.528
VARMA-LSTM (Market)	**17.270**	**12.804**	0.539	**0.551**
BERTFSIE-Multi LSTM	17.115	12.415	0.525	0.511
BERTFSIE-Multi LSTM (Market)	19.359	15.196	0.485	0.488
BERTFSIE-VARMA-LSTM	17.840	13.743	0.534	0.525
BERTFSIE-VARMA-LSTM (Market)	**16.466**	**11.925**	**0.598**	**0.596**
ARIE-Multi LSTM	17.073	12.877	0.492	0.490
ARIE-Multi LSTM (Market)	**16.801**	12.357	0.466	0.455
ARIE-VARMA-LSTM	17.159	12.508	0.521	0.457
ARIE-VARMA-LSTM (Market)	16.864	**12.278**	**0.539**	**0.515**

Note. The boldface **numbers** stand for the best performance under the specific metric in that block. The numbers in red color stand for the best performance under the specific metric in the table. The RMSE and MAE are their original values $\times$ 1000.

CHAPTER 5

CONCLUSIONS

In financial asset price prediction topics, statistical models are widely applied. Recently, machine learning models (e.g., recurrent neural networks models, support vector regressions, etc.) are also investigated because of the flexible assumptions. But, the flexible assumptions of machine learning models have pros and cons. Hence, some hybrid models are designed to show their superiority. However, the limited information provided by only the stock prices is not enough. Therefore, more information should be considered in the prediction process of a model. With the increase of computational power and the specialized processor (e.g., Graphics processing unit), the technological innovations of Natural language processing (NLP) models (e.g., BERT, Generative Pre-trained Transformers) could help us to use more unstructured data in the financial time series forecasting problems. Further, the market indices might also be supportive of improving the prediction ability of quantitative methods. Therefore, there are mainly five types of comparisons are made in terms of the prediction ability of the selected stock log returns in four selected sections (Technology, Finance, Pharmaceutical, and Retail) with three testing and validation proportion (10%, 20% and 30%):

- The prediction ability of univariate and multivariate statistical models and machine learning models are compared.
- The influence of the market indices as covariates would be tested by comparing with the models without considering these market indices.
- The performance of the BERTFSIE-type models is compared with the original models.
- The forecast ability of BERTFSIE-type models is compared with ARIE-type models.

5.1 Conclusions on the Models Performance in Finance Industry

The model with the lowest RMSE and MAE among all the models tends to be the model consisting of an LSTM component. However, there is no specific model with the best RMSE and MAE compared with all the other models, and their performance is not consistent when the testing proportion changes. When the evaluation metric is Weighted F_1-Score, and Weighted AUC., the

BERTFSIE type model has the best performance, which is much better than other types of models. When it comes to performance stability, the statistical time series models are relatively stable for all the metrics.

Among statistical time series models, multivariate models tend to have a better performance in all the selected metrics than univariate models. The same conclusion is still valid when the machine learning models are discussed.

The effect of considering the selected market indices as covariates is not apparent, and it varies when the different models and metrics are under consideration.

The influence of the individual investors' reactions to the stocks' logarithmic returns is significant. Even if the testing proportion has been changed, the BERTFSIE type of model still has the best performance in the Weighted F_1-Score, and Weighted AUC score.

Because the analysts' recommendations are too sparse, generally speaking, the overall performance is not satisfactory in this section.

5.2 Conclusions on the Models Performance in Pharmaceutical Industry

With the increasing testing proportion (including validation), ARIE models tend to have the best RMSE and MAE. Additionally, the BERTFSIE type model has the best performance in the Weighted F_1-Score, and Weighted AUC score. Similar to what we found in the financial industry, traditional statistical time series models have a stable performance when compared with the other type of models.

For the traditional statistical models, the VARMA (Market) model tends to have the best performance in Weighted F_1-Score, and Weighted AUC score compared with the other three models in different testing proportions. However, for RMSE and MAE metrics, the ARIMA model has the best value among the statistical time series models. Furthermore, considering the market indices as covariates is helpful in the statistical model block. The individual investors' reactions having a significant influence on the stocks' logarithmic returns are supported by our experiment study.

Besides, different from what we found in the financial industry, the ARIE-type models have competitive performance and are considered helpful.

5.3 Conclusions on the Models Performance in Retail Industry

Even if the testing proportion increased, the statistical time series model has a stable performance compared with other models. Furthermore, multivariate models tend to perform better in all the selected four evaluation metrics.

Taking the selected market indices as covariates is helpful for this industry section. We can see that the best-performing model in different metrics tends to be the models that include the market indices.

Same as we found in the previous two industries, the individual investors' reactions significantly impact the stocks' logarithmic returns when we employ Weighted AUC and MAE as evaluation metrics. In addition, the ARIE type of model is considered to be helpful.

5.4 Conclusions on the Models Performance in Technology Industry

With the testing and validation proportion increasing, the statistical model with the best performance is not the same when the evaluation metrics are different. The superiority is shown in the multivariate statistical model with a higher testing proportion. Besides, considering market indices as covariates might not be helpful.

However, for the other type of models, when the testing proportion increases, the models tend to perform better by including the selected market indices.

When it comes to BERTFSIE-type models, they are the best in RMSE, MAE, Weighted F_1-Score, and Weighted AUC among the selected models. Therefore, the individual investors' reactions significantly impact the stock log returns in the selected data set.

Finally, the influence of analysts' recommendations on the stocks' logarithmic returns is insignificant, and the experiment study supports this conclusion. The sparsity of analysts' recommendations may cause no competitive performance of ARIE-type models.

5.5 Current Challenges

The current hyper-parameters tuning method (e.g., Algorithm 5 to Algorithm 8) is testing all the possible combinations in the defined searching space, and it is a time consuming and inefficient way to do hyper-parameters tuning. Therefore, some efficient hyper-parameters tuning method is needed. The running time summary can be found from Table A.1 to Table A.3.

A traditional way to tune the hyper-parameters is to search the hyper-parameter space by the best selected metric score in cross-validation. Cross-validation is the backtest when sequential data are modeled. Furthermore, the model is easily over-fitted when the neural network is complicated in small data size. Some problems need to be solved during the cross-validation of sequential data modeling:

- There is more than one weighted scheme that could be employed in evaluating the models' performance of Algorithm 5 and Algorithm 6. Moreover, we applied the equal-weighted scheme in this dissertation. Another machine learning model or a statistical model might be employed to learn the weights instead of predefining the weights.
- All the neural network method's performance is sensitive to the predefined hyperparameters (e.g., the size of the validation set).
- The criterion is not comparable when the size of the validation set is different.

5.6 Future Work

There are many aspects that could be improved in the future, and they are stated as follows:

- All the LSTM can be changed to Gated Recurrent Unit (GRU), and there are more combinations between the BERT-based financial sentiment index and other models. In addition, there are also more combinations between analysts' recommendation index enhanced models and other models.
- The MSE is selected as the criteria to update the parameters in the neural network models, and a weighted AUC score is chosen to be the metric for preventing over-fitting. Furthermore, the different criteria could be applied to the Algorithm 5 and Algorithm 6. Also, a weighted metric could be considered in this stage.
- The same wights of linear and non-linear components in the hybrid model are straightforward, but they may not be the best. Statistical models could be employed to select the weights (e.g., linear regression models).
- The same wights are chosen to take advantage of the bright side of different evaluation metrics in the model selection stage (Algorithm 8). Additionally, different selecting weights strategies could be employed.
- Transformer-based machine learning models could be employed to forecast time-series data (Wu et al., 2020).
- More models (e.g., XGboost, SVM) could be involved in future work.

- Some more efficient hyper-parameter optimization methods could be considered. For example, the hyperparameters of the neural network methods could be optimized by Bayesian Optimization and Hyperband (BOHB) in Neural Network Intelligence (NNI) toolkit.

APPENDIX A

TRAINING & PERFORMANCE EVALUATION ALGORITHMS

Algorithm 5: Pseudocode for the Backtest.

```
Input: Entire Dataset; Test Proportion; Validation Proportion; Validation_Stop Proportion; $Input_{Length}$;
       $Size_{layers}$; $Size_{hidden\ states}$; Optimizer; Evaluation_Metric_Training;
       Evaluation_Metrics_Set_Validation; Evaluation_Metrics_Set_Testing.
// Evaluation_Metric_Training is MSE in this dissertation
Output: The Results of Evaluation Metrics in Testing Dateset.
1  Split Entire Dataset into Entire_Training_Dataset and Testing_Set by Test Proportion;
2  Split Entire_Training_Dataset into Training_Dataset and Validation_Set by Validation Proportion;
3  Set $Performance\_Validation = [\,]$;
4  Set $Performance\_Test = [\,]$;
5  For $i \in \{1, ..., Input_{Length}\}$ do
6      For $j \in \{1, ..., Size_{layers}\}$ do
7          For $l \in \{1, ..., Size_{hidden\ states}\}$ do
8              Set $Pred\_Validation\_Set = [\,]$;
9              Set $Pred\_Testing\_Set = [\,]$;
10             For $k$ in $range(1, ..., Size_{test})$ do
11                 Plug in Algorithm 7;
12             end
13             Set $Perf\_validation = [i, j, l]$;
14             For $criterion \in Evaluation\_Metrics\_Set\_Validation$ do
15                 Calculate $criterion_{current}$;
16                 Append $criterion_{current}$ to $Perf\_validation$;
17             end
18             Append $Perf\_validation$ to $Performance\_Validation$;
19         end
20     end
21 end
22 Plug in Algorithm 8;
23 For $criterion \in Evaluation\_Metrics\_Set\_Testing$ do
24     Calculate $criterion\_test_{current}$;
25     Append $criterion\_test_{current}$ to Performance_Test;
26 end
27 return Performance_Test;
```

Algorithm 6: Pseudocode for the Backtest of Hybrid Models.

```
Input: Entire Dataset; Test Proportion; Validation Proportion; Validation_Stop Proportion; $Input_{Length}$;
       $Size_{layers}$; $Size_{hidden\ states}$; Optimizer; Evaluation_Metric_Training;
       Evaluation_Metrics_Set_Validation; Evaluation_Metrics_Set_Testing.
// Evaluation_Metric_Training is MSE in this dissertation
Output: The Results of Evaluation Metrics in Testing Dateset.
1  Split Entire Dataset into Entire_Training_Dataset and Testing_Set by Test Proportion;
2  Split Entire_Training_Dataset into Training_Dataset and Validation_Set by Validation Proportion;
3  Set $Performance\_Validation = [\,]$;
4  Set $Performance\_Test = [\,]$;
5  For $i \in \{1, ..., Input_{Length}\}$ do
6      For $j \in \{1, ..., Size_{layers}\}$ do
7          For $l \in \{1, ..., Size_{hidden\ states}\}$ do
8              Set $Pred\_Validation\_Set = [\,]$;
9              Set $Pred\_Testing\_Set = [\,]$;
10             For $k$ in $range(1, ..., Size_{test})$ do
11                 Potential_Orders = [ ];
12                 For $criterion \in Criteria_{set}$ do
13                     Train the time series models with possible orders combination by the training data;
14                     Select the best orders by $criterion$ and add the best orders into Potential_Orders;
15                 end
16                 Select the best majority orders combination form Potential_Orders;
17                 Refit the time series model with the majority best orders with the training data set;
18                 Make one-step forward prediction by the refit model;
19                 Make the predictions with the target label (To get $pred\_validation\_TS_k$ and
                     $pred\_test\_TS_k$);
20                 Replace the Corresponding features in Training_Dataset by the residual values from the
                     time series model;
21                 Plug in Algorithm 7;
22                 $Pred\_Testing\_Set[pred\_test_k] += pred\_test\_TS_k$;
23                 $Pred\_Validation\_Set[pred\_validation_k] += pred\_validation\_TS_k$
24             end
25             Set $Perf\_validation = [i, j, l]$;
26             For $criterion \in Evaluation\_Metrics\_Set\_Validation$ do
27                 Calculate $criterion_{current}$;
28                 Append $criterion_{current}$ to $Perf\_validation$;
29             end
30             Append $Perf\_validation$ to $Performance\_Validation$;
31         end
32     end
33 end
34 Plug in Algorithm 8;
35 For $criterion \in Evaluation\_Metrics\_Set\_Testing$ do
36     Calculate $criterion\_test_{current}$;
37     Append $criterion\_test_{current}$ to Performance_Test;
38 end
39 return Performance_Test;
```

Algorithm 7: Pseudocode for the Backtest (Stage 2: Early Stopping for Sequential Data to Prevent Over-Fitting).

```
Input: Predefined hyperparameters of a neural network model; Training Data; Validation_Stop Proportion;
       $Pred\_Validation\_Set$; $Pred\_Testing\_Set$; Prediction Round $k$.
Output: The Well-trained Model; $Pred\_Validation\_Set$; $Pred\_Testing\_Set$;
Split Training_Dataset into Training_Set and Validation_Stop_Set by Validation_Stop Proportion;
if  $k == 1$ then
    $Validation\_Stop_{criteria} = [\,]$ and set $d = 0$;
    Standardize Training_Set by the standardization parameters of Training_Set;
    while $d < Epoch$ do
        Update the neural network model's parameters;
        $val\_stop_{criterion}$ into $Validation\_Stop_{criteria}$;
        Set $Predicted\_Validation\_Stop\_Set = [\,]$
        For $p \in \{1, ..., Size_{Validation\_Stop\_Set}\}$ do
            Standardize the input data for prediction to Validation_Stop_Set by the standardization
              parameters of Training_Set;
            Do prediction by the current model (to get $pred\_val\_stop\_p$);
            Apply inverse transformation to the prediction value by the standardization parameters of
              Training_Set;
            Append $pred\_val\_stop\_p$ to $Predicted\_Validation\_Stop\_Set$;
        end
        Calculate $val\_stop_{criterion}$ based on $Predicted\_Validation\_Stop\_Set$ and $Validation\_Stop\_Set$ ;
        Append $val\_stop_{criterion}$ to $Validation\_Stop_{criteria}$;
        if $val\_stop_{criterion} == best(Validation\_Stop_{criteria})$ then
            Save current model as $model_{best}$;
            $Epoch+ = 1$;
        end
        $d+ = 1$;
    end
    Load $model_{best}$;
    Standardize the input data for prediction to Validation Data and Testing Data individually by the
      training standardization parameters;
    Use $model_{best}$ to do prediction (To get $pred\_validation_k$ and $pred\_test_k$);
    Apply inverse transformation to the predictions value by the standardization parameters of
      Training_Set;
    Append $pred\_validation_k$ to $Pred\_Validation\_Set$; Append $pred\_test_k$ to $Pred\_Testing\_Set$.
else
    Load $model_{best}$ and set $o = 0$;
    Standardize Training_Set by the standardization parameters of Training_Set;
    while $o < Fine\ tune\ Epoch$ do
        Update the neural network model's parameters;
        $val\_stop_{criterion}$ into $Validation\_Stop_{criteria}$;
        Set $Predicted\_Validation\_Stop\_Set = [\,]$
        For $p \in \{1, ..., Size_{Validation\_Stop\_Set}\}$ do
            Standardize the input data for prediction to Validation_Stop_Set by the standardization
              parameters of Training_Set;
            Do prediction by the current model (to get $pred\_val\_stop\_p$);
            Apply inverse transformation to the prediction value by the standardization parameters of
              Training_Set;
            Append $pred\_val\_stop\_p$ to $Predicted\_Validation\_Stop\_Set$;
        end
        Calculate $val\_stop_{criterion}$ based on $Predicted\_Validation\_Stop\_Set$ and $Validation\_Stop\_Set$ ;
        Append $val\_stop_{criterion}$ to $Validation\_Stop_{criteria}$;
        if $val\_stop_{criterion} == best(Validation\_Stop_{criteria})$ then
            Save current model as $model_{best}$;
            $Fine\ tune\ Epoch+ = 1$;
        end
        $o+ = 1$;
    end
    Load $model_{best}$;
    Standardize the input data for prediction to Validation Data and Testing Data individually by the
      training standardization parameters;
    Use $model_{best}$ to do prediction (To get $pred\_validation_k$ and $pred\_test_k$);
    Apply inverse transformation to the predictions value by the standardization parameters of
      Training_Set;
    Append $pred\_validation_k$ to $Pred\_Validation\_Set$; Append $pred\_test_k$ to $Pred\_Testing\_Set$.
end
return The Well-trained Model; $Pred\_Validation\_Set$; $Pred\_Testing\_Set$;
```

Algorithm 8: Pseudocode for Hyperparameters Selection (Score Voting).

```
Input: $Performance\_Validation$; $Evaluation\_Metrics\_Set\_Validation$.
Output: Selected Hyperparameters.
Set $Score\_Votes = [\,]$;
For $criterion \in Evaluation\_Metrics\_Set\_Validation$ do
    Score the items in $Performance\_Validation$ by their Ranking of $criterion$ (To get $Score_{criterion}$);
    // Best performance item should have the largest score
    For $item \in Performance\_Validation$ do
        if $item[index] \notin Score\_Votes$ then
            Append [item[index], $Score_{criterion}$] to $Score\_Votes$;
        else
            $Score\_Votes[index][Score_{criterion}] = Score\_Votes[index][Score_{criterion}] + Score_{criterion}$ ;
        end
    end
end
return $item$ with the largest $Score_{criterion}$ in $Performance\_Validation$;
```

Table A.1: Running Time Comparison of Models in Forecasting Price Log-Returns and Directions of the Selected Stock. (Testing Proportion: 10%)

Model	Finance	Pharmaceutical	Retail	Technology
ARIMA	1.286	**0.113**	0.308	**0.287**
ARIMAX	**0.240**	0.170	**0.288**	0.538
VARMA	0.527	0.399	0.378	0.395
VARMA (Market)	13.449	16.639	16.515	16.420
Uni-LSTM	11.801	13.400	10.419	12.208
Multi-LSTM	12.052	11.381	9.845	11.251
Multi-LSTM (Market)	**11.216**	**10.863**	**9.658**	**10.327**
ARIMA-LSTM	57.920	23.371	26.817	28.881
VARMA-LSTM	35.942	22.623	22.499	23.150
VARMA-LSTM (Market)	**32.668**	**22.563**	**20.306**	**22.313**
BERTFSIE-Multi LSTM	10.724	10.690	8.565	9.762
BERTFSIE-Multi LSTM (Market)	**9.732**	**10.171**	**7.934**	**8.726**
BERTFSIE-VARMA-LSTM	37.140	26.770	16.745	16.950
BERTFSIE-VARMA-LSTM (Market)	37.266	28.550	16.100	25.507
ARIE-Multi LSTM	**12.278**	12.994	**10.362**	**12.133**
ARIE-Multi LSTM (Market)	13.032	**11.009**	10.847	12.598
ARIE-VARMA-LSTM	43.270	23.089	24.320	24.625
ARIE-VARMA-LSTM (Market)	32.093	26.107	22.659	26.625

Note. The boldface **numbers** stand for the least running time in that block. The numbers in red color stand for the least running time in the table. The unit is hour.

Table A.2: Running Time Comparison of Models in Forecasting Price Log-Returns and Directions of the Selected Stock. (Testing Proportion: 20%)

Model	Finance	Pharmaceutical	Retail	Technology
ARIMA	2.559	**0.202**	**0.504**	**0.912**
ARIMAX	**0.449**	0.330	0.598	1.325
VARMA	1.282	0.930	0.853	0.980
VARMA (Market)	24.413	28.168	28.394	27.013
Uni-LSTM	**15.607**	**16.779**	16.796	**15.953**
Multi-LSTM	17.980	18.892	15.358	16.058
Multi-LSTM (Market)	17.264	16.968	**14.589**	16.168
ARIMA-LSTM	**36.986**	**24.409**	37.238	**35.426**
VARMA-LSTM	52.645	41.928	42.492	43.252
VARMA-LSTM (Market)	53.136	42.895	**35.791**	45.383
BERTFSIE-Multi LSTM	17.953	18.777	**14.609**	**15.959**
BERTFSIE-Multi LSTM (Market)	**17.295**	**18.760**	15.117	16.582
BERTFSIE-VARMA-LSTM	51.476	40.272	41.045	42.634
BERTFSIE-VARMA-LSTM (Market)	52.311	40.241	41.889	42.972
ARIE-Multi LSTM	17.449	**15.596**	**15.351**	17.016
ARIE-Multi LSTM (Market)	**17.415**	17.706	16.507	**15.442**
ARIE-VARMA-LSTM	47.979	44.653	36.938	39.021
ARIE-VARMA-LSTM (Market)	47.979	42.433	38.643	38.735

Note. The boldface **numbers** stand for the least running time in that block. The numbers in red color stand for the least running time in the table. The unit is hour.

Table A.3: Running Time Comparison of Models in Forecasting Price Log-Returns and Directions of the Selected Stock. (Testing Proportion: 30%)

Model	Finance	Pharmaceutical	Retail	Technology
ARIMA	3.539	**0.305**	**0.693**	1.805
ARIMAX	**0.831**	0.722	1.043	2.616
VARMA	1.867	1.356	1.260	**1.422**
VARMA (Market)	41.208	47.355	46.921	43.602
Uni-LSTM	21.507	23.434	**16.918**	**18.825**
Multi-LSTM	21.877	23.864	17.652	19.537
Multi-LSTM (Market)	**19.705**	**20.876**	17.992	20.529
ARIMA-LSTM	**28.117**	**35.131**	37.682	**35.389**
VARMA-LSTM	43.451	38.974	39.413	37.780
VARMA-LSTM (Market)	42.164	36.105	**37.562**	37.305
BERTFSIE-Multi LSTM	22.931	24.661	17.228	**19.341**
BERTFSIE-Multi LSTM (Market)	**21.885**	**18.843**	**17.484**	19.390
BERTFSIE-VARMA-LSTM	42.783	38.559	33.231	38.254
BERTFSIE-VARMA-LSTM (Market)	42.915	36.955	37.096	37.812
ARIE-Multi LSTM	22.598	24.738	**15.929**	**18.046**
ARIE-Multi LSTM (Market)	**20.980**	**23.634**	17.253	19.536
ARIE-VARMA-LSTM	44.755	40.019	37.462	42.889
ARIE-VARMA-LSTM (Market)	41.504	32.611	38.724	39.212

Note. The boldface **numbers** stand for the least running time in that block. The numbers in red color stand for the least running time in the table. The unit is hour.

REFERENCES

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974. doi: 10.1109/TAC.1974.1100705.

Cagdas Hakan Aladag, Erol Egrioglu, and Cem Kadilar. Forecasting nonlinear time series with a hybrid methodology. *Applied mathematics letters*, 22(9):1467–1470, 2009. ISSN 0893-9659.

Dogu Araci. Finbert: Financial sentiment analysis with pre-trained language models. 2019.

George Athanasopoulos and Farshid Vahid. Varma versus var for macroeconomic forecasting. *Journal of business & economic statistics*, 26(2):237–252, 2008. ISSN 0735-0015.

Mattia Atzeni, Amna Dridi, and Diego Reforgiato Recupero. Fine-grained sentiment analysis on financial microblogs and news headlines. In *Semantic Web Challenges*, Communications in Computer and Information Science, pages 124–128. Springer International Publishing, Cham, 2017. ISBN 3319691457.

Malcolm Baker and Jeffrey Wurgler. Investor sentiment and the cross-section of stock returns. *The Journal of finance (New York)*, 61(4):1645–1680, 2006. ISSN 0022-1082.

Wei Bao, Jun Yue, and Yulei Rao. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PloS one*, 12(7):e0180944–e0180944, 2017. ISSN 1932-6203.

Nicholas Barberis, Andrei Shleifer, and Robert Vishny. A model of investor sentiment. *Journal of financial economics*, 49(3):307–343, 1998. ISSN 0304-405X.

Christopher M Bishop. *Pattern recognition and machine learning / Christopher M. Bishop.* Information science and statistics. Springer, New York, 2006. ISBN 0387310738.

George E. P. Box and Gwilym M Jenkins. *Time series analysis : forecasting and control / [by] George E. P. Box and Gwilym M. Jenkins.* Holden-Day series in time series analysis. Holden-Day, San Francisco, 1970.

George E.P. Box. *Time series analysis: forecasting and control.* Wiley series in probability and statistics. Wiley, New York, 5th ed edition, 2016. ISBN 9781118675021.

G.E.P. Box, G.M. Jenkins, G.C. Reinsel, and G.M. Ljung. *Time Series Analysis: Forecasting and Control.* Wiley Series in Probability and Statistics. Wiley, 2015. ISBN 9781118674925. URL `https://books.google.com/books?id=rNt5CgAAQBAJ`.

C. G. Broyden. The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations. *IMA Journal of Applied Mathematics*, 6(1):76–90, 03 1970. ISSN 0272-4960. doi: 10.1093/imamat/6.1.76. URL `https://doi.org/10.1093/imamat/6.1.76`.

Angela C Caliwag and Wansu Lim. Hybrid varma and lstm method for lithium-ion battery state-of-charge and output voltage forecasting in electric motorcycle applications. *IEEE access*, 7: 59680–59689, 2019. ISSN 2169-3536.

Qing Cao, Karyl B Leggio, and Marc J Schniederjans. A comparison between fama and french's model and artificial neural networks in predicting the chinese stock market. *Computers & operations research*, 32(10):2499–2512, 2005. ISSN 0305-0548.

Chung-Chi Chen, Hen-Hsen Huang, and Hsin-Hsi Chen. Nlp in fintech applications: Past, present and future. 2020.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. 2014.

Dami Choi, Christopher J Shallue, Zachary Nado, Jaehoon Lee, Chris J Maddison, and George E Dahl. On empirical comparisons of optimizers for deep learning. *arXiv preprint arXiv:1910.05446*, 2019.

Hyeong Kyu Choi. Stock price correlation coefficient prediction with arima-lstm hybrid model. 2018a.

Hyeong Kyu Choi. Stock price correlation coefficient prediction with arima-lstm hybrid model. 2018b.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018.

Luca Di Persio and Oleksandr Honchar. Recurrent neural networks approach to the financial forecast of google assets. *International journal of Mathematics and Computers in simulation*, 11:7–13, 2017.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Graham Elliott, C. W. J. (Clive William John) Granger, and Allan Timmermann. *Handbook of economic forecasting. Volume 1 edited by Graham Elliott, Clive W.J. Granger, Allan Timmermann.* Handbooks in economics; bk. 24. Elsevier, Boston, Mass., 2006. ISBN 9780444513953.

Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006. ISSN 0167-8655.

R Fletcher. A new approach to variable metric algorithms. *Computer journal*, 13(3):317–322, 1970. ISSN 0010-4620.

A Galántai. The theory of newton's method. *Journal of computational and applied mathematics*, 124(1):25–44, 2000. ISSN 0377-0427.

Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471, 2000. ISSN 1530-888X.

M Ghiassi, J Skinner, and D Zimbra. Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network. *Expert systems with applications*, 40 (16):6266–6282, 2013. ISSN 0957-4174.

Donald Goldfarb. A family of variable-metric methods derived by variational means. *Mathematics of computation*, 24(109):23–26, 1970. ISSN 0025-5718.

Banhi Guha and Gautam Bandyopadhyay. Gold price forecasting using arima model. *Journal of advance Management Journal*, 03 2016. doi: 10.12720/joams.4.2.117-121.

Mustafa Gülerce and Gazanfer Ünal. Forecasting of oil and agricultural commodity prices: Varma versus arma. *Annals of Financial Economics (AFE)*, 12(03):1–30, 2017. URL `https://EconPapers.repec.org/RePEc:wsi:afexxx:v:12:y:2017:i:03:n:s2010495217500129`.

Zahra Hajirahimi and Mehdi Khashei. Hybrid structures in time series modeling and forecasting: A review. *Engineering applications of artificial intelligence*, 86:83–106, 2019. ISSN 0952-1976.

E. J. Hannan. The identification of vector mixed autoregressive-moving average systems. *Biometrika*, 56(1):223–225, 1969. ISSN 0006-3444.

E. J. Hannan and B. G. Quinn. The determination of the order of an autoregression. *Journal of the Royal Statistical Society. Series B (Methodological)*, 41(2):190–195, 1979. ISSN 00359246. URL `http://www.jstor.org/stable/2985032`.

Donald Olding Hebb. The organization of behavior; a neuropsycholocigal theory. *A Wiley Book in Clinical Psychology*, 62:78, 1949.

Bruno Miranda Henrique, Vinicius Amorim Sobreiro, and Herbert Kimura. Literature review: Machine learning techniques applied to financial market prediction. *Expert systems with applications*, 124:226–251, 2019. ISSN 0957-4174.

Joshua Zoen Git Hiew, Xin Huang, Hao Mou, Duan Li, Qi Wu, and Yabo Xu. Bert-based financial sentiment index and lstm-based stock return predictability. 2019.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Shih-Feng Huang and Hsiang-Ling Hsu. Prediction intervals for time series and their applications to portfolio selection. *Revstat*, 18(1):131–151, 2020. ISSN 1645-6726.

R.J. Hyndman and G. Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018. ISBN 9780987507112. URL `https://books.google.com/books?id=_bBhDwAAQBAJ`.

Rob J Hyndman. Variations on rolling forecasts. `https://robjhyndman.com/hyndsight/rolling-forecasts/`, 2014.

Christopher James, Sergio Koreisha, and Megan Partch. A varma analysis of the causal relations among stock returns, real output, and nominal interest rates. *The Journal of finance (New York)*, 40(5):1375–1384, 1985. ISSN 0022-1082.

K Kamijo and T Tanigawa. Stock price pattern recognition-a recurrent neural network approach. In *1990 IJCNN International Joint Conference on Neural Networks*, pages 215–221 vol.1, 1990.

Yakup Kara, Melek Acar Boyacioglu, and Ömer Kaan 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*, 38(5):5311–5319, 2011. ISSN 0957-4174.

Resat Kasap. An Analysis of the Istanbul Stock Exchange National-100 Index: A Statistical Approach. *Istanbul Stock Exchange Review*, 2(6):27–34, 1998. URL `https://ideas.repec.org/a/bor/iserev/v2y1998i6p27-34.html`.

Colm Kearney and Sha Liu. Textual sentiment in finance: A survey of methods and models. *International review of financial analysis*, 33:171–185, 2014. ISSN 1057-5219.

Mergani Khairalla and Xu Ning. Financial time series forecasting using hybridized support vector machines and arima models. In *Proceedings of the 2017 International Conference on wireless communications, networking and applications*, WCNA 2017, pages 94–98. ACM, 2017. ISBN 1450353444.

Kyoung-jae Kim and Ingoo Han. Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. *Expert systems with applications*, 19 (2):125–132, 2000. ISSN 0957-4174.

Yoon Kim. Convolutional neural networks for sentence classification. 2014.

T Kimoto, K Asakawa, M Yoda, and M Takeoka. Stock market prediction system with modular neural networks. In *1990 IJCNN International Joint Conference on Neural Networks*, pages 1–6 vol.1, 1990.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Weicong Kong, Zhao Yang Dong, Youwei Jia, David J Hill, Yan Xu, and Yuan Zhang. Short-term residential load forecasting based on lstm recurrent neural network. *IEEE transactions on smart grid*, 10(1):841–851, 2019. ISSN 1949-3053.

Manish Kumar and Thenmozhi M. Forecasting stock index returns using arima-svm, arima-ann, and arima-random forest hybrid models. *International Journal of Banking, Accounting and Finance*, 5:284, 01 2014. doi: 10.1504/IJBAAF.2014.064307.

R.C Lacher, Pamela K Coats, Shanker C Sharma, and L.Franklin Fant. A neural network for classifying the financial health of a firm. *European journal of operational research*, 85(1): 53–65, 1995. ISSN 0377-2217.

Moshe Leshno and Yishay Spector. Neural network prediction analysis: The bankruptcy case. *Neurocomputing (Amsterdam)*, 10(2):125–147, 1996. ISSN 0925-2312.

Bing Liu. *Sentiment analysis and opinion mining Bing Liu.* Synthesis digital library of engineering and computer science. Morgan & Claypool, San Rafael, Calif. 1537 Fourth Street, San Rafael, CA 94901 USA, 2012. ISBN 1-60845-885-7.

Yanli Liu, Yuan Gao, and Wotao Yin. An improved analysis of stochastic gradient descent with momentum. 2020.

Martin Längkvist, Lars Karlsson, and Amy Loutfi. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern recognition letters*, 42:11–24, 2014a. ISSN 0167-8655.

Martin Längkvist, Lars Karlsson, and Amy Loutfi. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern recognition letters*, 42:11–24, 2014b. ISSN 0167-8655.

Spyros Makridakis. A survey of time series. *International statistical review*, 44(1):29–70, 1976. ISSN 0306-7734.

Xiliu Man, Tong Luo, and Jianwu Lin. Financial sentiment analysis(fsa): A survey. In *2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*, pages 617–622. IEEE, 2019. ISBN 1538685000.

Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943. ISSN 0007-4985.

Aidan Meyler, Geoff Kenny, and Terry Quinn. Forecasting irish inflation using arima models. 1998.

Marvin Lee Minsky and Seymour Papert. *Perceptrons : an introduction to computational geometry / Marvin Minsky and Seymour Papert.* MIT Press, Cambridge, Mass., 1969.

Y. E. Nesterov. A method for solving the convex programming problem with convergence rate $o(1/k^2)$. *Dokl. Akad. Nauk SSSR*, 269:543–547, 1983. URL `https://ci.nii.ac.jp/naid/10029946121/en/`.

Clemens Nopp and Allan Hanbury. Detecting risks in the banking system by sentiment analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 591–600, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1071. URL `https://www.aclweb.org/anthology/D15-1071`.

Ping-Feng Pai and Chih-Sheng Lin. A hybrid arima and support vector machines model in stock price forecasting. *Omega (Oxford)*, 33(6):497–505, 2005. ISSN 0305-0483.

Alan Pankratz. *Forecasting with dynamic regression models / Alan Pankratz.* Wiley series in probability and mathematical statistics. Applied probability and statistics. Wiley, New York, 1991. ISBN 9781118150528.

Seongik Park and Yanggon Kim. Building thesaurus lexicon using dictionary-based approach for sentiment classification. In *2016 IEEE 14th International Conference on Software Engineering Research, Management and Applications (SERA)*, pages 39–44. IEEE, 2016. ISBN 9781509008094.

Kriti Pawar, Raj Srujan Jalem, and Vivek Tiwari. Stock market price prediction using lstm rnn. In *Emerging Trends in Expert Applications and Security*, Advances in Intelligent Systems and Computing, pages 493–503. Springer Singapore, Singapore, 2018. ISBN 9811322848.

Ďurka Peter and Pastoreková Silvia. Arima vs. arimax–which approach is better to analyze and forecast macroeconomic time series. In *Proceedings of the 30th International Conference Mathematical Methods in Economics, Karviná, Czech Republic*, pages 11–13, 2012.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. 2018.

B.T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964. ISSN 0041-5553. doi: https://doi.org/10.1016/0041-5553(64)90137-5. URL `https://www.sciencedirect.com/science/article/pii/0041555364901375`.

Federico Alberto Pozzi, Elisabetta Fersini, Enza Messina, and Bing Liu. *Sentiment analysis in social networks.* Morgan Kaufmann, 2016.

Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12 (1):145–151, 1999.

Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 22(3):400 – 407, 1951. doi: 10.1214/aoms/1177729586. URL `https://doi.org/10.1214/aoms/1177729586`.

F Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386–408, 1958. ISSN 0033-295X.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature (London)*, 323(6088):533–536, 1986. ISSN 0028-0836.

Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. 2014.

Gideon Schwarz. Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2):461 – 464, 1978. doi: 10.1214/aos/1176344136. URL `https://doi.org/10.1214/aos/1176344136`.

Aliaksei Severyn and Alessandro Moschitti. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 959–962, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450336215. doi: 10.1145/2766462.2767830. URL `https://doi.org/10.1145/2766462.2767830`.

Dev Shah. Stock market analysis: a review and taxonomy of prediction techniques. *International journal of financial studies*, 7(2/26):1–22, 2019. ISSN 2227-7072.

D. F Shanno. Conditioning of quasi-newton methods for function minimization. *Mathematics of computation*, 24(111):647–656, 1970. ISSN 0025-5718.

Sima Siami-Namini and Akbar Siami Namin. Forecasting economics and financial time series: Arima vs. lstm. 2018.

Christopher A. Sims. Macroeconomics and reality. *Econometrica*, 48(1):1–48, 1980. ISSN 0012-9682.

Prerana Singhal and Pushpak Bhattacharyya. Sentiment analysis and deep learning: a survey. *Center for Indian Language Technology, Indian Institute of Technology, Bombay*, 2016.

Eugen Slutzky. The summation of random causes as the source of cyclic processes. *Econometrica*, 5(2):105–146, 1937. ISSN 0012-9682.

M. G. Sousa, K. Sakiyama, L. d. S. Rodrigues, P. H. Moraes, E. R. Fernandes, and E. T. Matsubara. Bert for stock market sentiment analysis. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1597–1601, 2019. doi: 10.1109/ICTAI.2019.00231.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. Lexicon-based methods for sentiment analysis. *Computational linguistics - Association for Computational Linguistics*, 37(2):267–307. ISSN 1530-9312.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. 2015.

Ayşe Soy Temür, Melek Akgün, and Günay Temür. Predicting housing sales in turkey using arima, lstm and hybrid models. *Journal of business economics and management*, 20(5):920–938, 2019. ISSN 1611-1699.

Paul C. Tetlock. Giving content to investor sentiment: The role of media in the stock market. *The Journal of finance (New York)*, 62(3):1139–1168, 2007. ISSN 0022-1082.

George C Tiao and Ruey S Tsay. Multiple time series modeling and extended sample cross-correlations. *Journal of business & economic statistics*, 1(1):43–56, 1983. ISSN 0735-0015.

Tijmen Tieleman, Geoffrey Hinton, et al. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2): 26–31, 2012.

Luis Torgo. *Data mining with R : learning with case studies / Luis Torgo.* Chapman & Hall/CRC data mining and knowledge discovery series. Taylor & Francis, CRC Press, Boca Raton, second edition. edition, 2017. ISBN 9781315399096.

Ruey S. Tsay. *Analysis of financial time series Ruey S. Tsay.* Wiley series in probability and statistics. Wiley, Hoboken, NJ, 3rd edition edition, 2010. ISBN 1-282-70783-3.

Ruey S. Tsay. *Multivariate time series analysis : with R and financial applications / Ruey S. Tsay.* Wiley series in probability and statistics. Wiley, Hoboken, New Jersey, 2014. ISBN 9781118617793.

Ruey S Tsay and Ruey S Tsay. *Multivariate time series analysis: with R and financial applications (wiley series in probability and statistics).* Wiley series in probability and statistics. WILEY, Somerset, 2013. ISBN 1118617797.

Raymond Y.C Tse. An application of the arima model to real-estate prices in hong kong. *Journal of property finance*, 8(2):152–163, 1997. ISSN 0958-868X.

John W. (John Wilder) Tukey. *Exploratory data analysis / John W. Tukey.* Addison-Wesley series in behavioral science. Addison-Wesley Pub. Co., Reading, Mass., 1977. ISBN 0201076160.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

Gilbert Walker. On periodicity in series of related terms. *Proceedings of the Royal Society of London. Series A, Containing papers of a mathematical and physical character*, 131(818): 518–532, 1931. ISSN 0950-1207.

Gang Wang, Tianyi Wang, Bolun Wang, Divya Sambasivan, Zengbin Zhang, Haitao Zheng, and Ben Zhao. Crowds on wall street: Extracting value from collaborative investing platforms. In *Proceedings of the 18th ACM Conference on computer supported cooperative work & social computing*, CSCW '15, pages 17–30. ACM, 2015. ISBN 1450329225.

Jung-Hua Wang and Jia-Yann Leu. Stock market trend prediction using arima-based neural networks. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 4, pages 2160–2165 vol.4. IEEE, 1996. ISBN 0780332105.

David West, Scott Dellana, and Jingxia Qian. Neural network ensemble strategies for financial decision applications. *Computers & operations research*, 32(10):2543–2559, 2005. ISSN 0305-0548.

Neo Wu, Bradley Green, Xue Ben, and Shawn O'Banion. Deep transformer models for time series forecasting: The influenza prevalence case. 2020.

Peter Yamak, Li Yujian, and Pius Gadosey. A comparison between arima, lstm, and gru for time series forecasting. In *Proceedings of the 2019 2nd International Conference on algorithms, computing and artificial intelligence*, ACAI 2019, pages 49–55. ACM, 2019. ISBN 9781450372619.

Steve Y Yang, Sheung Yin Kevin Mo, and Anqi Liu. Twitter financial community sentiment and its predictive relationship to stock market movement. *Quantitative finance*, 15(10):1637–1656, 2015. ISSN 1469-7688.

G. Udny Yule. On a method of investigating periodicities in disturbed series, with special reference to wolfer's sunspot numbers. *Philosophical transactions of the Royal Society of London. Series A, Containing papers of a mathematical or physical character*, 226(636-646):267–298, 1927. ISSN 0264-3952.

G.Peter Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing (Amsterdam)*, 50:159–175, 2003a. ISSN 0925-2312.

G.Peter Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing (Amsterdam)*, 50:159–175, 2003b. ISSN 0925-2312.

Z. Zhao, R. Rao, S. Tu, and J. Shi. Time-weighted lstm model with redefined labeling for stock trend prediction. In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1210–1217, 2017. doi: 10.1109/ICTAI.2017.00184.

BIOGRAPHICAL SKETCH

Siqi Mao was born and raised in Inner Mongolia, China. He got admission to Inner Mongolia University in 2011. After completing a Bachelor of Science in Mathematics and a Bachelor of Economics in Finance, Siqi joined the Ph.D. program in Statistics at Florida State University in Fall 2017. He will start his career in the financial industry after graduation.