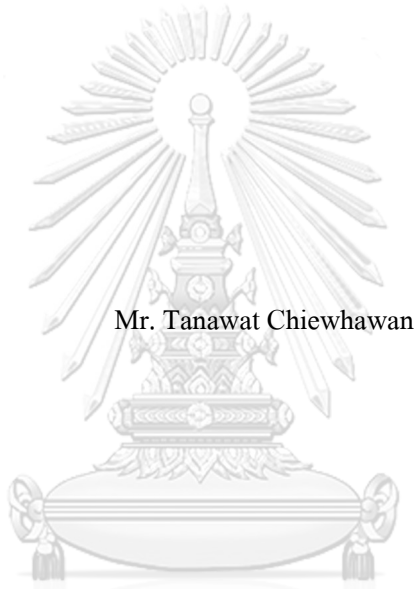
THAI STOCK RETURN PREDICTION USING DEEP LEARNING MODELS WITH STOCK INDICATORS AND TEXTUAL FEATURES

Mr. Tanawat Chiewhawan



A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science in Computer Science

Department of Computer Engineering

FACULTY OF ENGINEERING

Chulalongkorn University

Academic Year 2019

การทำนายผลตอบแทนหุ้นไทยด้วยแบบจำลองการเรียนรู้เชิงลึกประกอบกับตัวชี้วัดหุ้นและข้อมูลเชิงตัวอักษร

นายธนวัฒน์ ชิวหวรรณ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2562

Thesis Title	THAI STOCK RETURN PREDICTION USING DEEP LEARNING MODELS WITH STOCK INDICATORS AND TEXTUAL FEATURES
By	Mr. Tanawat Chiewhawan
Field of Study	Computer Science
Thesis Advisor	Assistant Professor PEERAPON VATEEKUL, Ph.D.

Accepted by the FACULTY OF ENGINEERING, Chulalongkorn University in Partial Fulfillment of the Requirement for the Master of Science

..... Dean of the FACULTY OF ENGINEERING

(Professor SUPOT TEACHAVORASINSKUN, D.Eng.)

THESIS COMMITTEE

..... Chairman

(Professor BOONSERM KIJSIRIKUL, D.Eng.)

..... Thesis Advisor

(Assistant Professor PEERAPON VATEEKUL, Ph.D.)

..... Examiner

(DUANGDAO WICHADAKUL, Ph.D.)

..... Examiner

(Assistant Professor TANAKORN LIKITAPIWAT, Ph.D.)

..... External Examiner

(Thanapat Kangkachit, Ph.D.)

ธนวัฒน์ ชิวหวรรณ : การทำนายผลตอบแทนหุ้นไทยด้วยแบบจำลองการเรียนรู้เชิงลึกประกอบกับตัวชี้วัดหุ้นและข้อมูลเชิงตัวอักษร. (THAI STOCK RETURN PREDICTION USING DEEP LEARNING MODELS WITH STOCK INDICATORS AND TEXTUAL FEATURES) อ.ที่ปรึกษาหลัก : ผศ. ดร.พีรพล เวทีกูล

การทำนายหุ้นเป็นสิ่งที่ยากเนื่องจากความไม่แน่นอนและความผันผวนจากปัจจัยภายนอกที่หลากหลายและ ส่งผลต่อพฤติกรรมของหุ้น ในปัจจุบัน ได้มีการนำเทคนิคการเรียนรู้เชิงลึก (Deep Learning) มาใช้กันมากขึ้นในงานวิจัย ทั้งนี้งานวิจัยส่วนมากจะสนใจ ข้อมูลรับเข้าของแบบจำลองเพียง เฉพาะ ข้อมูลเชิงตัวอักษร หรือ ข้อมูลเชิงตัวเลข อย่างใดอย่างหนึ่งเท่านั้น อีกทั้งมักที่จะเลือกสนใจ เพียง หุ้นตัวเดียว หรือ ราคาตลาด เพียงตัวเดียว. งานวิจัยนี้ มุ่งที่จะทำนายหุ้นหลายตัว ด้วยข้อมูลทั้งสองรูปแบบ โดยแบบจำลองที่เรานำเสนอประกอบด้วย โครงข่ายประสาทเทียมเวียนซ้ำประกอบคู่กลไกจุดสนใจ (Dual-stage attention model) การอนุมานความสัมพันธ์ระหว่างหุ้น และ การผสมผสานข้อมูลเชิงตัวอักษร . เริ่มจากกระบวนการอนุมานความสัมพันธ์ระหว่างหุ้น ที่เรานำเสนอ ได้ถูกออกแบบเพื่อ จัดการกับข้อมูลเชิงอนุกรมเวลาที่หลากหลาย (เช่น ข้อมูลหุ้น เชิงพื้นฐาน หรือ เชิงเทคนิคอล) รวมถึง เพื่อเพิ่ม ความสามารถในการจัดอันดับหุ้น ด้วยฟังชั่นวัตถุประสงค์ที่เสริมการจัดอันดับ เราได้สาธิตวิธีการที่จะแปลงและสร้างตัวแทนจาก ข้อมูลเชิงตัวอักษร ด้วยวิธี เครือข่ายประสาทเทียมเชิงลำดับขึ้น (Hierarchical neural network) และ ตัวแทนการเข้ารหัสทวิทิศทางจากตัวแปลง (BERT) .ในลำดับสุดท้าย เราได้ศึกษา แนวทางที่จะ บูรณาการทุกแบบจำลองเข้าด้วยกัน ให้สามารถ จัดการข้อมูลนำเข้าทั้งสองประเภทอย่างมีประสิทธิภาพ เพื่อมุ่งที่จะเพิ่มความสามารถในการทำนายหุ้นให้ดียิ่งขึ้น.

สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่อนิสิต .....

ปีการศึกษา 2562 ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

6170929321 : MAJOR COMPUTER SCIENCE

KEYWORD: Deep learning model, Natural language processing, Stock return prediction, Machine Learning, textual sentiment analysis, Neural Network, Attention mechanism, LSTM

Tanawat Chiewhawan : THAI STOCK RETURN PREDICTION USING DEEP LEARNING MODELS WITH STOCK INDICATORS AND TEXTUAL FEATURES. Advisor: Asst. Prof. PEERAPON VATEEKUL, Ph.D.

Stock prediction task is notoriously challenging due to the uncertainty and dynamic external factor which influence the stock behavior. Recently, Deep learning research is gaining popularity on this task but often focuses on only a particular type of data; numeric indicators or textual information. Moreover, most researches focus on only a single stock or a market index. In this paper, we aim to predict multiple stock returns using both types of data. The model consists of dual-stage attention recurrent neural network, our proposed stock relation inference framework, and textual features integration. The proposed stock relation inference help tackles multiple time-series features (stocks indicators such as fundamentals or technicals) as well as add the ranking ability to the model from the combined ranking loss function. We demonstrate how to represent textual features with the hierarchical neural network and the BERT embedding method. Finally, we explore the approaches to integrate all elements in order to handle both types of data effectively, aiming to improve the performance of the stock prediction task.

Field of Study:	Computer Science	Student's Signature .....
Academic Year:	2019	Advisor's Signature .....

ACKNOWLEDGEMENTS

This thesis and master's degree are some of the most valuable experiences in my life. I was interested in computer science since young, but I decided to pursue petroleum engineering instead of computer engineering during the bachelor's degree. However, I finally made the decision to engage the computer area in the master's degree, and I was not disappointed, I enjoyed this area so much working and learning from many amazing people. I am grateful for all the kind support I received from my family, advisor, friends, and various people during this time to complete my thesis.

I feel fortunate to be apart of my advisor's lab. Having worked with him, Asst. Prof. Peerapon Vateekul, Ph. D., I felt that he is one of the most dedicated professors I have ever met. The coaching, knowledge sharing, and problem-solving he support us throughout the course is astonishing. Friends and colleagues in the Datamind lab are amicable and helpful. I could never achieve this without their support too.

Thank you to all the committees consists of Prof. Boonserm Kijsirikul, D.Eng., Duangdao Wichadakul, Ph.D., Asst. Prof. Tanakorn Likitapiwat, Ph.D., Thanapat Kangkachit, Ph.D., for making valuable suggestions for this thesis.

Our work is also partly supported on the research budget by Capital Market Research Institute (CMRI), The Stock Exchange of Thailand (SET), during Capital Market Research Innovation contest 2019. Moreover, the permission to use the SET SMART dataset for academic purposes was granted from the Financial Laboratory, Chulalongkorn Business School with support from Asst. Prof. Tanakorn Likitapiwat.

We would like to express our appreciation to all the parties who supported us and involved with this research. Thank you so much.

Tanawat Chiewhawan

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Pages

CHAPTER 1

INTRODUCTION

1.1 Motivation

Stock prediction task is notoriously a challenging subject because of the high volatility and the influence of dynamic external factors such as the global economy and investor's behavior. This topic of stock predictability has long been controversial. The earlier works on the efficient-market hypothesis (EMH) [1, 2] suggest the price reflects all information suddenly, and the movement is random processes. However, various studies from many fields attempt to explore this challenge. Recently deep learning is one of the emerging areas showing promising results.

Most of the deep learning research for stock prediction focused on a single asset or an index such as the popular S&P500 index [3, 4]. While the work [5, 6] shows that prediction on multiple stocks with consideration to its correlation provides a better result comparing to independently making the prediction. Furthermore, because of the high uncertainty in the stock market, we design our problem to be multiple stock predictions, then rank those predictions and choose only the best-expected stock to be invested. This investment from multiple stock rankings will reduce the risk when trading on only a single asset or stock.

Most of the time, the researcher focused on maximizing accuracy for the classification task or minimized regression error for the regression task when optimizing the stock prediction model. However, when making multiple stock predictions or ranking stock, there is a discrepancy between such accuracy with the optimal profit, as demonstrated in Table 1. Method 1) suggests a higher profit stock for trading while method 2) with higher regression accuracy (less mean square error) suggests a lower profit stock. The research in [6] shows a way to incorporate a ranking-aware score to the deep learning model.

Table 1 An intuitive explanation for the discrepancy in prediction accuracy and actual profit, reference from Table 2 of [6]

	Ground Truth			**1) Ranking-aware prediction**					**2) MSE optimized prediction**				
Stock	**A**	**B**	**C**	**A**	**B**	**C**	**MSE**	**Profit**	**A**	**B**	**C**	**MSE**	**Profit**
Stock returns	30	10	-50	**50**	-10	-50	266	30	20	**30**	-40	200	10

Additionally, countless factors could drive the price for a particular stock, which they come in the form of both numerical and textual forms. The work in [7] adds an attention mechanism to improve the model with the high number of time-series inputs. However, they still focus on only numerical types, while in the real world, most of the investors able to consider both textual and numerical data. The works like [5, 8-10] incorporate both of them into a single model with promising results, and most of them agreed that using the news headlines or news titles should be sufficient for stock prediction representing the whole article. For example, Ding, Zhang, Liu and Duan [9] research proposed a decent method using The OpenIE (Open Information Extraction) on English news headlines. The results show better accuracy comparing to their baseline. Unfortunately, the OpenIE is an example of a technique unavailable to other languages rather than English.

Consequently, we will explore two deep learning architectures to represent textual information. The first is the hierarchical neural network from DeepClue [11], which constructs hierarchical layers such as word, bigram, and news title to represent textual information for stock prediction. The second is an adaptation to one of state of the art in NLP, Bidirectional Encoder Representations from Transformers (BERT) [12].

Finally, there are only a few deep learning pieces of research for the Thai stock market, and to the best of our knowledge, none of them incorporates Thai textual inputs to the deep learning prediction model yet. Also, the Thai language has its unique challenges, such as the tokenization problems which might affect the performance of the model. This research will also aim to explore this Thai news title integration to the stock prediction deep learning model.

This research aims to improve the model's profitability from the prediction by exploring and addressing multiple keys aspects. First, a ranking aware model for resolving profit and accuracy discrepancy. The Attention mechanism to tackle numerous factors (inputs) to stock prediction and Next, how to incorporate both textual and numerical inputs exploring textual representation and integration approach and finally investigate and address Thai textual input challenges.

1.2 Objective

There are three main objectives of this research:

1. Propose a deep learning framework for multiple Thai stock return predictions with stock relation inference
2. Enhancement of the proposed model for numerous features input, both numerical and textual type inputs
3. Explore methods to tackle challenges of Thai textual news inputs while to improve prediction performance

1.3 Scope of works

1. Stock Exchange of Thailand (SET) data duration from 2008 to 2018 focusing on 64 target stocks as described in "Target stocks pre-selection" 5.2 section
2. Textual data of the economic news headline from online sources
3. Measure performance of the model with root mean square error for regression evaluation (RMSE), Mean Reciprocal Rank score of the top stock (MRR-Top) for the stock ranking performance, and profitability from trading simulation
4. Investigate effect model performance by adding textual information into our model

1.4 Expected result

1 Improve the deep learning model performance on multiple stock return predictions.

2 Propose a suitable technique to integrate Thai textual stock news input into the deep learning model

1.5 Research Plan

1 Study the related works and literature review

2 Prepare the experimental design

3 Develop, implement and experiment models

4 Summarize preliminary result

5 Prepare for thesis proposal topic examination

6 Experiment further as in the proposal examination comments

7 Evaluate experimental result and tuning model as needed

8 Publish academic paper

9 Conclude results and write up the thesis

10 Prepare for the final thesis examination

1.6 Publications

"Stock Return Prediction Using Dual-Stage Attention Model with Stock Relation Inference" Chiewhawan T., Vateekul P. (2020). In: Nguyen N., Jearanaitanakij K., Selamat A., Trawiński B., Chittayasothorn S. (eds) Intelligent Information and Database Systems. ACIIDS 2020. Lecture Notes in Computer Science, vol 12033. Springer, Cham

"Explainable Deep Learning for Thai Stock Market Prediction Using Textual Representation and Technical Indicators" Chiewhawan T., Vateekul P. (2020). In: 2020 The 8th International Conference on Computer and Communications Management (ICCCM 2020)

1.7 Awards

Capital Market Research Innovation Paper 2018

"Attention-based Deep Learning Model on Financial Big Data"

Capital Market Research Institue Research Grant 2019

"Stock Market Prediction Using Deep Learning Based Model with Event Embedding and Technical Indicators"

CHAPTER 2

BACKGROUND KNOWLEDGE

The background knowledge related to this research is separated into three main topics stock prediction and ranking task, artificial neural networks, and textual representation.

2.1 Stock Prediction and Ranking Task in Deep Learning

2.1.1 Stock prediction in deep learning

Almost all of the stock prediction tasks in deep learning is in the form of either regression or classification problem. On the regression task, the output of the model is usually expressed in price or return for the future time-step, for example, a task to predict tomorrow's return (percentage of profit changes) of the S&P500 index. For the classification task, it is often referring to as a trend prediction task, which requires additional labeling steps to convert future information to classes. Classification examples such as a task to predict whether the S&P500 index will be up (return >0%) or will be down (return < 0%). Also, both tasks could lead to different trading actions depending on their strategy. Figure 1 shows that a good prediction model on the regression task might perform poorly on classification and vice versa. Thus, depending on the objective of the model, choosing and designing a task for the prediction model is critical.

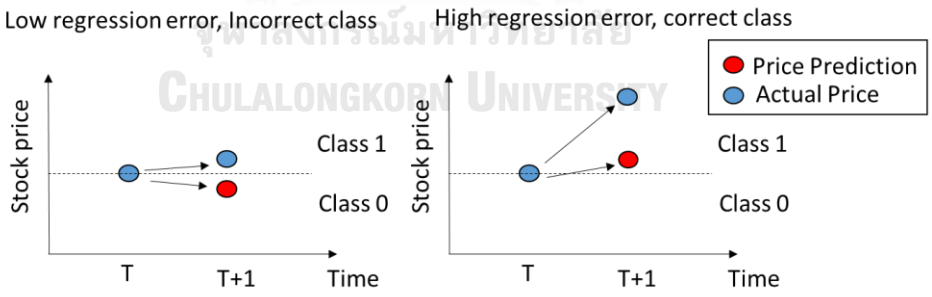


Figure 1 Regression and Classification task in Stock Prediction

2.1.2 Mean Reciprocal Rank (MRR)

In this research, we select a regression task to predict the return of multiple stocks in the future time step and then rank between those predicted returns to support investment decisions. This ranking task will be score using the mean reciprocal rank (MRR) [13], which is a statistic

measure to evaluate the process that produces a list of possible query responses. Equation 1 calculates the mean reciprocal rank as follows:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \quad (1)$$

where $rank_i$ refers to the rank position within the list size Q. We adopt this MRR to evaluate the rank of the top stock from our predicted ranking task. An example of how to calculate MRR for the top stock is shown in Table 2. First, the objective is to measure the ability to rank the stock "A," which, for simplicity, we assume actual rank for stock A, B, and C are equal to 1st,2nd,3rd for all three days. The MRR for the stock A is equal to (1/2+1/1+1/3)/3 =0.61. If we correctly predict "A" as the first rank for all three days, the calculated MRR will be equal to 1.

Table 2 Example MRR score calculation for stock A

Stock Name	Actual rank three days	Predict day 1	Predict day 2	Predict day 3
A	1st	2nd	1st	3rd
B	2nd	1st	3rd	2nd
C	3rd	3rd	2nd	1st
Reciprocal rank for stock A each day		1/2	1/1	1/3
MRR		(1/2+1/1+1/3)/3 =0.61		

2.2 Artificial Neural Network (ANN) & Deep Neural Networks (DNN)

This section describes the core neural network architectures used in this study.

2.2.1 Artificial Neural Network (ANN)

Artificial Neural Networks (ANN) is one of the most popular methods within the Machine Learning area. The model consists of interconnected computational units called neurons that simulating a more straightforward structure of neurons present in the human brain. Those neurons are constructed into groups of the layer, including an input layer, an output layer and hidden layer(s) in between the input and output layer. The neurons in each layer calculate their inputs before feed outputs to the next subsequent layer. Then finally, the output layer provides prediction, which can

be, for example, classification or regression value. The calculating process in each neuron usually involves a matrix of constant weight and an activation function, which allows the model to capture the nonlinearity and complexity of the problems. The structure in which there is no input-output cyclic between interconnected layers is called the Feed Forward Neural network (FNN). In comparison, Deep Neural Network (DNN) is often referred to as an ANN with many hidden layers or a more complex neuron structure. Figure 2 below illustrates an example of a feed-forward neural network perceptron.

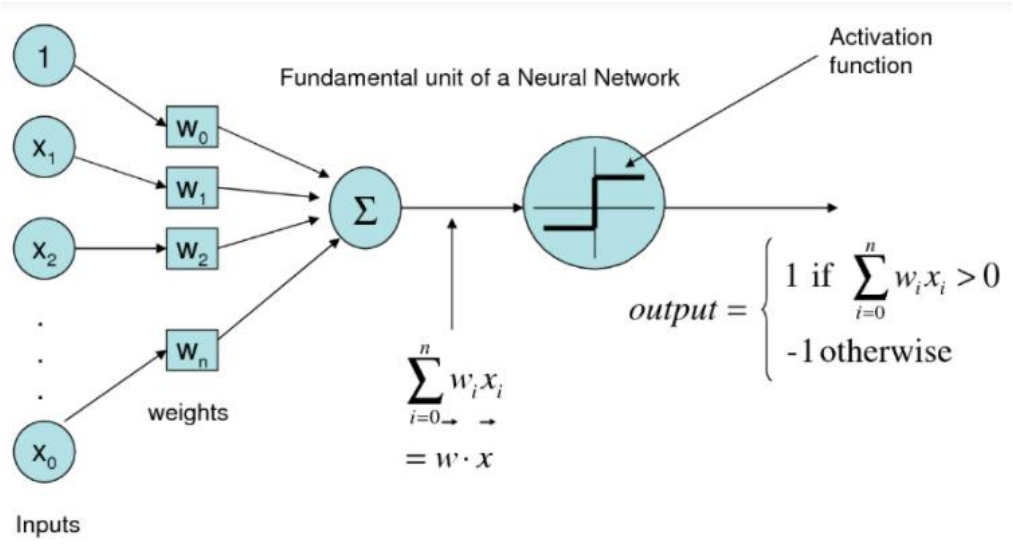


Figure 2 Fundamental of a Neural Network [14]

2.2.2 Recurrent Neural Network (RNN)

Recurrent Neural Network (RNN) is a type of ANN that was designed for processing sequential data. Unlike FNN, RNN's neurons form a directed graph that allows neurons to learn from its previous state. Outputs from the hidden layer can be recurrent into the same neuron in the next time steps, causing a self-loop. Figure 2 illustrates how RNN loops its output back to itself. RNN's ability to learn sequential information make them suitable with the task such as handwriting recognition, speech recognition. Also, Financial data are usually presented in forms of time series such as stock prices, buy and sell volume. Thus, RNN can be useful for financial data. There are various types of RNN architecture, depending on the objective of each problem. For example, a Long short-term memory (LSTM) model is becoming more popular. The model shows better

performance than RNN when dealing with longer terms of sequences of data. More detail on LSTM will be cover in the next section.

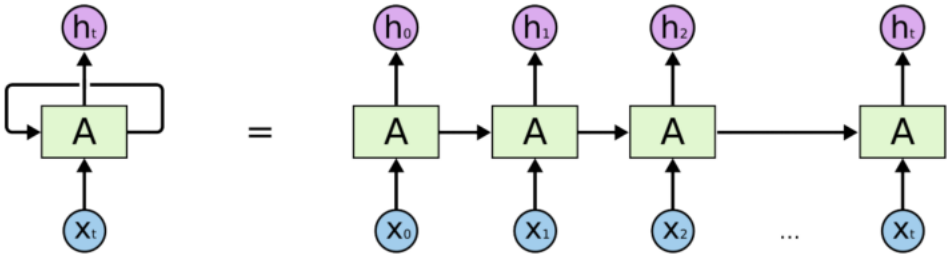


Figure 3 An RNN's recurrent connection, looping its hidden layer output to the next sequence reference from [15]

2.2.3 Long-Short Term Memory (LSTM)

Long Short Term Memory Networks (LSTMs) is a special type of Recurrent Neural Network which able to handle long period dependencies of its inputs. The models work tremendously well on various problems and have actively become more widespread.

LSTMs are explicitly designed to deal with long-term problems. The structure shares recurrent similarities with RNN, but the core computation units contain unique four interacting gates. These gates help guiding them on how to memorize and forget the inputs. Figure 4 illustrates the four gates mechanism in the LSTMs. First, the forget gate with the Sigmoid activation function decides which previous information in the memory cell should be forgotten. Then the next two gates decide what to memorize and what its memory magnitude or values are utilizing both Sigmoid and the Tanh activation function. Finally, the last gate combines LSTM input with cell memory to generate output for the current unit. Those outputs can be fed into the next LSTM layer along with memory cells to make the next prediction. The memory cell which passes through each LSTMs helps avoid vanishing gradient problem that could be observed in a vanilla RNN model.

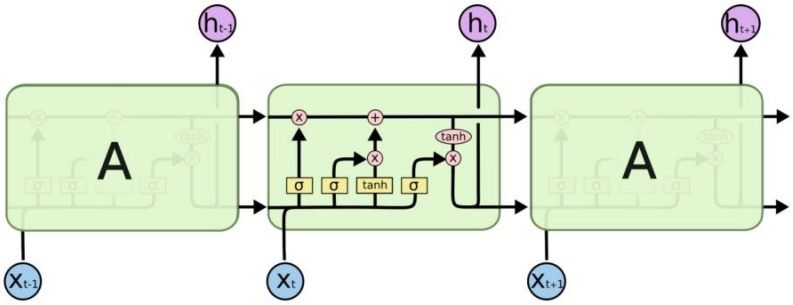


Figure 4 LSTMs module and its four gates [16]

2.3 Textual Representation

The textual representation is an essential process for Natural Language Processing (NLP). The model could not directly comprehend textual information, so text transformation to numeric is needed. There are various techniques available to process these texts, for example:

2.3.1 Bags of Words

This method represents textual information with a list of words in vocabulary count or word frequency for example with the text: "I love dog, you love cat" can be represented as [1,2,1,1,1,0] if the word vocabulary is ["I", " love", " dog", " you", " cat", " fish"].

2.3.2 Term Frequency Inverse Document Frequency (TF-IDF)

This method is similar to the bags of word but instead represents the text in Term Frequency (TF) multiply by the Inverse Document Frequency (IDF). Equation 2 and 3 shows the calculation of TF-IDF.

$$TFIDF = TF \; x \; IDF \tag{2}$$

$$IDF = log(\frac{N}{n_t}) \tag{3}$$

Where N is the total number of documents in the data set, n_t is the number of the document which contains the word.

2.3.3 Word Embedding

This method converts each word into a word vector. This vector was designed in the way that similar words should have similar vector values with a small distance between them. Word2Vec[17] is one example of the popular methods used. For this study, we select a pre-train ULMfit (thai2fit) framework to aid in the word embedding.

2.3.4 Bidirectional Encoder Representations from Transformers (BERT)

BERT (Bidirectional Encoder Representations from Transformers) [12] is one of the breakthrough states of the art model in Natural Language Processing (NLP). Researchers at the Google AI Language published this paper in 2018 results in a significant advancement in varieties of tasks, including Question Answering, Natural Language Inference. BERT architecture consists of Transformers, an attention mechanism built to learn contextual relations between words. Google also provides its pre-trained model, which was pre-training on enormous textual Corpus the BooksCorpus (800M words) (Zhu et al., 2015) and English Wikipedia (2,500M words). This pre-trained can be transferred and fine-tuned for any specifics tasks. However, for the Thai language, there are two pre-trained options available.

1. All Thai Bert training from ThaiWikipedia https://github.com/ThAIKeras/bert
2. Multilingual Cased (BERT-Base) 104 languages https://github.com/google-research/bert/blob/master/multilingual.md

We select the option 2. Multilingual Cased Bert for this research due to the perplexity performance after fine-tuning with our news. Figure 5 shows example illustrations of the BERT model.

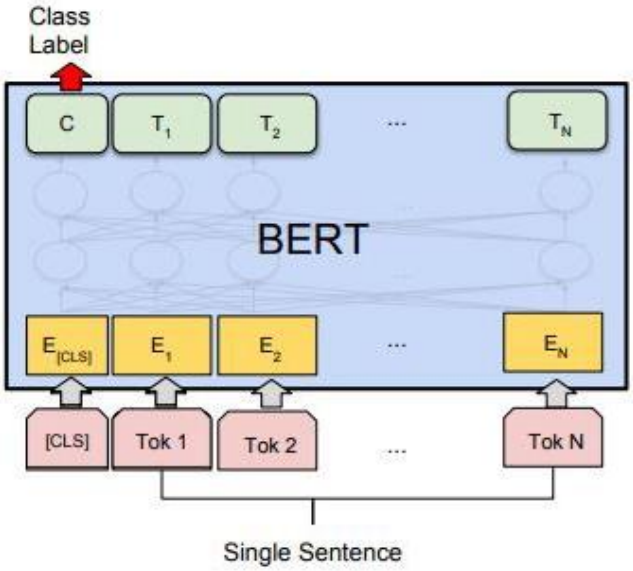


Figure 5 BERT example architecture for classification task[12]

CHAPTER 3

LITERATURE REVIEW

3.1 Machine Learning and Deep Learning in Stock Prediction

Machine learning has become popular in the stock prediction research due to its performance and ability to handle increasing information. For example, the works from [18, 19] conduct comparative experiments using multiple algorithms such as Support Vector Machine (SVM), Random Forest (RF), and Artificial Neural Network (ANN). The results show that RF outperforms other baseline models in the metrics of accuracy on stock trend classification as well as trading profit. Recently, more modern approaches start utilizing a deep learning model in their studies. In references [20-22], implement a Long-Short term memory recurrent neural network (LSTM) [23] with successful results. This LSTM is one of the most widespread algorithms to process time-series data. In [22] used the LSTM with numerous features of generated technical indicators to predict stock trends successfully. While [24] explores a modified LSTM to enhance the model's feature extraction.

However, with a various selection of features available in the financial market, the model becomes more challenging to converge for the solution. Hence, recent deep learning researches aim to implement more techniques to enhance models on those challenges. T. Hollis, S.E. Yi, and A. Viscardi [25] investigate LSTM with an attention mechanism. Their results align with other researches showing time-series forecast improvement [7, 26]. We will also utilize the attention mechanism to boost model prediction with numerous features time series.

3.2 Stock Prediction with Multi-variate Numerical Input

In 2017, David M. Q. Nelson et al. [22] explored the feasibility of the Long-Short-term memory model on the stock trend prediction task. The LSTM model is best-known for sequential prediction [23] in which they use in the experiment on five individual stocks with 180 features, including generated technical indicators and price information. The results show that, on average, the LSTM outperforms other baselines. However, in some particular stock, Random forest shows better results.

In the same year, Qin Y et al. [7] proposed a nonlinear autoregressive exogenous model called "Dual-Stage Attention Recurrent Neural Network (DA-RNN)" for time-series task. DA-RNN is an attention-based recurrent neural network with two attention mechanisms, input attention, and temporal attention. First, the input attention layer enhances a recurrent model that handles multiple time series inputs. It captures and differentiates importance among feature inputs then applies attention weights to them. Second, a temporal attention layer processes the encoded information from the input attention layer and grasps the significance of them at each time step then applies the temporal attention weight ahead of the LSTM prediction layer. Their experiment tests DA-RNN on the NASDAQ100 dataset with more than 80 inputs for driving features. The DA-RNN shows promising prediction performance, but the author designed DA-RNN for a single time series prediction, which does not fulfill our multiple stocks prediction objectives. We will modify this DA-RNN specifically to suits our multiple stock predictions objective.

3.4 Multiple Stock Prediction and Stock Ranking

In 2017, Fischer T. and Krauss C.[20] conducted a comprehensive study on stock trend prediction using LSTM. Their works focus on 500 stocks prediction in the S&P 500 index. They select top-k stocks from multiple stocks prediction for long positions and bottom-k stocks for short positions. This setting with LSTM shows better results than other models such as the Random Forest and vanilla Deep Neural Networks. However, their numerical feature was only a single sequence of the stock return, neglecting other possible relevance features.

In 2019, Feng F. et al.[6] proposed a model framework called "Temporal Relational Ranking" for stock prediction. Their model proposed a temporal graph convolution, which processes stock relations such as ownership, partnership (sparse binary features). They also introduce stock relation ranking frameworks that utilized ranking loss in their model. However, they conduct the model experiments on only five feature series, which are the closed price and four periods of a simple moving average of the close price. We apply their proposed relation ranking framework to our model by adding multi-features processing of the attention mechanism.

There are three main benefits of multiple stocks prediction over a single stock prediction. Firstly, we could capture the relations between stocks. Akitas [5] shows that grouping stocks within the same industry could benefit model predictions. For example, companies that are competitors could have an opposite trend, while companies that are trading partners could have similar trends. Secondly, an optimum buying or selling signal for a single stock does not occur very often.

Exploring multiple stocks increases trading opportunities and reducing risk. Finally, with multiple stock returns prediction, we can train the model to rank among those stocks and suggest only the top expected return for trading. This ranking method can help the model to improve profits. As mentioned earlier in [6], there is a discrepancy between optimizing the model accuracy and maximizing profit.

3.3 Stock Prediction with Textual Information

News article had been one of the significant influences on stock trends. Early researches utilized this text type input to the stock trend prediction model relying on shallow feature methods such as bag-of-words, named entities, and noun phrases, etc. Ding, Zhang, Liu and Duan [9] applied a more advanced method called Open Information Extraction (Open IE) to capture the structured event of the news headline for trend prediction in the S&P 500 index. The result shows that this event representation works well on more than 550,000 news headlines and outperforms the bag-of-words method by increasing its prediction accuracy. They suggest the news headlines should be sufficient for textual features comparing to using the whole article. Minh, Sadeghi-Niaraki, Huy, Min and Moon [27] also investigate the implementation of various recurrent neural network architectures, namely LSTM, GRU, BGRU. These layers were added over the event representation method [3] to improve model prediction. His experiments show that the text input model gains 3-4% accuracy improvement by adding those layers. Shi, Teng, Wang, Zhang and Binder [11] proposed a framework for stock return prediction on textual news input with the Hierarchical neural network structure. Their model embedded textual input into three layers of representation: word, bigram and news title layer before feeding into a feed-forward regression dense layer. Figure 6 shows a hierarchical neural network structure for news title representation. This hierarchical structure will be one of our experiments for Thai stock news.

Recently, BERT [12] published by Google in 2018, is one of the breakthrough states of the art model in NLP, leading to significant advancement in a variety of tasks, such as Question Answering, Natural Language Inference. BERT architecture consists of Transformers, an attention mechanism built to learns contextual relations between words. In 2019, some researcher start applying the BERT to the financial domain, Araci [28] proposed the finBERT which is a pre-trained language model on the financial corpus, it significantly improves sentiment accuracy on financial news. [29, 30] also shows enhancement in sentiment classification using BERT based model. We will also consider BERT architecture as one of our candidates to represent textual information.

Most of the previous researches focus on either textual or numerical information as an input, but not both for stock prediction. Vargas, Lima and Evsukoff [8], however, is one of the researches that explore this idea of combining two types of data as input to the model. The model consisted of a textual input layer with word2vec embedding and technical indicators (TIs) layer which inputs a delayed sequence of seven technical indicators generated in chronological order. These two layers were then joined with the LSTM layer for stock trend classification. His work compared extensively with multiple combinations of baseline, mostly derived from Ding, Zhang, Liu and Duan [9]'s works. The technique such as word embedding, event embedding, a bag of words and event tuple representation are compared with their prposed model which added technical indicator together with recurrent CNN layer. Akita, Yoshihara, Matsubara and Uehara [5] is another example that adopted the idea of utilizing both types of data input to the model. The works focused on Nikkei stock market data from 2001 to 2008. They also consider the relationship between the company's stocks by training and predicting all ten stocks within the same industry all together in a single model. Their proposed model consisted of two input layers. First, the textual layer used Paragraph Vector representation for input news headlines. Second, the parallel layer input normalized price inputs of each company. Both are then concatenated into the LSTM layer for the regression closing price. The results show that considering both textual and numerical could significantly improve profitability on trading simulation and training data for a company within the same industry together could significantly impact model performance. From our observation, most

of the research concatenate numerical inputs and textual input (after vectorized) into deep learning model with successful performance.

Another interesting approach for textual news input integration to deep learning models is using sentiments information. Jiawei and Murata [31] combined sentiments feature from textual inputs with numerical features, as shown in Figure 7. The numerical inputs including fundamentals, technical and macro time series, are all fed into the autoencoder model before concatenating with sentiment representation. However, it may require tremendous effort to label sentiment value for all news if the data was unlabeled. Yadav, Jha, Sharan and Vaish [32] suggest an approach to auto labeling sentiment news using market feedback. They also investigate a method to predict those sentiment values using a technique such as Point-wise Mutual Information-Information Retrieval (PMI-IR), an unsupervised approach as well as testing the SVM + word2vec method which is a supervised approach. The sentiments representation and integration will be one of our scope of the investigation to incorporate Thai textual input to the model.

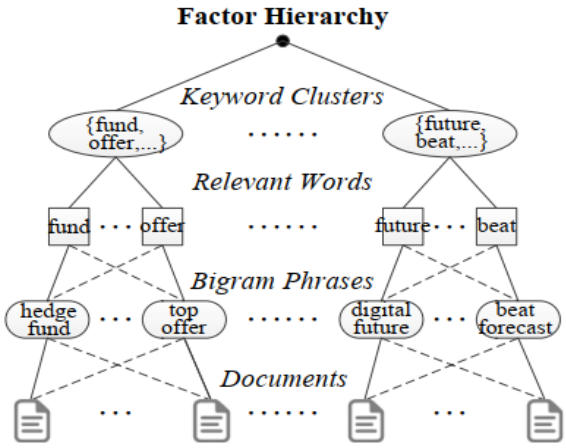


Figure 6 Hierarchy structure representation for textual news input reference from [11]

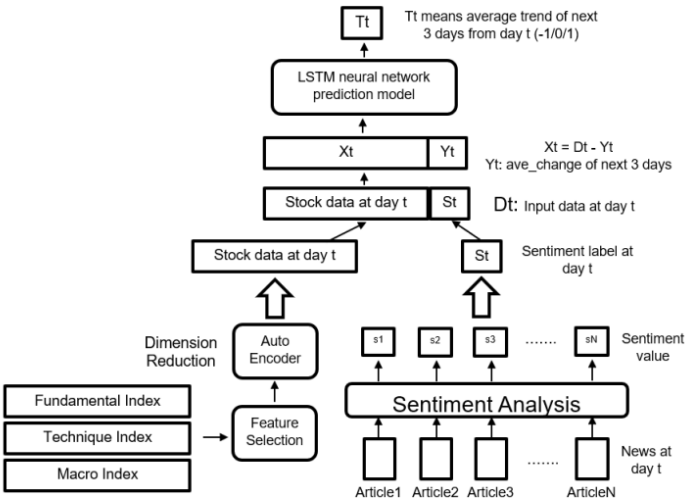


Figure 7 Sentiments feature integration with numerical input for stock trend prediction task, reference from [31]

CHAPTER 4

PROPOSED FRAMEWORK

Our proposed framework aims to improve the performance of multiple stock returns prediction using both time-series inputs and textual features together with the stock relation inference. The framework, as shown in Figure 8, starts with the data stream from numerical and textual data, then proceeds for preprocessing with multiple techniques and finally feeds to a prediction model with stock relation inference.

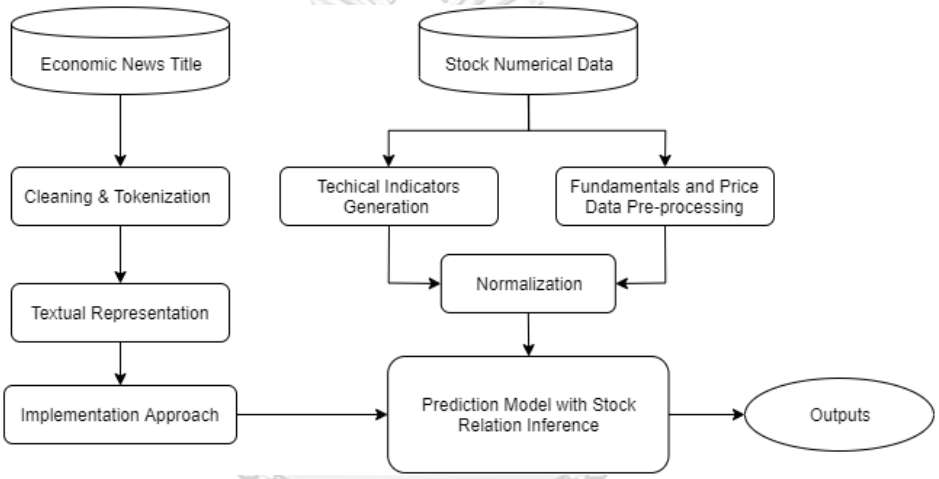


Figure 8 Proposed Framework

4.1 Data Preprocessing

4.1.1 Numerical time-series data

4.1.1.1 Fundamental and price data

The fundamentals data for each stock are transformed with forward-filling if they are quarterly updated to be consistent with other daily frequency data. Also, there are seven fundamentals attributes that were represented in other forms, namely, in the percentage of changes from last quarter, the percentage of changes from last year, and cumulative value since the beginning of the year.

Table 3 describes details of all 52 attributes of fundamental and price data used in this study.

Table 3 Fundamental and price data summary

<table>
<tr><th>Attribute Name</th><th colspan="4">Description</th><th>Count</th></tr>
<tr><td>A/P Turnover</td><td colspan="4" rowspan="7">Seven attributes presented in 5 forms below
· Q - at the quarter data
· Cum. Q - cumulative quarter value since the first day of the year
· QoQ % - percent change from the previous quarter
· YoY % - percent change from the previous year, same quarter
· YoY Cum. - percent change from YE data (cumulative)</td><td rowspan="7">35</td></tr>
<tr><td>D/E Ratio</td></tr>
<tr><td>Fixed Asset</td></tr>
<tr><td>Shareholder Equity</td></tr>
<tr><td>Total Asset</td></tr>
<tr><td>Total Liability</td></tr>
<tr><td>Total Revenue</td></tr>
<tr><td colspan="5">Attribute Name: Quarter data</td><td rowspan="3">6</td></tr>
<tr><td>Cash Cycle Period</td><td>Net Profit Margin</td><td colspan="2">Net Profit</td></tr>
<tr><td>Earnings per Share</td><td>Return of Asset</td><td colspan="2">Return of Equity</td></tr>
<tr><td colspan="5">Attribute Name: Daily data</td><td rowspan="4">11</td></tr>
<tr><td>Close Price</td><td>Open Price</td><td>Stock Trade Volume</td><td>P/E Ratio</td></tr>
<tr><td>High Price</td><td>Book Value</td><td>Transaction Volume</td><td>P/BV Ratio</td></tr>
<tr><td>Low Price</td><td>Market Value</td><td>Market Capital</td><td></td></tr>
</table>

4.1.1.2 Technical indicator generation

For the stock ranking, we adopt a list of indicators proposed in [9] then generated them with our proposed period from short to long terms: (5, 7, 10, 14, 20, 30, 50, 75, 100) days. The total number of generated indicators is nine-periods multiply by 17 time-series from 15-indicators (the MACD provides three series) equals 153 features.

Table 4 shows full list of our technical indicator features. We applied this 15 techinical indicators generation for every single stock in our target stocks.

Table 4 List of 15 technical indicators generated with nine periods for individual stocks

RSI	EMA	TripleEMA	MACD	CMFI
William%R	SMA	CCI	PPO	DMI
WMA	HMA	CMO	ROC	PSI

For the textual architecture experiments, we generated technical indicators from the "ta" package [33] with default settings which results in 73 time-series. The reason we introduce a different set of technical indicators was due to feedback received from our first publication. The reviewer commented that multiple period generations (5, 7, 10, 14, 20, 30, 50, 75, 100) days as shown in thesection 4.1.1.2 might provide indifferent features; thus, we add variety in types of technical indicators instead of the variety in the period for this textual architecture experiments. Table 5 shows the name of all technical indicators.

Table 5 List of technical indicators with the default parameter setting within "ta" package for the market index in textual architecture experiments

Trend	PSAR	Williams R	Chaikin Money Flow
MACD	Ichimoku	Stochastic Oscillator	Ease Of Movement
ADX	KST	Awesome Oscillator	Force Index
Aroon	**Momentum**	**Volatility**	Negative Volume Index
CCI	KAMA	Average True Range	On Balance Volume
DPO	MFI	Bollinger Bands	Volume Price Trend
EMA	ROC	Donchian Channel	**Other**
Mass Index	RSI	Keltner Channel	Cumulative Return
TRIX	TSI	**Volume**	Daily Log Return
Vortex	Ultimate Oscillator	Acc Dist Index	Daily Return

4.1.1.3 Data Normalization

Standardization is applied to the input features because each of them has a different range of values. The z-score normalization formula is as follows.

$$z = \frac{(x-\mu)}{\sigma} \tag{4}$$

Where μ is the mean of the input x, and σ is the standard deviation of the input x. The calculation of both σ and μ is within the validation and training dataset only to avoid our model observation on the distribution of the testing dataset.

4.1.2 Textual data

This section describes textual data preprocessing for hierarchical representation architecture, including tokenization, word embedding, and data labeling for sentiment information. For the BERT, we did not utilize this tokenization and word embedding but applied the default method from the pre-train BERT model.

4.1.2.1 Tokenization

On our Thai stock news title data, we utilize the existing pre-trained Thai word tokenizers (e.g., "Attacut," "newmn"). Additionally, we propose three enhancements using the regular expression (regex) to preprocess before this tokenization.

(i) Trading symbol extraction

We have found that we could not appropriately tokenize some documents because usually, the news expresses the trading symbol without space next to the Thai character. So we perform a symbol extraction before using the tokenization Table 6 below shows an example of this regex enhancement. We could straightforwardly achieve this extraction since we know all the list of trading symbols.

Table 6 Trading Symbols Extraction before Tokenization

Original	ไล่หุ้นKTBวอลุ่มกระฉูดราคาแลกการ์ดเป้า21บ.
Attacut	ไล่|หุ้น|KTBวอลุ่มกระฉูด|ราคา|แลก|การ์ด|เป้า|21บ.
Regex + Attacut	ไล่|หุ้น|KTB|วอลุ่ม|กระฉูด|ราคา|แลก|การ์ด|เป้า|21บ.

(ii) Remove "Hyphen," keep the "Minus" sign

We have found that removing or splitting text with unique character could fundamentally improve Thai stock news tokenization, as shown in Table 7.

Table 7 Removing Hyphen "-" before Tokenization

Original	พาณิชย์'ลุ้นข่าวดีไก่สดไทยเกาหลียุ่นปินส์จ่อเลิกแบน-
Attacut	พาณิชย์|'|ลุ้น|ข่าว|ดี|ไก่|สด|ไทย|เกาหลี|ยุ่น-|-|ปินส์|จ่อ|เลิก|แบน
Attacut + remove "-"	พาณิชย์|'|ลุ้น|ข่าว|ดี|ไก่|สด|ไทย|เกาหลี|ยุ่น|ปินส์|จ่อ|เลิก|แบน

However, when we apply the same condition to another title where the “-“ character represents the minus sign, this could lead to the different contexts of the title, as shown in Table 8, the context of adverse market is gone when removing the “-“ symbol.

Table 8 Adversarial effect when removing "-" on the minus sign

Original	ปิด1,694.39จุด-3.22จุด
Attacut	ปิด|1|,|694|.|39|จุด|-|3|.|22|จุด
Attacut + remove “-“	ปิด|1|,|694|.|39|จุด|3|.|22|จุด

(iii) Numerical value with comma and period

Some numerical entities of the news title are split by the “ . ” or the “ , ” as shown in Table 8. The number 1,694.39 was tokenized incorrectly to the number 1 number 694 and 39 of which the context of the quantifying value is misleading. We identify those symbols with regular expression and remove them in between the digits.

4.1.2.2 Word Embedding

On word representation and embedding, we use the “thai2fit” pre-trained model to obtain the word embedding representation. This thai2fit was developed from ULMFit [34] with implementation from fast.ai. They pre-trained a language model with 60,005 embeddings on the Thai Wikipedia corpus.

4.1.2.3 Documents labeling for sentiment features construction

One approach to label the document is to read the news title and to label them manually. Consequently, it will result in high success and correct sentiment. However, this requires much human effort to go through all of the datasets. As suggested in [32], the author labels the document automatically using some market feedback, which is not perfect but fast and straightforward to implement. We adopted this similar approach and labeled the group document that occurred on the same day to the next day market return changes: On the day the news is published, the Δp is calculated with Equation 5.

$$\Delta p = (p_t - p_{t-1}) \tag{5}$$

Where p_t is today close price and p_{t-1} is yesterday's close price. We apply the Δp as an absolute sentiment instead of 0 or 1 to emphasize the magnitude of the sentiments. Here is an example to clarify the process; there are 200 news occurs on a single day, all of this 200 news will be group as textual feature to the model with the label of % return change calculated from that day and the following day close price.

4.1.3 Data labelling for stock return prediction

The ground truth for next step prediction is a 1-day return ratio for the following day, as shown in Equation 6.

$$y_t^i = \frac{(p_{t+1}^i - p_t^i)}{p_t^i} \tag{6}$$

Where y_t^i is the return ratio for the stock i at time step t and p_t^i is the close price of the day t while p_{t+1}^i is the close price of the next day.

4.2 Proposed Prediction Model

Our proposed model aims to simultaneously predict sets of stock returns with the relation inference between stocks as well as textual input integration. The model structure consisted of four parts: (i) base neural network, (ii) stock relation inference, and (iii) textual representation architectures (iv) textual feature integration.

4.2.1 Base Neural Network

We select the state-of-the-art attention model for time series predictions called “Dual-Stage Attention-Based Recurrent Neural Network (DA-RNN)” as a base neural network to enhance feature and temporal relevance. Our core deep learning network is a modification from the original work of Qin [7], as shown in Figure 9. We added a batch normalization [35] layer before the Softmax layer in the input attention layer, as shown in to enhance attention weights calculation.

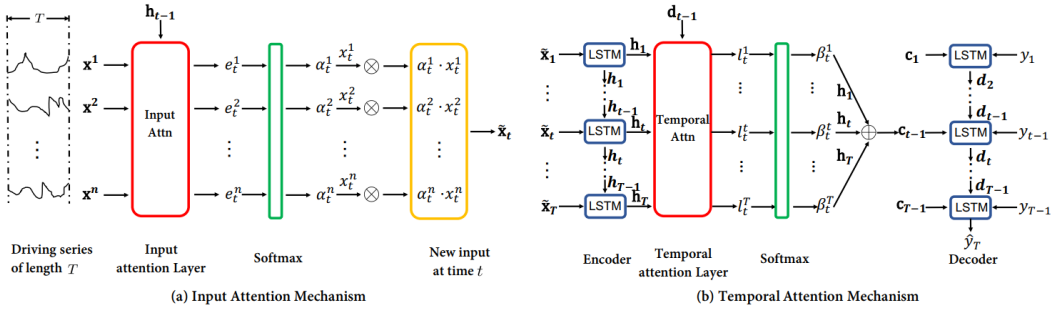


Figure 9 Dual-Stage Attention Recurrent Neural Network (DA-RNN) diagram, reference in [13]

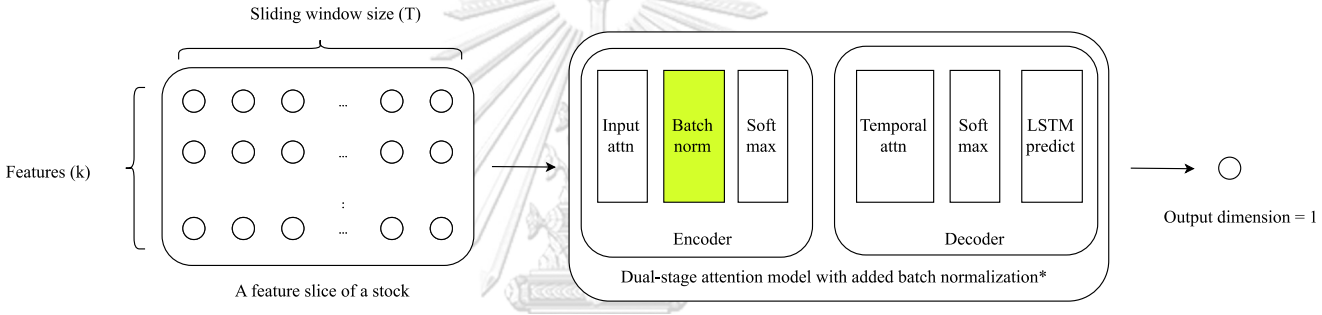


Figure 10 A simplified DA-RNN with added Batch Normalization

4.2.2 Stock Relation Inference

This second part of the model's purpose is to infer stock relations during model training. We impose two methods (i) Fixed-batch training for a shared-parameter model and (ii) Pair-wise ranking loss.

4.2.2.1 Fixed-batch training for a shared-parameter model.

Fixed-batch training is a design we adopted from [6] to achieve ranking loss while training a model. This design fixed the size of a training data batch to equal the number of target stocks. Thus, the model can simultaneously train stock data within the same period and calculate a ranking loss. Also, with this setting, the weights of models are shared among all target stock rather than we construct multiple models separately per stock. We called this a shared-parameter model. Figure 9 shows the proposed fixed batch transformation. A slice of single stock's features has a dimension of T x k, where k is the number of time-series features for each stock (e.g., technical indicators, financial parameter series), and T is the sliding window for those features. We prepare this slice for each stock within our target N stocks. This collection of N feature slices is size-equivalent to the training batch size and represents multiple stock information during the same period. All N stocks share the same weight in the modified DA-RNN model, Figure 11 (b), as a result of our fixed batch size setting. These model's shared weights are updated when the model observes all N slices of stock features in a batch during training.

There are three benefits to this design. First, it favors the ranking loss calculation, which we will cover in the next section. Second, the model becomes universal from the shared-parameter among target stocks concept, with the ability to predict particular stock independently. To be more specific, the model treats individual stock as one separate set of features within a training batch. The trained model could still predict any stock without the need to retraining the whole model again when any stock ceases to trade in the market. Finally, it reduces the model weights per data by a factor of target stocks. The lower model weights imply faster training and more straightforward to converge for the solution.

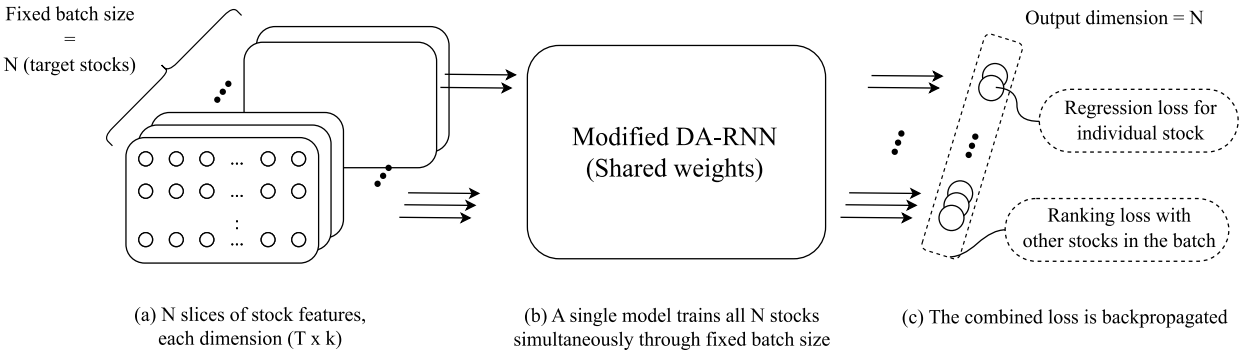


Figure 11 Diagram of the proposed model; (a) input features slices (b) a modified DA-RNN unit, the model's weights are shared among all stocks (c) combination of loss functions

4.2.2.2 Pair-wise Ranking Loss

We use a combination of regression loss and ranking loss to optimize our model. On the regression part, the widely used mean square error loss (MSE) is selected for the model to focus on the return prediction accuracy. This MSE loss calculation for stock i^{th} is displayed in Equation 7. Next, the pair-wise ranking loss is introduced to infer stock relations among all target stocks with their relative ranking score. The formula in Equation 8 calculates the relative ranking error for every pair in the matrix. Finally, the combined loss for both functions in Equation 9 is backpropagated to the model when learning a fixed batch size data Figure 11 (c).

$$MSE\ loss^i = \left(\hat{y}_t^i - y_t^i\right)^2 \quad (7)$$

$$Pairwise - Ranking\ loss = \sum_{i=0}^{N} \sum_{j=0}^{N} \max\left(0, -(\hat{y}_t^i - \hat{y}_t^j)(y_t^i - y_t^j)\right) \quad (8)$$

$$Combined\ Loss = \frac{\sum_{i=0}^{N} MSE\ loss^i}{N} + \alpha(Pairiwise - Rankingloss) \quad (9)$$

Where N is the number of target stocks to be predicted simultaneously, the $\hat{y}_t^i$ is the predicted return for stock i at time step t, the y_t^i is the label describes in the equation (1), and α (alpha) is a weighting ratio tradeoff between the regression accuracy and the ranking accuracy, which is one of the hyperparameters to be tuned.

4.2.3 Textual Representation Architectures

This section discusses two different textual representation architectures. The hierarchical neural network structure implemented following the DeepClue paper [11] and BERT aggregated embedding method. The experiment will test which architecture performs best for Thai text integration. We will perform this experiment on the market index (SET index) instead of among the target stocks because of the limited resource to distinguish news headlines for each particular stock.

4.2.3.1 The hierarchical neural network structure

We customed the architecture from the DeepClue paper [11], of which the author designed to optimize interpretation with vector representation from the word to bigram, title, and daily news representation level. We used the "Newmn" Thai language tokenization from pyThaiNLP frameworks and replaced the word2vec embedding with thai2fit embedding [36], a pre-trained Universal Language Model (ULMFiT) [34] on Thai Wikipedia corpus. Figure 12 shows the architecture for the hierarchical textual representation.

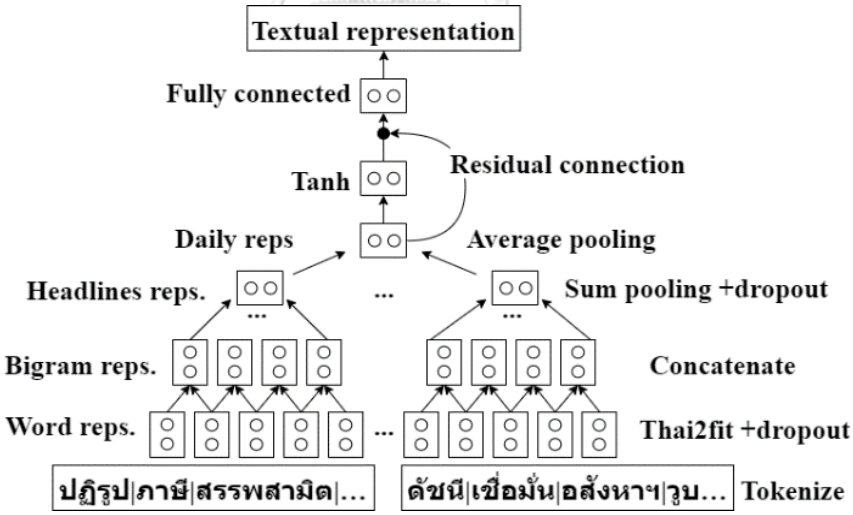


Figure 12 Modified hierarchical structure for textual representation

The hierarchical structure handles the uneven number of headlines per day with the average pooling layer at daily news title representation as well as the uneven number of words per headlines with the sum pooling layer at the news title representation.

The architecture to experiment on the market index return prediction is shown in Figure 13. We concatenate the output representation from the textual and numerical side together, each of the vectors has a dimension of [1 x hidden unit size] results in a vector of [2 x hidden unit size] before the going through the final prediction layer. We use only one day of textual news in this experiment because the model will be use to construct sentiment feauture for multiple days later in the full loop model.

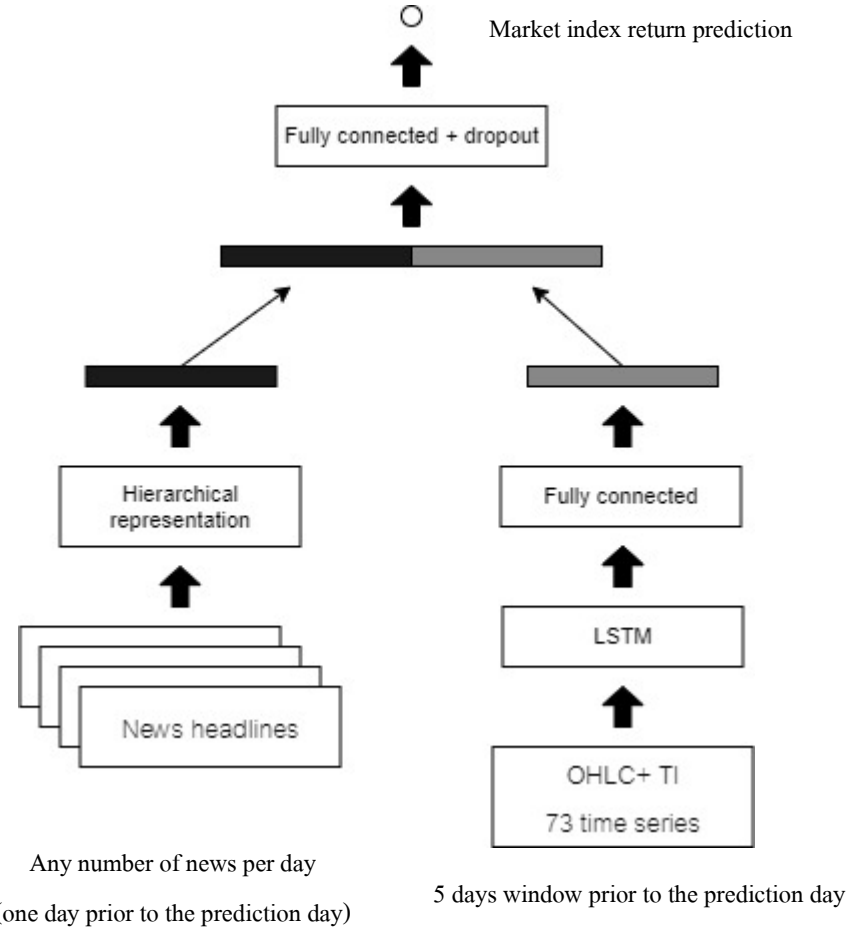


Figure 13 Hierarrchical representation for architecture experiments on the market index prediction

4.2.3.2 BERT aggregated embedding approach

The basic idea of using BERT is to transform words or token (a group of text characters) into numerical representation value. We select this BERT since it is becoming one of the best language models available in the NLP researches, outperforms many tasks and baselines. We utilized the pre-train weights of BERT-Base Multilingual Based (mBERT) available from Google research to fine-tune the language model on our 885K news headlines corpus. (Thai only BERT show more unsatisfactory perplexity results). After fine-tuning them for three epochs, we input each of the news headlines into the BERT model. Then the output becomes the embedded information (a vector size of [number of token or words x 768]), We select the first token (pooled hidden vector) [1 x 768] which often used for classification task to represent a news headline, The process is illustrated in Figure 14.

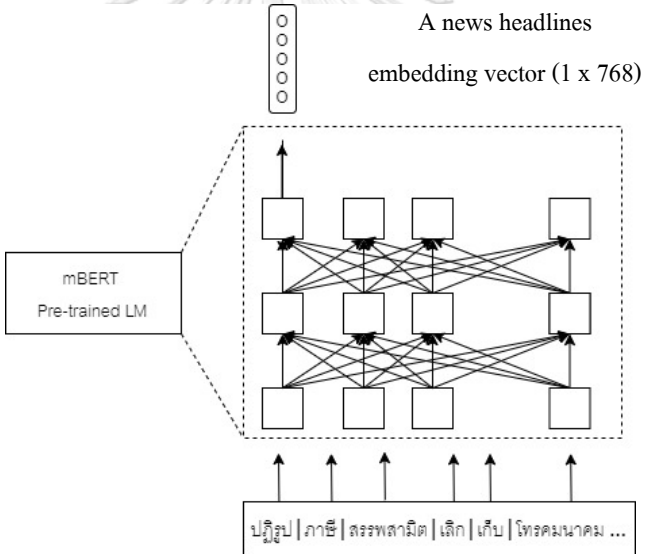


Figure 14 Embedding the news headline with pre-trained mBERT

Next, we proposed an aggregation method to handle the resource constraints on the GPU (Memory) because there are multiple news occurred per day that needed to be processed (up to 1275 in our datasets). As shown in Figure 10, We aggregate all the news headlines embedding vector occurred on the same day with either summation, averaging or maximum, results in a single day news headline BERT aggregated representation. This method might not be perfect because the backpropagation process of deep learning did not update the parameters within the mBERT embedding model.

Similarly to the hierarchical structure, we transform the BERT aggregated representation to a desire dimension using a fully connected layer. Then concatenate them with the numeric data side, each of the vectors has a dimension of [1 x hidden unit size] results in a vector of [2 x hidden unit size] before going through the final prediction layer. Figure 16 shows the architecture for the market prediction experiments.

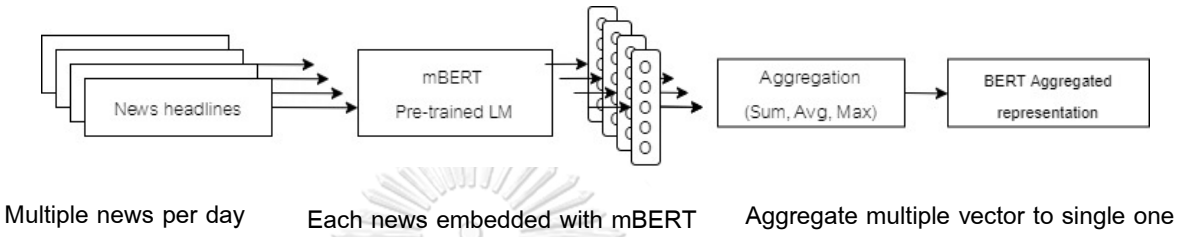


Figure 15 Aggregation process for multiple news to represent single day textual embedding

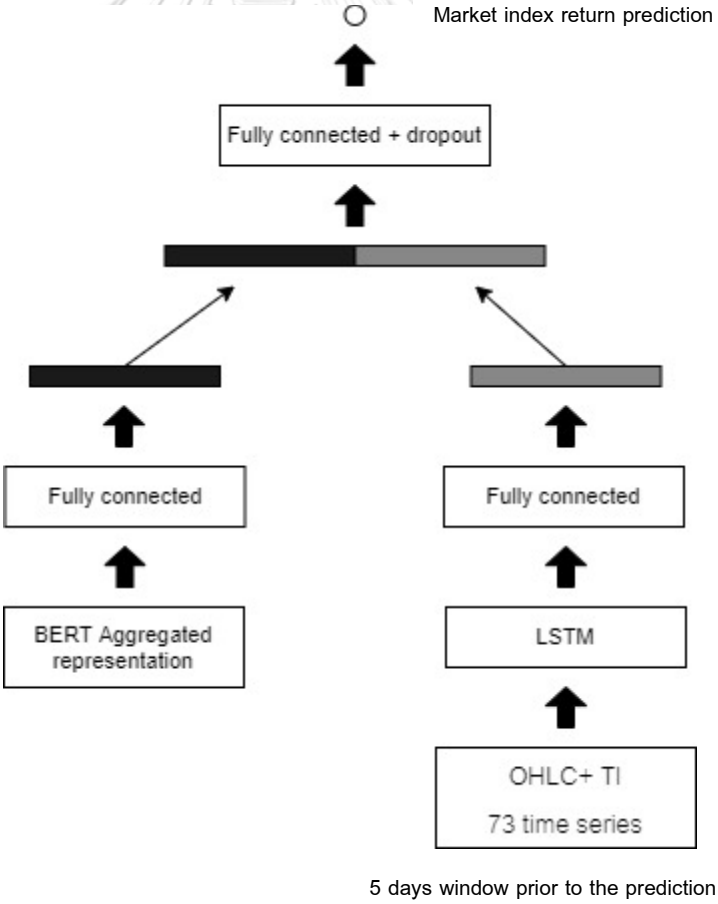


Figure 16 BERT aggregated representation for architecture experiments on the market index prediction

4.2.4 Textual Features Integration

The final part of the proposed model is to introduce textual features and integrate them with a dual-stage attention model. There are two elements to this part (i)*Textual Features Representation Method* and (ii) *Implementation approaches.* We select the best performed architecture from section 4.2.3 (in this case, hierarchical structure) to test the integration with the stock relation inference model (DA-RNN).

4.2.4.1 Textual Features Representation Method

We proposed to investigate textual features representations in two main areas. First, the full embedding representation, and second, the sentiment features representation.

1. The first full embedding representation embed all news titles in a single day to a vector using the hierarchical neural network structure [11] and feed directly to the return prediction model (DA-RNN), as shown in the workflow Figure 17. Figure 18 shows that the hierarchical neural network generates a textual representation vector at each time steps results in a vector dimension [window x hidden size]. The parameter weights within the hierarchical neural network structure will also be updated during full loop training.

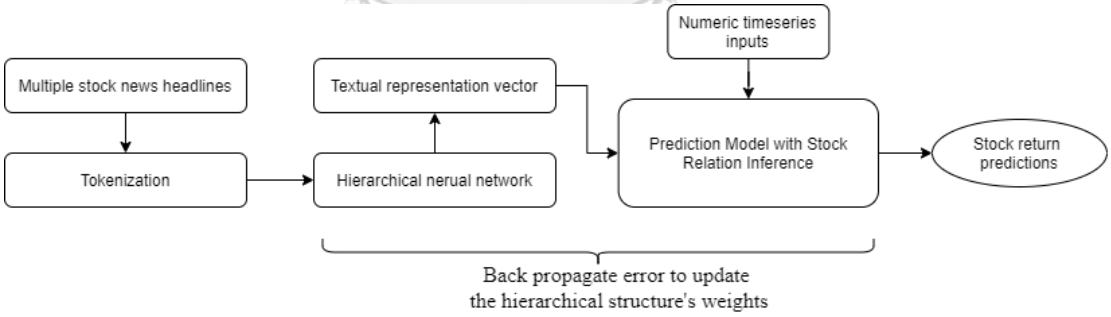


Figure 17 Full embedding representation

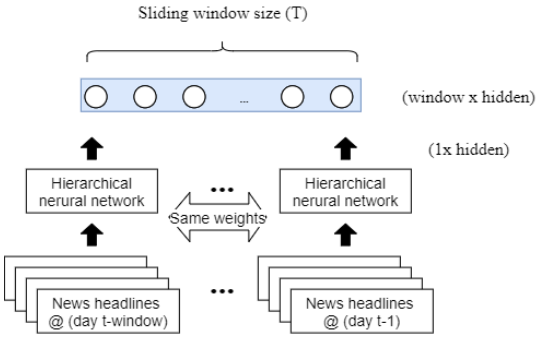


Figure 18 Hierarchical neural network structure embedding news each day for dual-stage attention model

2. The second textual representation aims to construct a static sentiments value for each time step. We utilized the trained hierarchical neural network from our architecture experiments to convert all one-day stock news headlines into a single vector representing daily sentiments information. This vector will be a static input for the DA-RNN. Figure 19 illustrates this process. We utilized the best model from section 4.2.3 to generate a textual representation for each time step, shown in Figure 20.

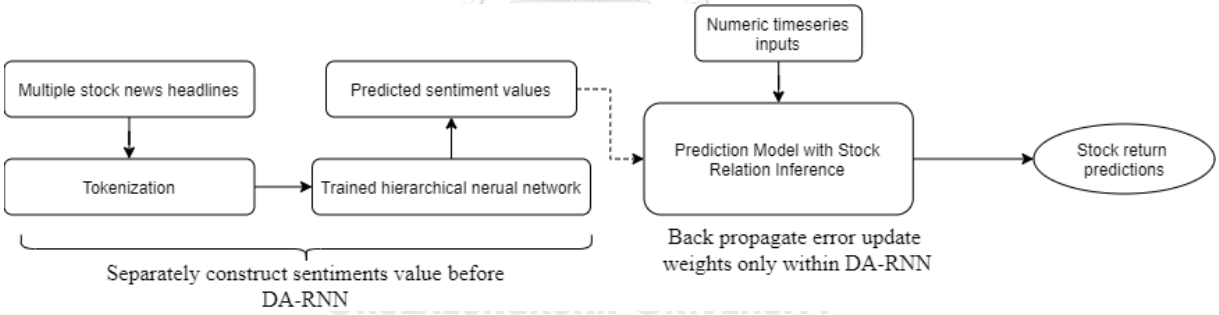


Figure 19 Sentiments features construction representation

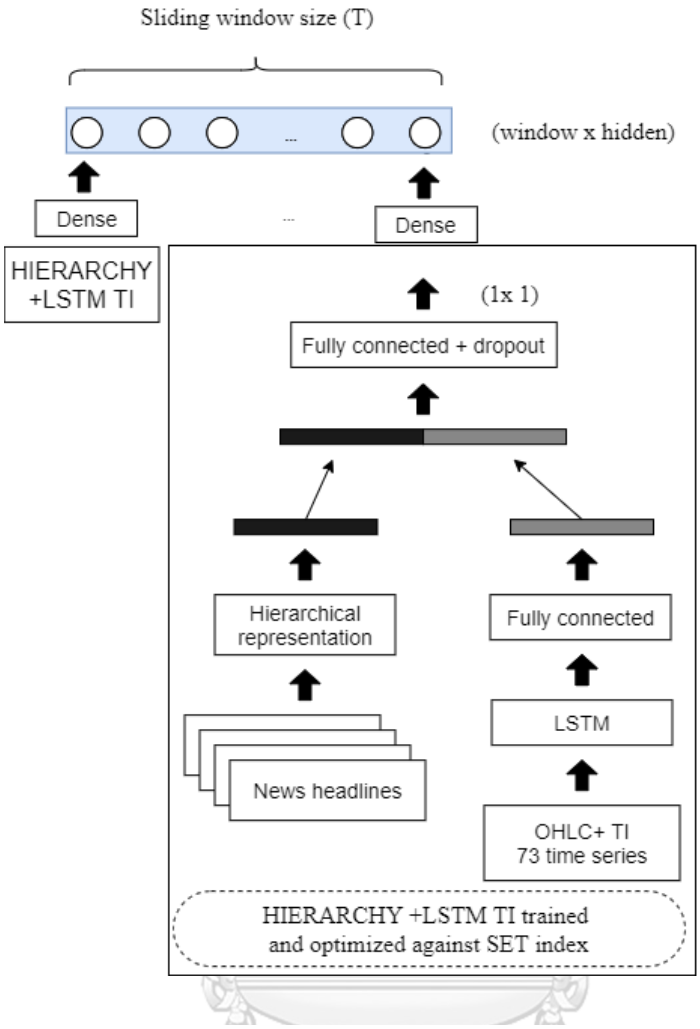


Figure 20 Hierarchical with LSTM techinical indicator (Best from architecture experiments) generates market sentiments at each time step

4.2.4.2 Implementation Approaches

After we prepared the textual representation, they will be implemented and tested by four approaches to the model, as described in Figure 21. We investigated the effectiveness of each method as discussed the result in the latter section. The hypothesized summary for them is described in Table 9. Figure 22 to Figure 25 shows the detail for each integration method.

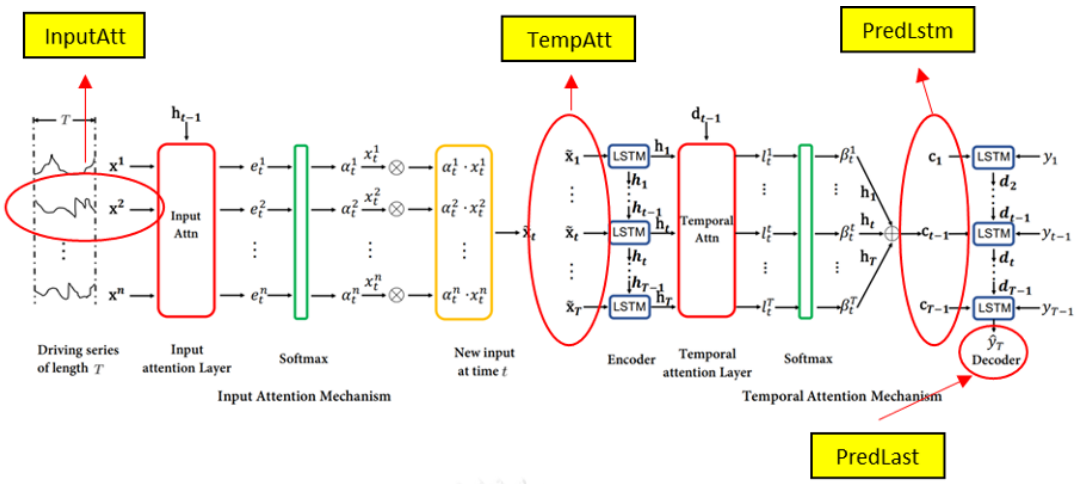


Figure 21 Four proposed approaches to implement textual features: 1) InputAtt at the beginning 2) TempAtt before the temporal attention mechanism and 3) PredLstm before the LSTM prediction layer, 4) PredLast before the final dense prediction layer

Table 9 Textual Implementation Approaches Summary

#	Name	Location	Pros	Cons
A	InputAtt	Before input attention	Attention weight for textual	Textual info might vanish with numerical feature
B	TempAtt	After input attention, Before temporal attention	Textual info get the benefit of temporal attention	No input attention for textual
C	PredLstm	After temporal attention, Before LSTM prediction	Temporal for textual in LSTM, multiple days of textual info	Lacks, attention layer for a textual info
D	PredLast	After LSTM prediction, Before the final dense layer	Previous day textual info get emphasized, Align with most papers	An only single day of textual information

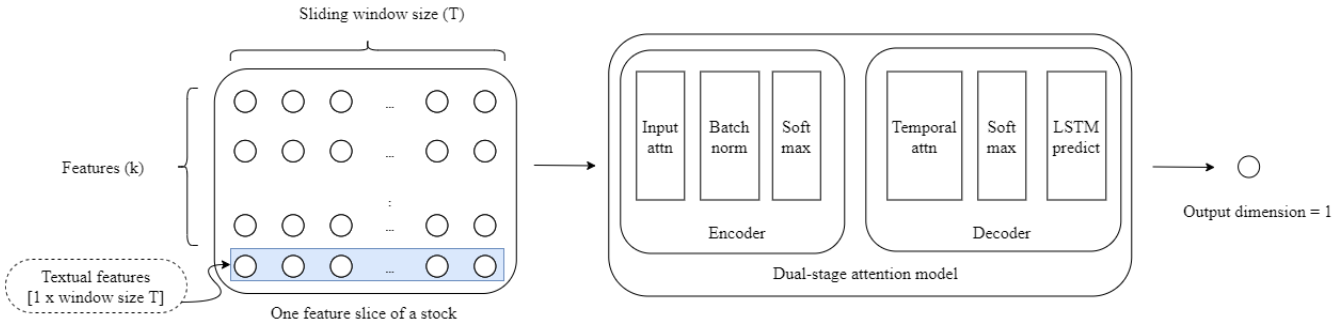


Figure 22 InputAtt: Add textual features before input attention

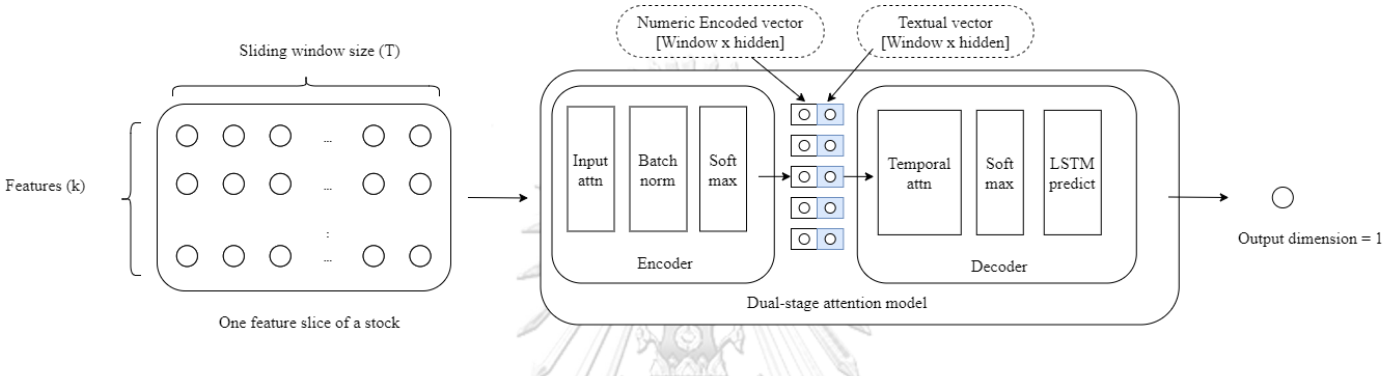


Figure 23 TempAtt: Add textual features before temporal attention

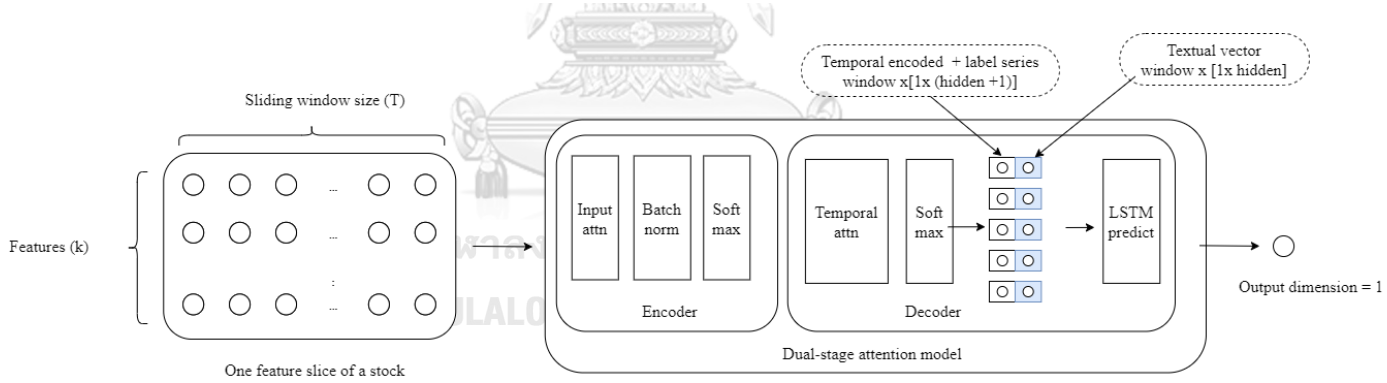


Figure 24 PredLstm: Add textual features before LSTM prediciton

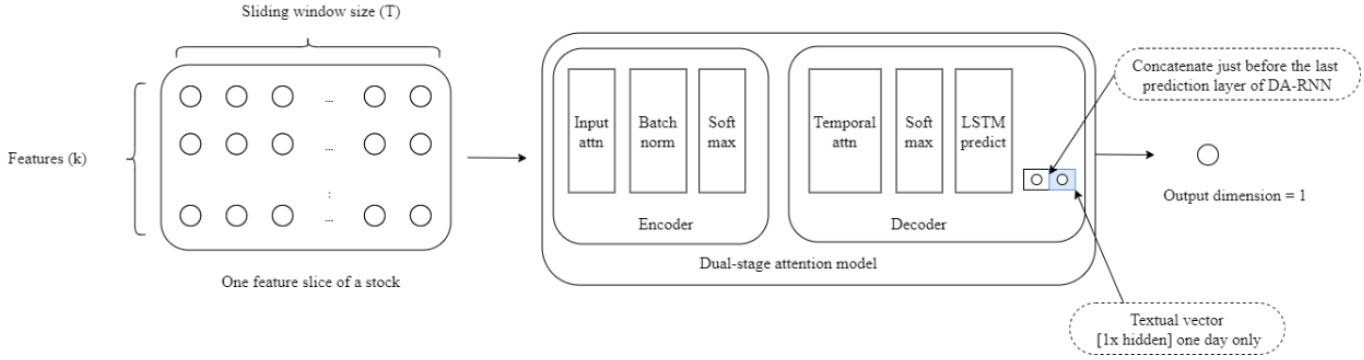


Figure 25 PredLast: Add textual features before final dense prediction and use only a single day's news

CHAPTER 5

EXPERIMENTAL SETUP

5.1 Dataset

The stock's end of the day (EOD) numerical data used in our research is from The Stock Exchange of Thailand (SET) Market, corresponding to the period from 1st of January 2008 to 28th December 2018. We use daily data due to computational and data limitations. The numeric data includes price information such as Open, High, Low, Close, Volume as well as the calculated fundamental value such as Book value, P/E Ratio.

Next on the textual features, we collected only the economic news topic (our news source explicitly group the economic news in one section) and only for the news title (headline) from various online data sources corresponding to the EOD data periods; approximately 885K news headlines for this researches. Initially, we aim to categorize news for each target stock by explicitly filter the news which contains the stock symbols in the title. There is 3.4 % of total news, which is tagged with our 64 target stocks, and another 5.9% of total news contains other stock symbols. Lastly, 90.6% of total news contains no trading symbols in its title. The tagged news was approximately 30K headlines, not covering every trading day. With this limited resource to tag news for all 64 stocks efficiently, we decided to use all economic news to represent the market instead.

The detail data statistic will be included in each experiment because not all experiments share the same scope of the data such as the textual architecture experiment where we predict the market return until the year 2019.

5.2 Target Stocks Pre-selection

There are three considerations for our selection of target stock principles. With the below criteria, we pre-select 64 target stocks out of the SET100 index.

1. Stock information availability through training to testing periods
2. Sufficient liquidity to assume order always get filled
3. Sufficient volume and big market cap, to avoid price manipulation and to assume that our trading effect on the price can be neglected

Table 10 List of our 64 target stocks

Industry Group	Stock symbols	Count
Agro & Food Industry	[CPF, MINT, TVO, TU, STA, GFPT]	6
Financials	[BBL, TCAP, KBANK, SCB, TMB, KKP, KTB, KTC, THANI]	9
Property & Construction	[SCC, TPIPL, TASCO, UV, LH, QH, BLAND, AP, SPALI, LPN, CPN, SIRI, AMATA, STEC, ITD, CK]	16
Resources	[BANPU, PTTEP, BCP, EGCO, IRPC, RATCH, PTT, TOP, SUPER, GLOW]	10
Services	[BJC, ROBINS, HMPRO, CPALL, ERW, CENTEL, BH, BDMS, BCH, BTS, THAI, PSL, AOT, MAJOR, RS, WORK]	16
Technology	[KCE, HANA, DELTA, INTUCH, ADVANC, TRUE, DTAC]	7
		64

5.3 Evaluation Metrics

We compare the performance of each model with three measures:

5.3.1 Root Mean Square Error (RMSE)

Standard evaluation of the regression task on the predicted return is as follows:

$$RMSE_t = \sqrt{\sum_{i=1}^{n} \frac{\left(\hat{y}_t^i - y_t^i\right)^2}{n}} \quad (10)$$

Where $\hat{y}_t^i$ is the predicted return at day t for the stock ith and y_t^i is the ground truth of the return at day t for stock i. Lastly, n is the total of target stocks (64 stocks for this research).

5.3.2 Mean Reciprocal Ranking for Top stock(MRR-Top)

We use MRR to evaluate the model on the ranking performance of the top stock (stock with the highest predicted return); the detail of MRR calculation is described in the background knowledge section.

5.3.3 Profit from Trading Simulation

We select the daily buy-hold-sell strategy with fixed investment (e.g., buy stock worth 1,000 dollars daily). The details as follows:

- At day t, run the model to predict returns for all target stocks then rank those predicted returns to select only the top stock to buy with fixed investment.
- At the day t+1, sell the stock bought from day t at the close price of day t+1.

This strategy assumes that the trading volume is always sufficient to satisfy buying or selling at the close price. We neglect the fee in the metric. However, we can recalculate percent profit after fee with Equation 11.

$$\%Return_{after\ feed\ during\ t\ days} = \frac{-2\times t\times fee+(1-fee)\times\sum_{i=1}^{i=t}\%Return_i}{1+fee} \quad (11)$$

Where fee is the commission fee per transaction (both buy and sell action) t is the number of trading occurred.

5.4 Baseline Models

This section proposes baseline methods and experiments for performance comparison.

5.4.1 Traditional trading

These are market and asset baseline with “Buy and hold” strategy.

- SET Index: buy & hold for the SET market index
- SET 100 Index: buy & hold SET100 Index
- SET 64 Index: buy & hold for our 64 target stocks equally in investment (e.g., 10,000 dollars per stock)

5.4.2 Neural network based models

- Artificial Neural Network (ANN)

Basic implementation of ANN with a dense layer, Relu activation function, and a linear layer

- Long-short term memory (LSTM)

A single layer vanilla LSTM, with Relu activation function and a linear layer

- Dual-Stage Attention (DA-RNN)

A modified DA-RNN as described in section 4.2

5.4.3 Relation inference performance (proposed framework for stock ranking)

All neural network-based models will be tested with the proposed relation inference, as described in section 4.2.2. The model with relation inference (ranking ability) will contain suffix "RANK" such as DA-RANK is Dual-stage attention RNN with relation inference.

5.4.4 Textual features integration

The DA-RNN with and without relation inference will be tested with textual features on two mains topic as describes in detail in the section 4.2.3 and 4.2.4.

1. Textual features representations: Sentiment representation vs. Full new embedding using the hierarchical neural network structure, details in section 4.2.3.
2. Textual integration approach: Four locations of the DA-RNN model, as describes in section 4.2.4, will be experimented to find the best approach.

5.5 Model Training and Hyperparameters Tuning

Generally, on all experiments: we optimized our model with the Adaptive Moment Estimation (Adam) algorithm with an initial learning rate of 0.001. Next, a grid search for hyperparameter was applied to the range of parameters as follows: hidden unit (4, 8), window size -T (5), and regression-ranking tradeoff: Alpha - α (0, 0.1, 1, 10, 100, 1000).

The model with the best MRR for the top stock in the validation dataset will be selected to test on the test set. We will explicitly describe within the experiment if the setting deviates from this section.

CHAPTER 6

EXPERIMENTS AND RESULTS

This section describes all the experiments as well as discussion on the results. The chapter is separated into six sections consists of (i) Numeric based experiments, (ii) Stock grouping by industry experiments (iii) Textual representation architecture experiments (iv) Textual integration experiments (v) Final comparison.

6.1 Numeric Based Experiments

This first experiment aims to improve stock return prediction and ranking ability of the existing model. We introduce relation inference, as mentioned in section 4.2.2. The experiments consist of a neural network, LSTM, and DA-RNN as the baseline.

6.1.1 Dataset and Partitioning

Data partitioning in these experiments was slightly different from the latter experiments due to the data available at that time. We use 64 target stock information corresponding to the period from 12th February 2008 to 28th December 2018. The total trading days during the studied period are 2655 days and are split into three sets, as summarized in Table 11. With 64 targets stocks in our scope, the total training, validating, and testing data for the model are approximately 92,000/31,500/15,000 records per period, respectively.

We perform the out of sample testing using a sliding window, as shown in Table 12, results in training data for eight years, validating and testing one year each. We split the data into three datasets, as suggested by [24] to evaluate the performance of the model through time.

Table 11 Numerical data records summary (trading days)

		Trading days		
No.	**Data period**	**Training**	**Validating**	**Testing**
1	Feb-2008 to Dec-2016	1,437	509	222
2	Jan-2009 to Dec-2017	1,464	487	243
3	Jan-2010 to Dec-2018	1,464	488	244

Table 12 Data splitting for the numeric based experiment

No.	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
1	Training						Validating		Test		
2		Training						Validating		Test	
3			Training						Validating		Test

6.1.2 Additional Implementation Details

We trained models with three different sets of random seed in the first experiment as the DA-RANK will be a core model for further experiments, and the stock prediction has high uncertainty.

6.1.3 Results

The experiment results in Table 13 show performance across three datasets, and Table 14 shows average performance across three datasets. The DA-RANK (DA-RNN with ranking) on some test results shows poorer profit than the LSTM but still able to produce the highest MRR score for top stock ranking with significantly better RMSE. Noted: the suffix "RANK" refers to the relation inference (ranking loss implementation). Also, we could observe the discrepancy between model regression accuracy and profit in the ANN model, where it shows the best RMSE while significantly weaker profit. On a three-year average, the DA-RANK shows the best performance on profit and MRR-Top metric.

Furthermore, Figure 26 shows that the relation inference (RANK) can improve the MRR metric on all neural network structures, except LSTM in the year 2017 dataset. Figure 27 shows that it also improves the profit of most of the models. Overall the DA-RANK shows the most promising results comparing to other baselines.

Table 13 Numeric based results (best MRR-top in the validation across three random seed)

	Profit			MRR-Top			RMSE		
Model	**2016**	**2017**	**2018**	**2016**	**2017**	**2018**	**2016**	**2017**	**2018**
ANN	51.8%	-2.3%	9.4%	0.110	0.075	0.108	0.030	0.030	0.022
ANN-RANK	74.3%	-44.7%	-28.3%	0.109	0.078	0.116	**0.025**	**0.027**	**0.021**
LSTM	**93.1%**	6.5%	-20.1%	0.129	0.099	0.092	0.137	0.075	0.046
LSTM-RANK	89.3%	13.8%	10.1%	0.134	0.089	**0.126**	0.154	0.080	0.090
DA-RNN	30.4%	43.6%	22.9%	0.124	0.092	0.107	0.033	0.037	0.052
DA-RANK	88.5%	**45.8%**	**54.0%**	**0.154**	**0.112**	0.116	0.029	0.029	0.044
SET	20.0%	12.2%	-12.1%						
SET100	20.2%	14.9%	-11.4%						
SET64	25.8%	19.1%	-16.0%						

Table 14 Numerical based results (average three-years)

Model	Profit	MRR-Top	RMSE
ANN	19.6%	0.098	0.0275
ANN-RANK	0.4%	0.101	**0.0242**
LSTM	26.5%	0.107	0.0861
LSTM-RANK	37.7%	0.116	0.1081
DA-RNN	32.3%	0.108	0.0405
DA-RANK	**62.7%**	**0.127**	0.0339
SET	6.7%		
SET100	7.9%		
SET64	9.6%		

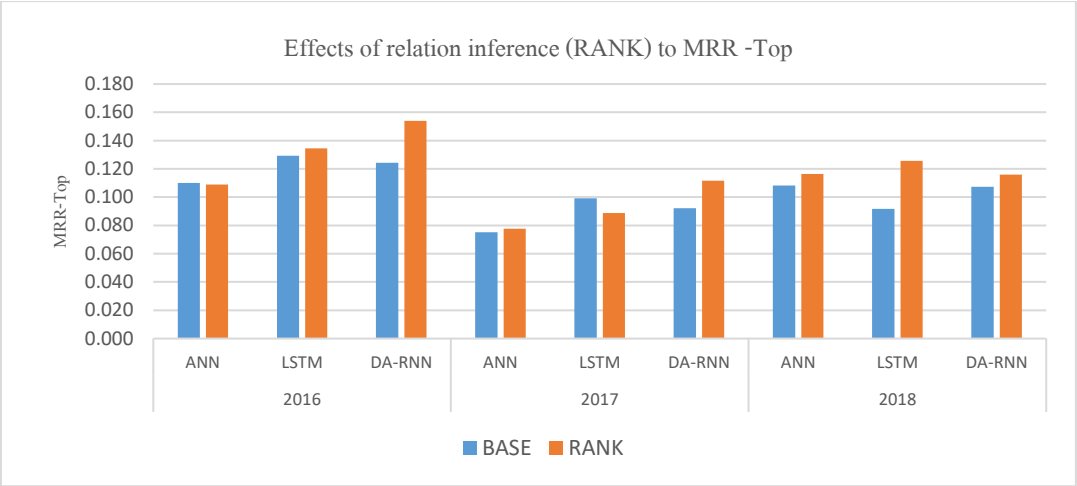


Figure 26 Effectiveness of relation inference to neural network models on MRR-top

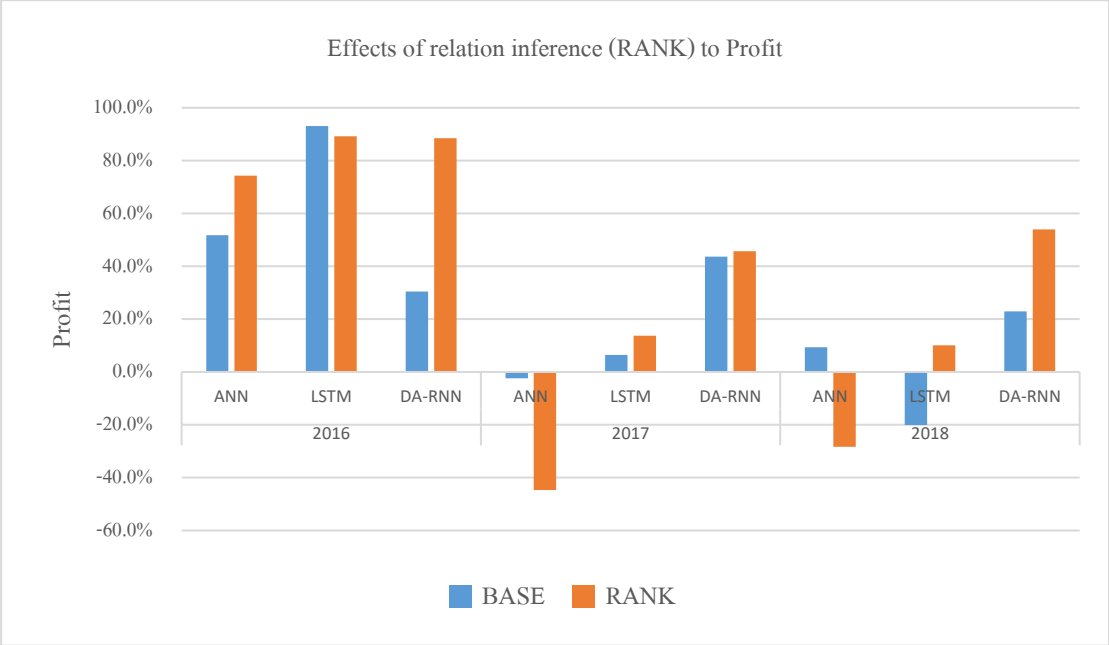


Figure 27 Effectiveness of relation inference to neural network models on profit

6.2 Stock Grouping by Industry Experiments

Stock within the same industry tends to behave similarly, and grouping them could improve model performance as researched in [5]. This experiment's objective is to test the DA-RANK performance when grouping stocks with its industry section. We categorized our 64 target stocks into six industries, as shown in Table 15. Next, we construct one DA-RNN per industry

group instead of the original single DA-RNN for all 64 stocks, the industry grouping model, as illustrates in

Figure 28. This structure still preserves fix batch training of 64 stocks but will have fewer data per model weights ratio (increasing number of hidden units). We will call this industry grouping with the suffix “IND” after the model name.

Table 15 Industry grouping for 64 target stocks

Industry Group	Industry symbol	Count	Stock symbols
Agro & Food Industry	AGRO	6	[CPF, MINT, TVO, TU, STA, GFPT]
Financials	FINCIAL	9	[BBL, TCAP, KBANK, SCB, TMB, KKP, KTB, KTC, THANI]
Property & Construction	PROPCON	16	[SCC, TPIPL, TASCO, UV, LH, QH, BLAND, AP, SPALI, LPN, CPN, SIRI, AMATA, STEC, ITD, CK]
Resources	RESOURC	10	[BANPU, PTTEP, BCP, EGCO, IRPC, RATCH, PTT, TOP, SUPER, GLOW]
Services	SERVICE	16	[BJC, ROBINS, HMPRO, CPALL, ERW, CENTEL, BH, BDMS, BCH, BTS, THAI, PSL, AOT, MAJOR, RS, WORK]
Technology	TECH	7	[KCE, HANA, DELTA, INTUCH, ADVANC, TRUE, DTAC]

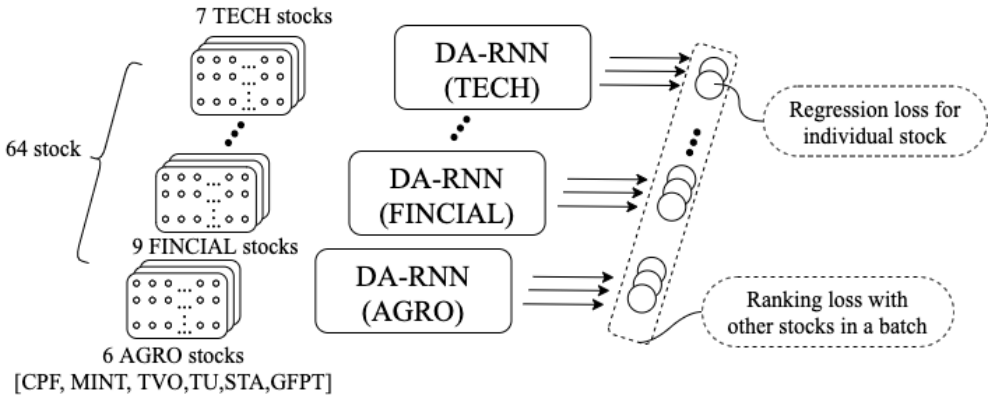


Figure 28 DA-RANK-IND: DA-RNN with industry grouping and stock relation inference, each DA-RNN observes stock, particularly in the same industry group.

6.2.1 Dataset and partitioning

The dataset for the stock ranking task (64 target stocks predictions) will be different from the previous experiments, we change the validating set to be one year only, and minor adjust the data period as describe in Table 16 and Table 17.

Table 16 Dataset statistic for industry grouping experiments

		Trading days		
No.	**Data period**	**Training**	**Validating**	**Testing**
1	Jan-2008 to Dec-2016	1,711	243	244
2	Jan-2009 to Dec-2017	1,707	244	244
3	Jan-2010 to Dec-2018	1,708	244	245

Table 17 Data split for industry grouping experiment

No.	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
1	Training							Validating	Test		
2		Training							Validating	Test	
3			Training							Validating	Test

6.2.2 Results

Table 18 shows comparisons across three years between industry grouping performance on DA-RANK with the original model. We could see that MRR-Top of the DA-RANK-IND is better than DA-RANK in the year 2016 and 2018. However, the results in Table 19 shows that, on average, the industry grouping method did not show any significant improvement over the original DA-RANK. However, we see a drop in regression accuracy (high RMSE). The industry grouping does not improve the DA-RANK may be due to the reduction of data records the model observes. Each DA-RNN in DA-RANK-IND observes only 7-16 stocks compared to the original one in which learns 64 stocks altogether. Finally, Figure 29 and Figure 30 show that relation inference could also improve the industry grouping method.

Table 18 Industry grouping results (best MRR-top in validation dataset)

	MRR-Top			Profit			RMSE		
Model	**2016**	**2017**	**2018**	**2016**	**2017**	**2018**	**2016**	**2017**	**2018**
DA-RNN	0.1101	0.0909	0.0829	81.9%	**68.2%**	-42.7%	0.115	**0.048**	0.093
DA-RANK	0.1141	**0.1103**	0.0943	**91.2%**	46.4%	13.0%	**0.063**	0.078	**0.064**
DA-RNN-IND	0.1086	0.0733	0.0700	49.5%	-24.6%	6.0%	1.378	0.913	0.965
DA-RANK-IND	**0.1169**	0.0995	**0.1018**	59.1%	23.8%	**46.4%**	0.237	0.379	3.144

Table 19 Industry grouping results (average three years)

Model	MRR-Top	Profit	RMSE
DA-RNN	0.0946	35.8%	0.0852
DA-RANK	**0.1062**	**50.2%**	**0.0682**
DA-RNN-IND	0.0840	10.3%	1.0853
DA-RANK-IND	0.1060	43.1%	1.2533

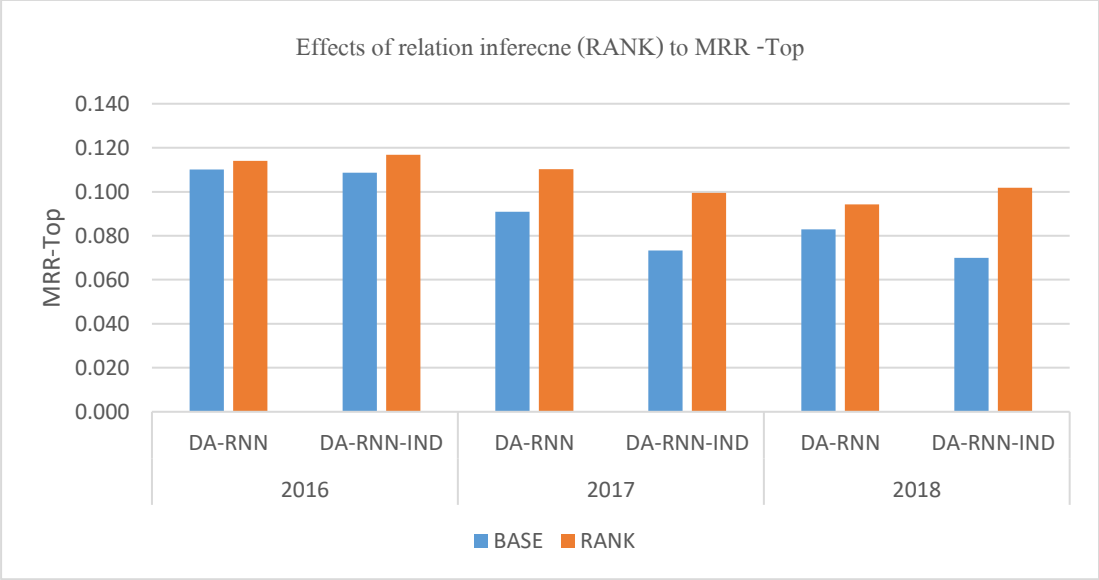


Figure 29 Effectiveness of relation inference to industry grouping on MRR-top

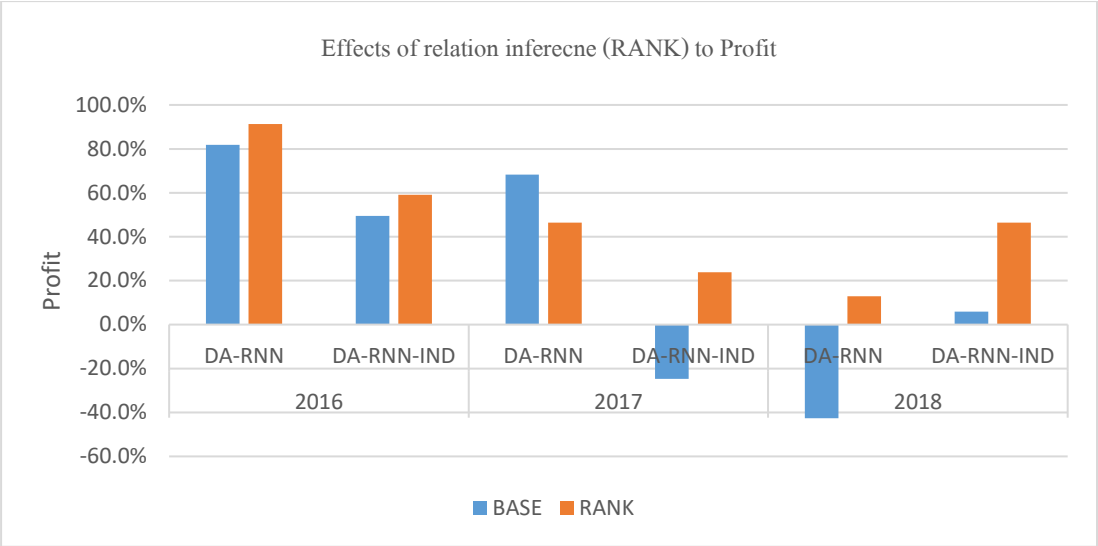


Figure 30 Effectiveness of relation inference to industry grouping on profit

6.3 Textual Representation Architecture Experiments

This architecture experiment aims to explore which model is suitable to represent news headline input for the stock prediction task. However, due to the constraints on news data, we could not tag the news efficiently for each target stocks (can be done via keywords or manual labeling). We have tried using the stock symbols to extract news, but it results in significantly reduced in the amount of data from 885K news down to only 30-40K news. Moreover, the news with specific stock symbols often contains less context than other general news. As a result, we do these experiments on the SET market index instead to identify which architecture could capture textual context to the market index prediction. In section 4.2.3, we described two architecture, the hierarchical neural network, and BERT embedding approach. The model objective in this experiment is to predict the next day return of the SET market index with a combination of numeric and textual features. We select the basic LSTM to handle the numeric time-series parts. Figure 31 illustrates the full model in this experiment.

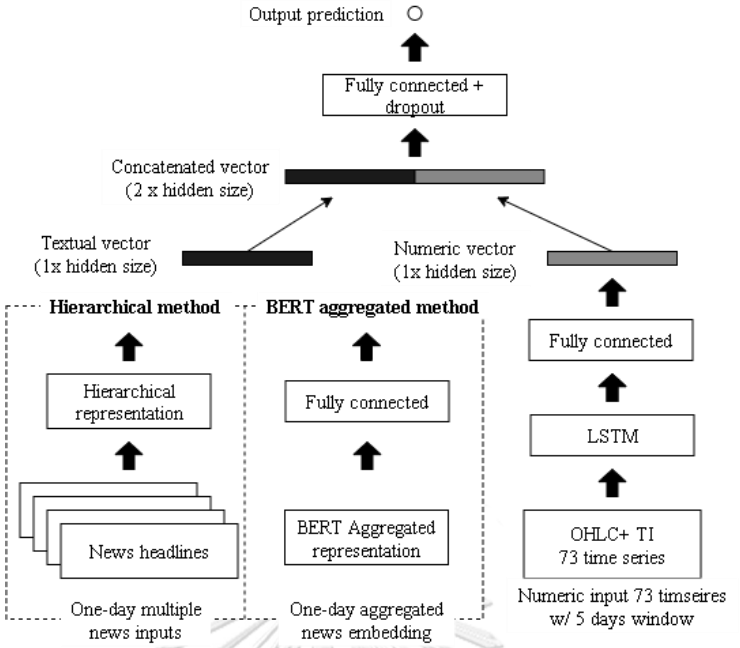


Figure 31 The combination of textual representation and LSTM handling numeric features (price and indicator)

6.3.1 Dataset and partitioning

This experiment contains 2019 year data in addition to other experiments because we update the data to deliver this result within the CMRI scholarship competition. Data statistics and data split are illustrated in Table 20 and Table 21, respectively.

Table 20 Data statistic for textual architecture experiments

		News records			Trading days		
No.	**Data period**	**Training**	**Validating**	**Testing**	**Training**	**Validating**	**Testing**
1	Jan-2008 to Dec-2017	600,220	75,425	73,923	1,952	244	244
2	Jan-2009 to Dec-2018	631,296	73,923	72,686	1,949	244	247
3	Jan-2010 to Dec-2019	604,764	72,686	62,756	1,950	247	244

Table 21 Data split for textual architecture experiment

No.	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
1	Training								Validating	Test		
2		Training								Validating	Test	
3			Training								Validating	Test

6.3.2 Evaluation metrics

Because the model objective is different from multiple stock predictions and ranking tasks, this section contains different metrics to evaluate textual architecture. Three metric used are as follows:

a) *Root mean square error (RMSE):* Standard evaluation to the predicted return

b) *Market direction prediction %accuracy:* This metric evaluates if the predicted %return is correct and aligns with the market direction. For example: if tomorrow the actual SET index returns +0.1%, but the model's prediction is negative at -0.2%. We count this as a miss (wrong direction) even the regression error is low. Vice versa, if actual SET index returns +1.9% and the model predicts +0.1%, we count this as a hit (correct trend). We describe the trend accuracy in Equation 12 and 13.

$$Accuracy = \frac{\sum_{t=1}^{t=n}(h_t)}{n} \tag{12}$$

$$h_t = \begin{cases} 1 & \text{if } \hat{y}_t \text{ and } y_t \text{ are both positive or negative,} \\ 0 & \text{other wise,} \end{cases} \tag{13}$$

Where h_t is the hit count when our predicted direction is correct, n is the total number of predictions.

c) *Hit profit*: This metric as shown in Equation 14 is the summation of all the absolute "hit" returns minus the all absolute "miss" returns for the testing periods. We propose this as the primary evaluation metric because it represents trade-strategy-independent profit.

$$Hit\ profit = \sum_{t=1}^{t=n}(2h_t - 1)\, y_t \tag{14}$$

Where h_t is the hit count when our predicted direction is correct. n is the total number of predictions and y_t is the actual return of the SET index at the time "t".

6.3.3 Baseline Models

We proposed comparing methods and baselines listed below.

a) *Random prediction model (RANDOM)* To benchmark the model on randomness, we randomly predict the SET index return each day (1000 simulations) using the historical distribution before the testing period.

b) *Numerical input only (LSTM)* Baselines to evaluate the model performance when using only numerical time-series. Also, the time-series used are OHLC (4 time-series) and OHLC+TI (73 time-series) to evaluate the performance of the model when supplementing with technical indicators. OHLC stand for Open, High, Low, Close data and TI stand for the technical indicators.

c) *Textual input only (HIERARCHY or BERT)* these baselines contain only the textual representation neglecting the time-series featured LSTM part. Instead of textual representation vector, HIERARCHY and BERT will output regression value directly.

d) *Combined method (HIERARCHY or BERT +LSTM TI)* These two are proposed models with combined textual and numeric features inputs, but different in the textual representation.

6.3.4 Implementation details

We optimized the deep learning model using the Adaptive Moment Estimation (Adam) algorithm and mean square error as a loss function for the regression task. The hyperparameters are as follows; dropout probability (0.0, 0.5); BERT aggregations (sum, avg, max); the numerical feature window size is five days and hidden unit (16,64). The hierarchical embedding hidden unit required the hidden unit equals the pre-trained word2vec embedding, which is 300 in this paper (thai2fit). We test the model with the best validation loss on the best validating epoch.

6.3.5 Results

The overall results in Table 22 show the model test performance across three years. On the RMSE, all deep learning models produce indifferent error around 0.58-0.59%. Next, On the accuracy, the LSTM with numerical features (OHLC + TI) delivers highest at 53.4%. Our model (HIERARCHY/BERT + LSTM TI) performs not so well on the trend accuracy. We could argue that the accuracy metric did not translate directly to more profit; for example, we investigate the BERT, and LSTM OHLC then found that the model only predicts one class either all negative or all positive trends. These reflect on the hit profit metric, both BERT and LSTM OHLC show less

profit even the BERT has accuracy at 52.5%. On the other hand, for hit profit, our proposed model significantly outperforms other models (HIERARCHY/BERT + LSTM TI), at 10.1%, and 5.8% hit profit.

Table 22 Textual architecture experiments result on three-year split

	RMSE				Accuracy (%)				Hit profit			
Model	**2017**	**2018**	**2019**	**Avg.**	**2017**	**2018**	**2019**	**Avg.**	**2017**	**2018**	**2019**	**Avg.**
RANDOM	0.0148	0.0160	0.0152	0.0154	50.2%	50.0%	50.1%	50.1%	0.2%	-0.4%	0.3%	0.0%
LSTM(OHLC)	0.0041	0.0076	0.0059	0.0059	44.3%	51.0%	52.5%	49.2%	-13.0%	-8.8%	4.6%	-5.8%
LSTM(OHLC+TI)	0.0041	0.0076	0.0059	**0.0059**	55.3%	49.4%	**55.3%**	**53.3%**	10.5%	-11.2%	14.4%	4.6%
BERT NEWS	0.0041	0.0076	0.0060	0.0059	**55.7%**	51.0%	50.8%	52.5%	**13.0%**	-10.7%	-1.4%	0.3%
BERT NEWS + LSTM TI	0.0042	0.0076	0.0059	0.0059	48.0%	48.2%	52.9%	49.7%	-4.4%	1.4%	20.2%	5.8%
HIERARCHY NEWS	**0.0041**	0.0076	0.0059	0.0059	52.9%	49.4%	51.2%	51.2%	8.0%	-11.2%	12.9%	3.2%
HIERARCHY NEWS + LSTM TI	0.0043	**0.0075**	**0.0058**	0.0059	47.5%	**52.2%**	54.1%	51.3%	-2.3%	**12.4%**	**20.3%**	**10.1%**

Effect of feature type for market prediction

The performance shows that OHLC features are not sufficient. The LSTM(OHLC) shows poor performance at only -5.8% hit profit. The technical indicator improves the hit profit to 4.6%. Using only textual features (BERT and HIERARCHY) could not succeed with the LSTM(OHCL+TI). On the other hand, we found a significant hit profit gain when we combined both types surpassing the LSTM(OHLC+TI). The last column in Table 23 shows that adding numerical features over textual model improves hit profit by +5.5%, +6.9% for the BERT, and HIERARCHY, respectively.

Table 23 Input type effectiveness (three-year average)

Type	Model	Hit profit	Compare w/ LSTM TI	Compare w/text only
Numeric	**LSTM(OHLC+TI)**	4.6%	-	-
Textual	**BERT**	0.3%	-4.3%	-
	HIERARCHY	3.2%	-1.4%	-
Numeric + Textual	**BERT + TI**	5.8%	+1.2%	+5.5%
	HIERARCHY + TI	**10.1%**	**+5.6%**	**+6.9%**

Effect of textual representation architecture

On the effectiveness of the textual representation architecture, we found that the HIERARCHY shows better results than BERT at around 3.0-4.4%. Table 24 shows the comparison results. This improvement can be because the embedding layer of the HIERARCHY is optimized and updated during the training process. Meanwhile, the BERT provide static representation for the textual feature.

Table 24

Representation	Model	Hit profit	Difference
BERT	**BERT**	0.30%	-
	BERT + TI	5.80%	-
HIERARCHY	**HIERARCHY**	3.20%	+3.0%
	HIERARCHY + TI	**10.10%**	**+4.4%**

6.4 Integration Experiments

This section utilized the hierarchical neural network, which shows the best performance in section 6.3. We use it to represent textual features before integrating with the DA-RNN. The four integration processes are described in section 4.2.4.

6.4.1 Dataset and partitioning

We experiment only in the year 2016 dataset due to the number of scenarios is tremendously large to be explored on all set of hyperparameters. We will evaluate the integration approach on the validation set only then the best approach will be selected to run in the full test for on the 2016-2018 year data set. Table 25 and Table 26 below shows the trading days and data split with news headline records corresponding to the same period.

Table 25 Data statistic for integration experiments

		News records		**Trading days**	
No.	**Data period**	**Training**	**Validating**	**Training**	**Validating**
1	Jan-2008 to Dec-2015	523,748	76,472	1,711	243

Table 26 Data split for integration experiments

No.	2008	2009	2010	2011	2012	2013	2014	2015
1	Training							Validating

6.4.2 Implementation details

We optimized our model with the Adaptive Moment Estimation (Adam) algorithm with an initial learning rate of 0.001. Next, a grid search for hyperparameter was applied to the range of parameters as follows: hidden unit (4, 8), window size -T (5), and regression-ranking tradeoff: Alpha - α (0, 0.1, 1, 10, 100, 1000). The hierarchical embedding hidden unit required the hidden unit equals the pre-trained word2vec embedding, which is 300 in this paper (thai2fit). We use the best architecture from the previous section called HIERARCHY + LSTM TI to generate sentiments features. We tune the model against the SET index and select the best performed model in the validation year (avoid model to observe test data). The integration approach with the best MRR for the top stock in the validation dataset will be selected further.

6.4.3 Baselines and list of model

From section 4.2.3, there are two types of representation, a full textual embedding using the hierarchical structure and a sentiment construction approach, together with four integration methods from section 4.2.4 that result in a total eights model to experiment. Table 27 list the name for implementation references.

Table 27 Model convention for integration experiments

Representation / Integration	Method A	Method B	Method C	Method D
Full hierarchical embedding (EMB)	InputAtt_EMB	TempAtt_EMB	PredLstm_EMB	PredLast_EMB
Sentiments construction (SENT)	InputAtt_SENT	TempAtt_SENT	PredLstm_SENT	PredLast_SENT

We describe each model as follows:

- InputAtt: the textual features integrate early just before the input attention layer.
- TempAtt: the textual features integrate before the temporal attention layer.
- PredLstm: the textual feature integrates before the LSTM prediction layer.
- PredLast: the textual feature integrates before the final dense prediction layer (single-day news).

6.4.4 Results

The results in Table 28 below shows the performance for each model within the validation dataset. It is observed that the model with textual integration could not outperform DA-RANK (numeric based only) except the PredLast_SENT, which provides MRR-Top at 0.134. Table 29 shows that, on average, the sentiment construction provides better results than the full embedding. We suspect that the sentiment feature is better because, during the sentiment construction, the technical indicator of the SET market is included during the model training. Next, in Table 30 shows integration methods PredLast and PredsLstm, performs better than other methods which integrate textual information quite early. Consequently, we select the integration method PredLast and PredLstm with both sentiments and full embedding to be tested further on the three datasets.

Table 28 Representation and integration performance on validation dataset

Integration	Representation	NAME	MRR-Top	Profit	RMSE
InputAtt	Full embedding	InputAtt_EMB	0.116	42.0%	0.2620
	Sentiment	InputAtt_SENT	0.111	45.0%	0.1070
TempAtt	Full embedding	TempAtt_EMB	0.110	26.0%	0.1080
	Sentiment	TempAtt_SENT	0.121	87.0%	3.3120
PredLstm	Full embedding	PredLstm_EMB	0.122	37.0%	0.2780
	Sentiment	PredLstm_SENT	0.120	66.0%	**0.0630**
PredLast	Full embedding	PredLast_EMB	0.119	41.0%	0.0640
	Sentiment	PredLast_SENT	**0.134**	31.0%	0.1650
Numeric only		DA-RANK	0.129	**103.0%**	0.0670
		DA-RANK-IND	0.108	26.0%	0.4250

Table 29 Representation approach average performance

Representation	MRR-Top	Profit	RMSE
DA-RANK	**0.129**	**103.0%**	**0.0670**
Sentiment	0.122	57.3%	0.9118
Full embedding	0.117	36.5%	0.1780

Table 30 Integration approach average performance

Integration	MRR-Top	Profit	RMSE
DA-RANK	**0.129**	**103.0%**	**0.0670**
PredLast	0.127	36.0%	0.1145
PredLstm	0.121	51.5%	0.1705
TempAtt	0.116	56.5%	1.7100
InputAtt	0.114	43.5%	0.1845

6.5 Final Experiments Comparison

In this final comparison, we refer to some of the models in experiments 6.1, 6.2, and 6.4. We trained all models on the same three datasets, which testing for the years 2016, 2017, and 2018. Then compare the performance on the proposed metric. Finally, at the end of this section, we illustrate a simple method to optimize the profit when it includes the commission fee.

6.5.1 Dataset and partitioning

The final testing consists of three datasets with testing period year 2016, 2017, 2018 and sliding window as already mentioned from the previous step.

Table 31 Data statistic for the final comparison

		News records			Trading days		
No.	**Data period**	**Training**	**Validating**	**Testing**	**Training**	**Validating**	**Testing**
1	Jan-2008 to Dec-2016	523,748	76,472	75,425	1,711	243	244
2	Jan-2009 to Dec-2017	555,871	75,425	73,923	1,707	244	244
3	Jan-2010 to Dec-2018	530,841	73,923	72,686	1,708	244	245

Table 32 Data split for the final comparison

No.	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
1	Training							Validating	Test		
2		Training							Validating	Test	
3			Training							Validating	Test

However, we felt that the integrated textual models are not a clear cut winner over the model without textual features (DA-RANK) as some of the textual models are still weaker than DA-RANK in MRR-Top. This hindered performance might be to the fact that the news input is the same generic economic news for all stock rather than specifically individual stock's news.

Table 33 Final comparison results (best MRR-top in validation dataset)

			MRR-Top			Profit			RMSE		
Integration	**Representation**	**Model**	**2016**	**2017**	**2018**	**2016**	**2017**	**2018**	**2016**	**2017**	**2018**
PredLstm	Full embedding	PredLstm_EMB	0.104	0.117	0.103	62.8%	41.4%	62.8%	0.1200	1.5075	0.1730
	Sentiment	PredLstm_SENT	0.095	0.099	**0.107**	68.0%	16.2%	**74.1%**	0.0570	0.3162	0.1626
PredLast	Full embedding	PredLast_EMB	0.100	0.108	0.102	58.4%	43.3%	13.5%	**0.0563**	0.1269	0.1598
	Sentiment	PredLast_SENT	**0.134**	**0.118**	0.091	**124.8%**	10.0%	22.8%	0.0671	**0.0536**	0.1350
Numeric only		DA-RANK	0.114	0.110	0.094	91.2%	**46.4%**	13.0%	0.0628	0.0776	**0.0641**
		DA-RANK-IND	0.117	0.099	0.102	59.1%	23.8%	46.4%	0.2371	0.3793	3.1436

Table 34 Final comparison results (three-year average)

Integration	**Representation**	**Model**	**MRR-Top**	**Profit**	**RMSE**
PredLstm	Full embedding	PredLstm_EMB	0.108	**55.7%**	0.6001
	Sentiment	PredLstm_SENT	0.100	52.8%	0.1786
PredLast	Full embedding	PredLast_EMB	0.103	38.4%	0.1143
	Sentiment	PredLast_SENT	**0.114**	52.6%	0.0853
Numeric only		DA-RANK	0.106	50.2%	**0.0682**
		DA-RANK-IND	0.106	43.1%	1.2533

Table 35 Integration approach average performance (three-year dataset)

Integration	**MRR-Top**	**Profit**	**RMSE**
DA-RANK	0.106	50.2%	**0.0682**
PredLstm	0.104	**54.2%**	0.3894
PredLast	**0.109**	45.5%	0.0998

Table 36 Representation approach average performance (three-year dataset)

Representation	MRR-Top	Profit	RMSE
DA-RANK	0.106	50.2%	**0.0682**
Full embedding	0.106	47.1%	0.3572
Sentiment	**0.107**	**52.7%**	0.1319

6.5.3 Profit improvement, commission fee issue

The proposed profit metric was not included commission fees in the first place. Moreover, because our strategy ranks and trade stock every day, the commission is substantially high when applied with a 0.2% fee per transaction (approximately a regular fee for the SET market). We compare the profit after fee for each model in

Table 37. The results show that none of our models could beat the profit benchmark (SET, SET100, SET64) with the simple buy and hold strategy.

A basic mitigate to reduce fee is to reduce the trading frequency. Thus, we re-run all trading simulation with new frequencies. Instead of trading every single day, we test it with every 2, 5, 10, 15, 20, and 30 days (5 trading days is approximately a week, and a month for 20 trading days). We illustrate this fee optimization in Table 38. The results show that some of our models can beat the benchmark when trading every 15 trading days or more (less than 17 transactions/year). Also, DA-RANK could overcome the 9.2% benchmark reaching a maximum of 6.0% profit.

Next, Figure 32 shows the profit comparison on different trading frequencies with and without a fee. The overall sweet spot is around 15 - 20 trading days (trading every month) as the distribution spread is small, and the average profit with the fee is the highest. However, we optimized our models against only the regression accuracy and the ranking loss. Thus in order to find the best suitable strategy to a deep learning model should be a different area of research.

Table 37 Profit comparison when applying a 0.2% fee

			Profit (trade every day)				Profit (trade every day) + 0.2% fee			
Integration	**Representation**	**Model**	**2016**	**2017**	**2018**	**Avg.**	**2016**	**2017**	**2018**	**Avg.**
PredLstm	Full embedding	PredLstm_EMB	62.8%	41.4%	62.8%	**55.7%**	-34.8%	-56.1%	-35.2%	-42.1%
	Sentiment	PredLstm_SENT	68.0%	16.2%	**74.1%**	52.8%	-29.7%	-81.2%	-24.0%	-45.0%
PredLast	Full embedding	PredLast_EMB	58.4%	43.3%	13.5%	38.4%	-39.2%	-54.2%	-84.4%	-59.3%
	Sentiment	PredLast_SENT	**124.8%**	10.0%	22.8%	52.6%	**26.9%**	-87.4%	-75.1%	-45.2%
Numeric only		DA-RANK	91.2%	**46.4%**	13.0%	50.2%	-6.5%	-51.2%	-84.9%	-47.6%
		DA-RANK-IND	59.1%	23.8%	46.4%	43.1%	-38.5%	-73.7%	-51.6%	-54.6%
Market benchmark		SET	20.0%	12.2%	-12.1%	6.7%	19.5%	11.8%	-12.5%	6.3%
		SET100	20.2%	14.9%	-11.4%	7.9%	19.7%	14.4%	**-11.8%**	7.5%
		SET64	25.8%	19.1%	-16.0%	9.6%	25.3%	**18.6%**	-16.3%	**9.2%**

Table 38 Profit performance comparison with different trading frequency

	Average profit when trade every x days with 0.2% fee						
Model/ trade every:	**1**	**2**	**5**	**10**	**15**	**20**	**30**
PredLstm_EMB	-42.1%	-41.7%	6.9%	-8.8%	-3.8%	**23.4%**	-9.8%
PredLstm_SENT	-45.0%	-50.0%	4.4%	-11.4%	-15.2%	-11.5%	-9.2%
PredLast_EMB	-59.3%	-36.5%	-0.4%	-11.1%	**15.9%**	18.4%	3.7%
PredLast_SENT	-45.2%	-21.5%	-42.8%	4.7%	12.6%	-2.7%	17.7%
DA-RANK	-47.6%	-24.3%	-20.3%	-23.5%	2.5%	-11.1%	6.0%
DA-RANK-IND	-54.6%	-37.8%	-10.9%	-14.9%	6.1%	21.9%	**18.2%**
SET64 (buy/hold)	**9.2%**	**9.2%**	**9.2%**	**9.2%**	9.2%	9.2%	9.2%
					Beat SET64 benchmark		
Avg trade counts/year:	**244.3**	**122.2**	**48.9**	**24.4**	**16.3**	**12.2**	**8.1**

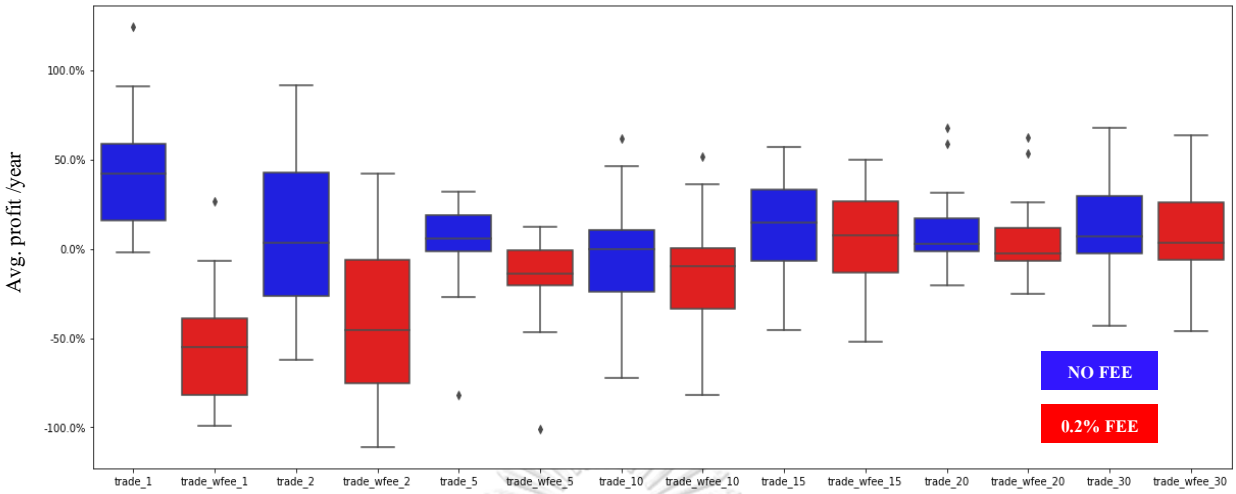


Figure 32 Distribution of profit on different trading frequency with and without fee

CHAPTER 7

SUMMARY AND FUTURE WORKS

7.1 Summary

This section concludes the thesis and all the experiments.

On the topic of stock relation inference in the numerically-based model

We have found that our tuned DA-RANK model provides the highest average returns and MRR-Top comparing to other methods with better ability to handle rich features inputs as well as stock ranking ability. Interestingly the ANN and LSTM, which are both a simpler version of the Neural Network model, made weak predictions. We can also observe a discrepancy between accuracy and the optimal profit as ANN shows the best RMSE but the weakest profit.

On the topic of industry grouping

We multiply the number of DA-RANK parameters by six times, corresponding to the number of stock's industry groups. The assumption was that a model per industry could have capture similar patterns among similar stocks better. However, we do not observe any significant improvement comparing to the DA-RANK. Finally, we found that our stock relation inference could also improve the industry grouping model, similar to that it can improve the DA-RNN.

On the topic of textual representation architecture

We design this experiment to seek for a suitable architecture to represent textual features for the stock prediction task. Moreover, we also propose a model to improve the performance of the SET index prediction using textual representation and numerical time-series. We explore the impact of input types, either textual, numeric, or both of them. The results show that combining two kinds of features for market prediction could significantly improve model performance by up to 5% hit profit. Next, we found that using the hierarchical neural network for textual representation yield better performance than the BERT aggregated embedding method at around 3-4% hit profit. The best model is the Hierarchical neural network with technical indicators showing hit profit at 10.1% (three-year average), we then use this hierarchical with technical indicators to generate sentiments features as well as using it for full embedding. We then investigate the method to integrate these textual representations with our core DA-RANK multiple stock prediction model.

On the topic of textual and numerical integration to the model with stock relation inference

There are numerous cases to be explored the textual feature integration (eight structures in our experiments). We found that on representation technique, a sentiment construction approach yield better performance than the full embedding. The sentiments were optimized, particularly against the SET market index, which is an essential factor that affects all stocks in the Thai market. Next, among the four integration approaches, the PredLstm and PredLast method, which integrates textual features at the near- end of the model, yield better results than other methods. Noted that we evaluated its performance on validation set so we could decide to select the best approach for testing in the next section.

Final comparison

Finally, the performance comparison on three datasets shows that our model PredLast_SENT, which is the dual-stage attention model with relation inference and sentiments feature integrated at before the prediction layer, performs the best. It shows the highest average MRR-Top at 0.114, with an average profit of 52.6% per year, which is better than the model without textual features (DA-RANK). However, when we include a commission fee to the trading simulation, all of our models show the negative profit and can not beats the market benchmark

(9.2% average profit per year). We improve this by reducing the trading frequency and found that trading every 15-20 days with our model prediction could beat the market benchmark. We also proposed that trading strategy optimization can be another area of improvement but were not in this research scope.

7.2 Recommendation for future works

The recommendation for future works are grouped into different section. They includes some comment from the committee during the thesis defence process

Data related

1. Intraday datasets for news data
2. More factors/features, for example, holidays, company announcements, twitter, commodity, global news, market normalization., blog (such as pantip)
3. Individual stock or industry group news tagging (improve news features preparation) to improve performance (how to tag news effectively?). Construct model per industry to automate news features relevance.
4. The rare event features construction, market or stock anomalies such as war or deceases

Trading strategy related

1. Trading/investment strategy-oriented model or optimization domain to reduce the trading fee and adopt to the practical use of the model
2. Enhance fee handling by certain criteria with prediction model
3. Evaluate best trading period to reduce fee within validation data before perform the tests

Model and objective related

5. Extend BERT model to full embedding integration
6. Quantify model's explainability. Framework for experts to feedback model explainability
7. Statistically investigate the correlation between stock for deep learning model
8. Anomaly detection task, using SD different for data labelling
9. Re-enforcement learning area
10. News feature could have different direction impact to different stock, how to construct model to capture the different

REFERENCES

1 Fama, E.F.: 'Efficient Capital Markets: A Review of Theory and Empirical Work', The Journal of Finance, 1970, 25, (2), pp. 383-417

2 Malkiel, B.G.: 'Reflections on the efficient market hypothesis: 30 years later', Financial Review, 2005, 40, (1), pp. 1-9

3 Ding, X., Zhang, Y., Liu, T., and Duan, J.: 'Using structured events to predict stock price movement: An empirical investigation', in Editor (Ed.)^(Eds.): 'Book Using structured events to predict stock price movement: An empirical investigation' (2014, edn.), pp. 1415-1425

4 Oncharoen, P., and Vateekul, P.: 'Deep Learning Using Risk-Reward Function for Stock Market Prediction'. Proc. 2018 2nd International Conference on Computer Science and Artificial Intelligence, Shenzhen,China2018 pp. Pages

5 Akita, R., Yoshihara, A., Matsubara, T., and Uehara, K.: 'Deep learning for stock prediction using numerical and textual information', in Editor (Ed.)^(Eds.): 'Book Deep learning for stock prediction using numerical and textual information' (2016, edn.), pp. 1-6

6 Feng, F., He, X., Wang, X., Luo, C., Liu, Y., and Chua, T.-S.: 'Temporal Relational Ranking for Stock Prediction', ACM Transactions on Information Systems, 2019, 37, pp. 1-30

7 Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., and Cottrell, G.: 'A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction', in Editor (Ed.)^(Eds.): 'Book A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction' (2017, edn.), pp.

8 Vargas, M.R., Lima, B.S.L.P.d., and Evsukoff, A.G.: 'Deep learning for stock market prediction from financial news articles', in Editor (Ed.)^(Eds.): 'Book Deep learning for stock market prediction from financial news articles' (2017, edn.), pp. 60-65

9 Ding, X., Zhang, Y., Liu, T., and Duan, J.: 'Deep learning for event-driven stock prediction', in Editor (Ed.)^(Eds.): 'Book Deep learning for event-driven stock prediction' (2015, edn.), pp. 2327-2333

10 Oncharoen, P., and Vateekul, P.: 'Deep learning for stock market prediction using event embedding and technical indicators', 2018

11 Shi, L., Teng, Z., Wang, L., Zhang, Y., and Binder, A.: 'DeepClue: visual interpretation of text-based deep stock prediction', IEEE Transactions on Knowledge and Data Engineering, 2018, 31, (6), pp. 1094-1108

12 Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K.: 'Bert: Pre-training of deep bidirectional transformers for language understanding', arXiv preprint arXiv:1810.04805, 2018

13 Craswell, N.: 'Mean Reciprocal Rank', Encyclopedia of database systems, 2009, 1703

14 Educba.com: 'What is Neural Network', 2017

15 Olah, C., and Carter, S.: 'Attention and Augmented Recurrent Neural Networks', Distill, 2016, 1, (9)

16 Olah, C., http://colah.github.io/posts/2015-08-Understanding-LSTMs/, accessed 2018, October 9 2018

17 Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., and Dean, J.: 'Distributed representations of words and phrases and their compositionality', in Editor (Ed.)^(Eds.): 'Book Distributed representations of words and phrases and their compositionality' (2013, edn.), pp. 3111-3119

18 Ballings, M., Van den Poel, D., Hespeels, N., and Gryp, R.: 'Evaluating multiple classifiers for stock price direction prediction', Expert Systems with Applications, 2015, 42, (20), pp. 7046-7056

19 Patel, J., Shah, S., Thakkar, P., and Kotecha, K.: 'Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques', Expert Systems with Applications, 2015, 42, (1), pp. 259-268

20 Fischer, T., and Krauss, C.: 'Deep learning with long short-term memory networks for financial market predictions', European Journal of Operational Research, 2018, 270, (2), pp. 654-669

21 Chen, K., Zhou, Y., and Dai, F.: 'A LSTM-based method for stock returns prediction: A case study of China stock market', in Editor (Ed.)^(Eds.): 'Book A LSTM-based method for stock returns prediction: A case study of China stock market' (IEEE, 2015, edn.), pp. 2823-2824

22 Nelson, D.M., Pereira, A.C., and de Oliveira, R.A.: 'Stock market's price movement prediction with LSTM neural networks', in Editor (Ed.)^(Eds.): 'Book Stock market's price movement prediction with LSTM neural networks' (IEEE, 2017, edn.), pp. 1419-1426

23 Hochreiter, S., and Schmidhuber, J.: 'Long short-term memory', Neural Comput, 1997, 9, (8), pp. 1735-1780

24 Sezer, O.B., and Ozbayoglu, A.M.: 'Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach', Applied Soft Computing, 2018, 70, pp. 525-538

25 Hollis, T., Viscardi, A., and Yi, S.E.: 'A Comparison of LSTMs and Attention Mechanisms for Forecasting Financial Time Series', CoRR, 2018, abs/1812.07699

26 Guo, T., and Lin, T.: 'Multi-variable LSTM neural network for autoregressive exogenous model', arXiv preprint arXiv:1806.06384, 2018

27 Minh, D.L., Sadeghi-Niaraki, A., Huy, N.D., Min, K., and Moon, H.: 'Deep Learning Approach for Short-Term Stock Trends Prediction Based on Two-Stream Gated Recurrent Unit Network', Ieee Access, 2018, 6, pp. 55392-55404

28 Araci, D.: 'FinBERT: Financial Sentiment Analysis with Pre-trained Language Models', arXiv preprint arXiv:1908.10063, 2019

29 Hiew, J.Z.G., Huang, X., Mou, H., Li, D., Wu, Q., and Xu, Y.: 'BERT-based Financial Sentiment Index and LSTM-based Stock Return Predictability', arXiv preprint arXiv:1906.09024, 2019

30 Othan, D., Kilimci, Z.H., and Uysal, M.: 'Financial Sentiment Analysis for Predicting Direction of Stocks using Bidirectional Encoder Representations from Transformers (BERT) and Deep Learning Models'

31 Jiawei, X., and Murata, T.: 'Stock Market Trend Prediction with Sentiment Analysis based on LSTM Neural Network', in Editor (Ed.)^(Eds.): 'Book Stock Market Trend Prediction with Sentiment Analysis based on LSTM Neural Network' (2019, edn.), pp. 13-15

32 Yadav, A., Jha, C.K., Sharan, A., and Vaish, V.: 'Sentiment Analysis of Financial News Using Unsupervised and Supervised Approach', in Editor (Ed.)^(Eds.): 'Book Sentiment Analysis of Financial News Using Unsupervised and Supervised Approach' (Springer International Publishing, 2019, edn.), pp. 311-319

33 Padial, D.L.: 'Technical Analysis Library using Pandas and Numpy', in Editor (Ed.)^(Eds.): 'Book Technical Analysis Library using Pandas and Numpy' (GitHub, 2019, edn.), pp.

34 Howard, J., and Ruder, S.: 'Universal language model fine-tuning for text classification', arXiv preprint arXiv:1801.06146, 2018

35 Ioffe, S., and Szegedy, C.: 'Batch normalization: Accelerating deep network training by reducing internal covariate shift', arXiv preprint arXiv:1502.03167, 2015

36 pyThaiNLP: 'thai2fit ULMfit Language Modelling', 2019

VITA

NAME	Tanawat Chiewhawan
DATE OF BIRTH	02 Feb 1993
PLACE OF BIRTH	Bangkok, Thailand
INSTITUTIONS ATTENDED	B.Eng. (First Class Honours), Petroleum Engineering, Faculty of Engineering, Chulalongkorn University (2012-2015)
HOME ADDRESS	590/52 Asoke-Dindaeng Road, Dindaeng district, Bangkok 10400
PUBLICATION	"Stock Return Prediction Using Dual-Stage Attention Model with Stock Relation Inference" Chiewhawan T., Vateekul P. (2020). In: Nguyen N., Jearanaitanakij K., Selamat A., Trawiński B., Chittayasothorn S. (eds) Intelligent Information and Database Systems. ACIIDS 2020. Lecture Notes in Computer Science, vol 12033. Springer, Cham
	"Explainable Deep Learning for Thai Stock Market Prediction Using Textual Representation and Technical Indicators" Chiewhawan T., Vateekul P. (2020). In: 2020 The 8th International Conference on Computer and Communications Management (ICCCM 2020)
AWARD RECEIVED	Capital Market Research Innovation Paper 2018 "Attention-based Deep Learning Model on Financial Big Data"
	Capital Market Research Institue Research Grant 2019 "Stock Market Prediction Using Deep Learning Based Model with Event Embedding and Technical Indicators"