

Stock Market Prediction

Submitted By

Dhruv K. Arora

22MCED01



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF TECHNOLOGY, INSTITUTE OF TECHNOLOGY
NIRMA UNIVERSITY
AHMEDABAD-382481**

May 2024

Stock Market Prediction

Major Project - II

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology in Computer Science and Engineering (Data Science)

Submitted By

Dhruv K. Arora

(22MCED01)

Guided By

Dr. Priyank Thakkar



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF TECHNOLOGY, INSTITUTE OF TECHNOLOGY

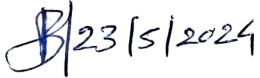
NIRMA UNIVERSITY

AHMEDABAD-382481

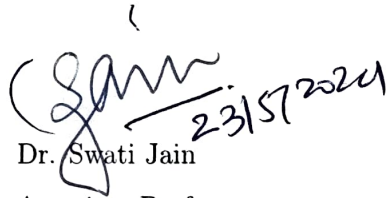
May 2024

Certificate

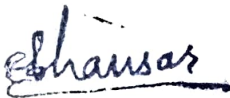
This is to certify that the major project entitled “**Stock Market Prediction**” submitted by **Dhruv K. Arora (Roll No: 22MCED01)**, towards the partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science and Engineering (Data Science) of Nirma University, Ahmedabad, is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project part-II, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.



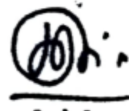
Dr. Priyank Thakkar
Guide & Associate Professor,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.



Dr. Swati Jain
Associate Professor,
Coordinator M.Tech - CSE (Data Science)
Institute of Technology,
Nirma University, Ahmedabad



Dr. Madhuri Bhavsar
Professor and Head,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.



Dr Himanshu Soni
Director,
School of Technology,
Nirma University, Ahmedabad

Statement of Originality

I, **Dhruv K. Arora**, Roll. No. **22MCED01**, give undertaking that the Major Project entitled "**Stock Market Prediction**" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science & Engineering (Data Science)** of Institute of Technology, Nirma University, Ahmedabad, contains no material that has been awarded for any degree or diploma in any university or school in any territory to the best of my knowledge. It is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. It contains no material that is previously published or written, except where reference has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

D.K. Arora

Signature of Student

Date: 21/05/2024

Place: Ahmedabad

B/23/5/2024

Endorsed by

Dr. Priyank Thakkar

(Signature of Guide)

Acknowledgements

It gives me immense pleasure in expressing thanks and profound gratitude to **Dr. Priyank Thakkar**, Associate Professor, Computer Engineering Department, Institute of Technology, Nirma University, Ahmedabad for his valuable guidance and continual encouragement throughout this work. The appreciation and continual support he has imparted has been a great motivation to me in reaching a higher goal. His guidance has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come.

It gives me an immense pleasure to thank **Dr. Madhuri Bhavsar**, Head of Computer Science and Engineering Department, Institute of Technology, Nirma University, Ahmedabad for his kind support and providing basic infrastructure and healthy research environment.

A special thank you is expressed wholeheartedly to **Dr. Himanshu Soni**, Hon'ble Director, School of Technology, Nirma University, Ahmedabad for the unmentionable motivation he has extended throughout course of this work.

I would also thank the Institution, all faculty members of Computer Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

Dhruv K. Arora
22MCED01

Abstract

The research explores the application of Transformer models with time embedding for stock price prediction, focusing on a selection of five diversified stocks from the New York Stock Exchange (NYSE): JP Morgan Chase Co. (JPM) from finance, Johnson Johnson (JNJ) from healthcare, Chevron Corporation (CVX) from energy, International Business Machines Corporation (IBM) from technology, and Procter Gamble Co. (PG) from consumer goods. The study rigorously evaluates the model's performance using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Mean Absolute Percentage Error (MAPE), indicating promising outcomes with low MSE and MAE values, albeit with a relative measure of prediction accuracy (MAPE) around 2.5%. Research aims to improve the accuracy of stock price forecasts by refining predictive models for reducing the Mean Absolute Percentage Error (MAPE) for these NYSE stock price forecasts. This involves exploring advanced feature engineering techniques, alternative machine learning algorithms, ensembling methods, and adjustments to hyperparameters and model architecture to increase prediction accuracy and reduce errors in financial time series prediction. Study highlights the effectiveness of Transformer models with temporal embedding in stock price prediction for these specific stocks, while also acknowledging areas for further improvement to increase prediction accuracy and reduce errors, providing valuable insights into the potential of advanced deep learning techniques in financial market analysis and forecasting..

Abbreviations

ML	Machine Learning
DL	Deep Learning
NLP	Natural Language Processing
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
RMSE	Root Mean Squared Error
MSE	Mean Squared Error
NYSE	New York Stock Exchange
LR	Linear Regression
KELM	Kernel Extreme Learning Machine
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
LSTM	Long Short-term Memory
GRU	Gated Recurrent Unit

Contents

Certificate	iii
Statement of Originality	iv
Acknowledgements	v
Abstract	vi
Abbreviations	vii
List of Figures	ix
1 Introduction	1
1.1 Knowledge Discovery Process	1
1.2 Motivation	4
2 Literature Survey	5
2.1 Introduction	5
2.2 Literature Review	6
3 Proposed Methodology	11
3.1 Workflow of Problem	11
3.2 Proposed Approach	15
3.2.1 Experiments	22
4 Result Analysis	25
4.1 Implementation Screenshots	26
4.2 Analysis	30
5 Conclusion and Future Plan	31
5.1 Conclusion	31
5.2 Future Plan	32
Bibliography	34

List of Figures

1.1	Architecure of Stock Maeket Prediction	3
3.1	Workflow of Problem	12
3.2	Architecture of Transformer [1]	16
3.3	Transformer Encoder Layer	20
4.1	Data Collection of IBM	26
4.2	Data Collection	27
4.3	Data Separation	27
4.4	Actual vs. Predicted price for Training Data	28
4.5	Actual vs. Predicted price for Testing Data	28

Chapter 1

Introduction

This part initiates the research process and provides a frame for the process that will come next. The many procedures necessary to generate new ideas and information in our profession are outlined in Subsection 1.1, “Knowledge Discovery Process.” This section provides a visual picture of the methodical process we use via the use of a reduced architecture. Following this, in section 1.2, “Motivation,” we look at why we are searching, showing the logic behind our desire to understand and be unique. These parts work together to make an interesting opening that sets the stage for the story of stock market forecast that we will be telling.

1.1 Knowledge Discovery Process

The stock market, where investors buy and sell shares in public companies, is a complex system of movement. It is a key element of the global economy, and it has an important influence on investment, growth as well as wealth redistribution. There are many factors that drive stock markets, including the economy, company performance, investor sentiment and political events. As investors will be able to make more accurate decisions about maximising returns and reducing risks, it is important for them to understand and predict the movements of stocks. Continuous analysis and adaptation to its inherent volatility are required for this constantly evolving financial landscape.

The prediction of stock prices is based on historical data and various predictions models, with a view to forecasting the movements in stocks over time. The accuracy of forecasts may provide investors with significant advantages, allowing them to foresee

trends in the market and choose their investment strategy. The process is to analyse past performance, identify patterns and predict behavior going forward. The ability to make more reliable predictions has been enhanced by advances in computational methods and data availability, despite the inherent uncertainty and complexity. In order to gain insight into market dynamics, this field links finance and technology, using sophisticated tools.

Machine learning (ML) and deep learning (DL) techniques are revolutionizing stock price prediction with their ability to analyze large amounts of data, which enables them to reveal concealed patterns. A structured approach to stock price modeling is offered by traditional machine learning methods such as linear regression, support vector machines, and decision trees. By contrast, Deep Learning techniques that include neural networks and recurrent neural networks are capable of modeling complicated, independent relationships in data that capture elaborate dependencies and improve prediction accuracy. These techniques, which provide a robust framework for the development of predictive models, have shown promise in dealing with the unpredictable nature of financial markets.

In spite of the advances made by ML and DL, these techniques may not be used to predict stock prices. The very volatile and noisy nature of financial data is often a problem for traditional models. Long range dependencies and complex temporal patterns associated with stock prices may not be taken into account. Some of these limitations have been addressed by transformers, a recent development in deep learning. Transformers, originally designed for the processing of natural languages, are excellent at capturing long term dependencies and contextual relationships, making them a good fit for sequential data such as stock prices. Their attention mechanism is capable of efficiently handling large amounts of data, which leads to more precise and reliable forecasts.

A significant change from traditional ML and DL methods is the introduction of transformers in stock price forecasts. The transformers will be able to more effectively model complicated interactions in finance data, allowing for enhanced prediction capabilities. transformers are able to handle larger data sets and provide better scale, in contrast to previous techniques. By providing a more flexible and comprehensive approach to analyzing sequential data, it addresses shortcomings in conventional models. This development

highlights the continued evolution in this area, which suggests that transformers may be at the forefront of stock price forecasting methodologies as a basis for investment strategies and market analysis.

- **Architecture**

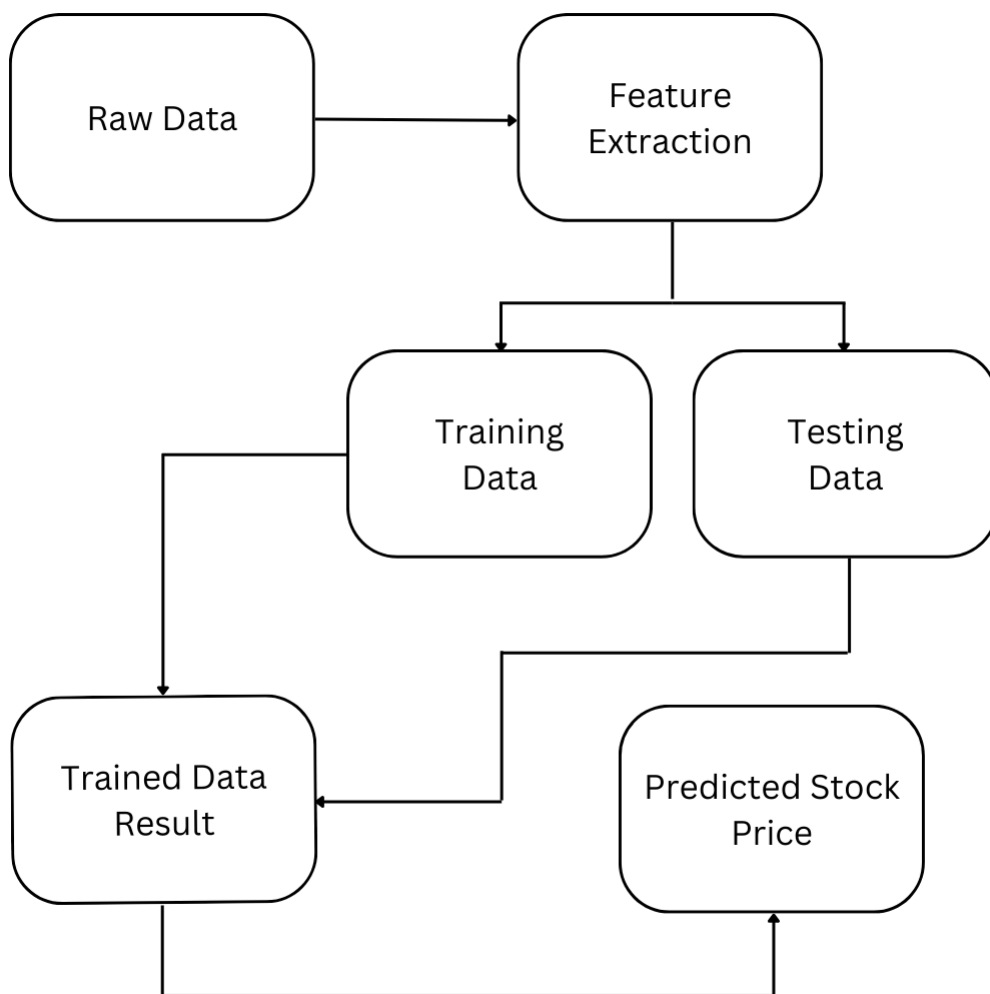


Figure 1.1: Architecture of Stock Market Prediction

Find trustworthy finance sites or APIs and get raw info from them. To make an SMP model, this is the first thing that needs to be done. This lets us learn more about this subject. Tech signs are found, lag traits are added, and mood analysis scores are used in the feature extraction layer. These are things from the raw data that the model can use to find patterns and trends that mean something. Next step is to train and test the

model using data. This lets you test and rate the model. The training set is used to teach the model new things. The factors are changed to make the estimates more accurate. A measure called Mean Absolute Error or Mean Squared Error is used by the review layer to check how well the model matches the root trends.

A graph displays how well the model matches the patterns. Based on the most recent data, this layer will use the model that was learned to guess what prices will be in the future. The chances are good that these things will happen. Last but not least, the release and ongoing improvement layer makes sure that the model can be used in real life. To do this, it tests it in real life, getting comments all the time on how to make it better the whole time.

1.2 Motivation

People who like to guess what will happen in the stock market should look to transformers for ideas. To help kids who were having trouble talking, the first Transformers were made. They teach us a new way to think about business opportunities today. They are better than anyone else at spotting trends over time. This is very helpful for us when we want to understand how stock prices change. If we know more about time, we can help the markets stay calm and guess what will happen when it does.

When things get tough, Transformers are the only ones who can handle them. The money markets can change a lot of things, and changers quickly get used to the huge amount of information they have to handle. Transformers learn how to make their own predictions by looking at a lot of different kinds of data, like past trends, news stories, and economic factors. We can learn how markets work and save money by being able to adapt to new situations.

For added peace and quiet, transformers are great. Transformers get rid of knowledge that isn't useful. It can be hard to understand and organize data about money. We know for sure that the most important things give us our numbers, which helps us make bad guesses. We want to use transformers' ease of use and accuracy to help people guess what will happen in the stock market better and change the way they do it.

Chapter 2

Literature Survey

We're going to discuss Section 2 of the Literature Review here. In Section 2.1, we discuss the introduction to stock price prediction and how ML and DL techniques are useful for predicting the price, as well as the challenges we get from using these techniques. In Section 2.2, we discuss some famous ML and DL techniques that researchers are using for that research, the results they get, and the problems with those techniques.

2.1 Introduction

There has been a big rise in the use of complicated machine learning systems to guess stock prices in the markets over the last few years. As of late, transformer-based models have grown into very powerful tools that can be used for a wide range of natural language processing jobs. Researchers are looking into how transformers can be used in the volatile and always-changing world of financial markets. Since they were first created for sequence-to-sequence processes, they have been useful in other areas. The point of this literature review is to show how the field of using transformer models to guess stock prices is changing as a whole.

It's hard to understand how the stock market works because so many things affect it at once, such as economic data, business success, market mood, and events happening around the world. Because the world is so complicated, standard financial models don't always show all of its parts. There is talk that deep learning could be one way to solve this situation. It was Vaswani et al. (2017) [2] who first used transformers to process natural language. They have shown that they are very good at using sequential data to find long-term patterns and connections. Transformers are used to guess stock prices

because they can find small links in time. This works well with financial time series data that changes over time.

This review will look at the structures, methods, and datasets that have been used in past research to look at transformer-based models for predicting stock prices. By putting together the outcomes of several studies, we hope to get a better sense of the good and bad points of using transformers in this area and figure out how to make future progress. We will also look at how other factors, such as the mood of the news and financial data, can help transformer-based models make more accurate predictions. This study tries to show researchers and practitioners how to use transformers to accurately guess stock prices so that the field can move forward.

2.2 Literature Review

We can use ML (Machine Learning) and DL (Deep Learning) to guess what will happen in the stock market in these ways.

Study on Convolutional Neural Network-Based Stock Price Prediction Method, IEEE 2019-Sayavong Lounnapha et al. This work aims to create a prediction model for stock price, which is based on convolutional neural networks, which are remarkably capable of self-learning. Convolutional neural networks and the Thai stock market are both taught and evaluated using this data set. The outcome demonstrates that the model, which is based on convolutional neural networks, can effectively identify and anticipate the trend in stock market price changes, which is highly suggestive of future stock price movements. The prediction's accuracy is shown to be high, and the banking industry might benefit from its promotion [3].

Improving Profitability with Deep Neural Network-Based Stock Price Prediction, IEEE 2019-Soheila Abrishami, et al., Economic time series prediction is a very difficult undertaking that has drawn the interest of several academics and is crucial for investors. In order to forecast the value of a stock, a deep learning system that utilizes a variety of data for a subset of NASDAQ equities is presented in this work. This model correctly predicts the stock's ending value for multi-step-ahead since it was trained on the shortest

possible amount of data for that specific stock. It uses time series data engineering to combine the new features with the original features and has an auto encoder to eliminate noise. A Stacked LSTM Autoencoder is equipped with these new capabilities to estimate the stock conclusion value multistep-ahead. Additionally, a profit-maximizing strategy uses this estimate to advise on the best times to purchase and sell a specific asset. According to the findings, the proposed framework performs better analytically and practically than the most advanced time series forecasting techniques [4].

An LSTM-Method: A Case Study for Bit-Coin Price Prediction Yahoo Finance Stock Market, Ferdiansyah et al., 2019 IEEE One sort of cryptocurrency that is now being traded on the stock market is bit-coin. There are several dangers associated with stock markets. And one type of cryptocurrency that has been growing recently is bit-coin, which occasionally drops unexpectedly without any apparent impact on the stock market. Because of its volatility, bit-coin in the stock market requires automated systems to anticipate. This work investigates the use of LSTM to generate mode prediction bit-coin stock market predictions. The study attempts to assess the outcomes using RMSE (the Root Mean Square Error) prior to verifying the findings. The MAE and the RMSE will always be greater or equal. The RMSE statistic evaluates a model's ability to compute a continuous value. The methodology used in this study to forecast Bit-coin prices on the stock market Yahoo Finance predicts that the outcome will be more than \$12600 USD during the following several days after the projection [5].

IEEE 2019-Jeevan B et al., "Share Price Prediction using Machine Learning Technique," The stock market has been the buzz of the town lately, with an increasing number of academics and businesspeople expressing interest in it. This study focuses on the method of forecasting stock prices on the National Stock Exchange using RNN (Recurrent Neural Network) and LSTM (Long Short Term Memory) utilizing a variety of factors, including the current market price and anonymous events. This study also mentions the usage of a recommendation system and models built on RNN and LSTM techniques in the company selection process [6].

Year	Objective	Approach	Dataset	Accuracy measures	Conclusion
2023 [7]	Predicting stock market prices using technical indicators.	Kernel Extreme Learning Machine (KELM)	YES bank and HDFC bank	MAPE, MAE, RMSE	Model efficiency observed with two different stock market prices.
2023 [8]	Predicting stock trends using LSTM and GRU models.	LSTM and GRU	Historical data set	RMSE, MAPE	LSTM models are more powerful than GRU models due to complex gating mechanisms and long-term dependencies.
2023 [9]	Creation of a web-based application for monitoring and analyzing stock data.	LSTM, RNNs, Prophet	Historical data of Microsoft Stock	RMSE, MAE, R^2 , MSE	Prophet outperforms other models and can be considered a benchmark for time series forecasting.
2023 [10]	To explore the effectiveness of using CNNs for stock market prediction, particularly when these networks are trained on data from multiple time frames	Multi-CNN model	EURUSD data	MSE and FLF	The Multi-CNN model demonstrates superior performance compared to LSTM, GRU, and 1D CNN architectures when applied to the EURUSD dataset. It effectively reduces both Mean Square Error and Forex Loss Function, indicating its ability to make more accurate predictions.

Table 2.1: Literature Review Table

Year	Objective	Approach	Dataset	Accuracy measures	Conclusion
2023 [11]	Compare SVR and RNN for stock price prediction.	RNN and SVM	Astra Agro Lestari	RMSE	RNN has higher error rate; SVM shows higher accuracy.
2023 [12]	A method for forecasting stock prices, referred to as "DeepNet." The primary focus seems to be on using deep learning techniques for this prediction task.	LSTM	Traditional data of many stocks from Yahoo Finance website	MAPE, MAE, MSE	The LSTM can be used in conjunction with social media sentiment analysis to more effectively train weights and advance machine learning models. It has led to the conclusion that machine learning techniques can be used to anticipate the stock market more effectively and accurately.
2023 [13]	Predicting stock prices using various machine learning techniques.	LR, LSTM, CNN	Traditional data(Google, Goldman Sachs, Microsoft, Tesla, Reliance)	MAE, MSE, RMSE	Three models, namely Linear Regression, LSTM, CNN, have been trained on the specified dataset. After determining the best parameters and validating their results using hold out validation, their performances have been compared using MSE, MAE and RMSE.LSTM provides more accurate forecasts for most of the sequence lengths than other models.

Table 2.2: Literature Review Table

Naadun Sirimevan et al., "Stock Market Prediction Using Machine Learning Techniques," IEEE 2020 The prices on the stock market are quite important in the modern economy. Researchers have shown that social media sites like Twitter and online news have an impact on people's ability to make decisions. The behavioral response towards web news is considered in this study to close the gap and improve the prediction's accuracy. For the next day, the next week, and the next two weeks, exact forecasts were created [14].

According to research published in 2023, a number of machine learning models, such as Kernel Extreme Learning Machine (KELM) and Long Short-Term Memory (LSTM), do exceptionally well in predicting stock market values. Because of its capacity to handle complex gating mechanisms and long-term dependencies, LSTM models routinely perform better than GRU and RNNs. While Multi-CNN models beat LSTM, GRU, and 1D CNN in lowering prediction errors for forex data, KELM is useful for some institutions. Additionally, Support Vector Regression (SVR) outperforms RNNs in accuracy.

Key discoveries and developments in machine learning-based stock market prediction are highlighted in the literature study. In terms of accuracy metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE), Long Short-Term Memory (LSTM) models routinely outperform other models like Gated Recurrent Units (GRU) and conventional regression techniques like Linear Regression.

Other methods such as Multi-CNN models and Kernel Extreme Learning Machine (KELM) are becoming more and more popular and have potential in certain situations. Bank stock prices can be accurately predicted by KELM, while complicated datasets spanning many time periods can be handled well by Multi-CNN models. Analyzing several models side by side reveals their advantages and disadvantages. For example, SVR performs better on some datasets than RNNs, which are frequently employed for sequence prediction tasks.

Chapter 3

Proposed Methodology

In this part, we'll look back at our trip so far. This part Section 3.1 talks about how we can solve the problem. we look at in Section 5. After that, in Section 3.2, we talk about which approach i used for Stock price prediction. This part talks about where we're going in Stock market prediction and what we've learned so far.

3.1 Workflow of Problem

In Transformers, you can guess what will happen next on the stock market. To do it, follow these steps. As part of it, you have to get data, clean data, train models, and guess what numbers will be in the future [15].

Before making predictions about the stock market, it's important to learn a lot of useful things. To do this, you need to look at more than just essential signs, business news, and social media posts. A business's growth depends on simple things like how much money it makes. People's feelings about the market and the world around it can be seen in news about money and statistics from social networks. We can tell how the market has changed over time by looking at how much stocks cost in the past.

It needs to be checked out first to make sure it's good enough to use for studying. The data needs to be cleaned up because some numbers are missing or don't make sense. Next, pick and choose traits to see which ones help or hurt the prediction. This is a big step toward making math easier and models work better.

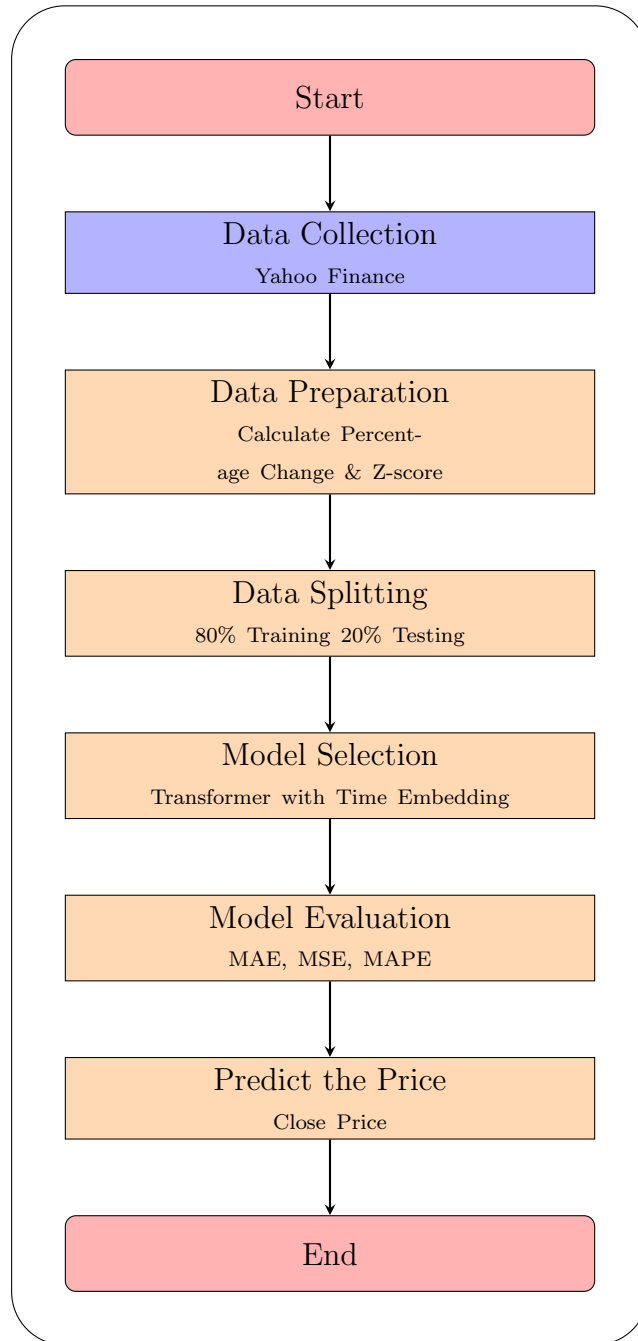


Figure 3.1: Workflow of Problem

We can check how well the model works by dividing the information into training and testing sets. If the hyperparameters are wrong, you can fix them with the validation set [16]. It tells the model what to do when it gets new ideas. With the test set, you can see how well the model can learn new things. After this step, you can be sure that the model is good enough to guess new market data that hasn't been used yet.

It's the first time it's broken up. For machine learning or deep learning, the training example tells it what to do. The model then changes its settings to get better at guessing as it learns from the ties and trends in the data. This stage is all about making small changes based on the training set data. The model needs to work as well as it can [17].

We use this set to test various transformer-based models after training them based on how well they do. The chosen model is right and broad, which is the best of both worlds. It's possible that the hyperparameters need to be changed right now to make the model more stable and better at predicting how the stock market will move.

Once it has been picked out and taught, the model can be used to guess what will happen next in the stock market. To do this, you need to teach the learned model something new, like tomorrow's stock prices, new basic signs, or news about recent financial events [18]. The model then tells us what the future market will be like. We can guess what might happen now. A lot of times, you should check on the plan and make sure it stays right as the market changes.

- **Data Collection:**

I have collected historical stock price data for five diversified stocks from the New York Stock Exchange (NYSE), covering the period from the inception of each stock until December 31, 2023, for the purpose of this study. The selected stocks represent different sectors of the economy, so that a well rounded and diversified dataset can be used for analysis. Johnson Johnson JNJ, Procter Gamble PGE, Chevron Corporation CVX, International Business Machines Corp IBM and JPMorgan Chase Co. JPM are the stocks included in this data set.

- **Architectural Development:**

In order to capture and understand the complex temporal dependencies of financial time series data, our advanced stock price prediction architecture aims at producing accurate forecasts. In order to tackle the unique challenges of stock price forecasting, this development is focused on incorporating cutting edge deep learning techniques, such as Transformer models that have been embedded with time.

- **Evaluation and Enhancement:**

The rigorous testing and comparison of our stock price prediction architecture to a variety of financial data sets is an essential element for the evaluation of its performance. In order to ensure the robustness and accuracy of our Transformer Based Model in various market conditions, this Comprehensive Assessment is intended to measure its effectiveness with time embedded models.

Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad [19] \quad (3.1)$$

Mean Squared Error (MSE):

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad [19] \quad (3.2)$$

Mean Absolute Percentage Error (MAPE):

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| * 100 \quad [19] \quad (3.3)$$

- y_i represents the actual or observed value of the dependent variable for the i^{th} data point.
- \hat{y}_i represents the predicted or estimated value of the dependent variable for the i^{th} data point.
- n denotes the total number of data points or observations in the dataset.

3.2 Proposed Approach

- **Data Collection:**

This research simplifies the 1 terabyte stock dataset by utilizing the International Business Machines Corporation (IBM), Johnson Johnson (JNJ), Procter Gamble Co. (PG), Chevron Corporation (CVX), and JPMorgan Chase Co. (JPM) stock price histories as an example. However, the technique in this post can be simply applied to a much larger dataset. The IBM dataset spans 15,000+ trading days, beginning on January 2, 1962, and ending on December 31, 2023. Each training day includes the IBM stock’s open, high, low, and close prices, as well as the trading volume (OHLCV).

- **Data preparation:**

The price and volume features are translated to daily stock returns and volume changes, then standardized using Z-score, and the time series is divided into training and test sets. Converting stock prices and volumes to daily change rates improves our dataset’s stationarity. Our dataset provides more valid learning for future forecasts. Here’s an overview of the modified data.

$$\hat{X}_t = \frac{X_t - \mu}{\sigma} \quad [20] \quad (3.4)$$

where:

- \hat{X}_t is the normalized price at time.
- μ is the sample mean of the training set.
- σ is the sample standard deviation of the training set.

Finally, training and test sets are divided into 128-day sequences. Each sequence day includes four price features (open, high, low, and close) and a volume feature, totaling five features each day. In a single training step, our Transformer model will

receive 32 sequences (batch size = 32) with 128 days (seq len = 128) and 5 features per day as input.

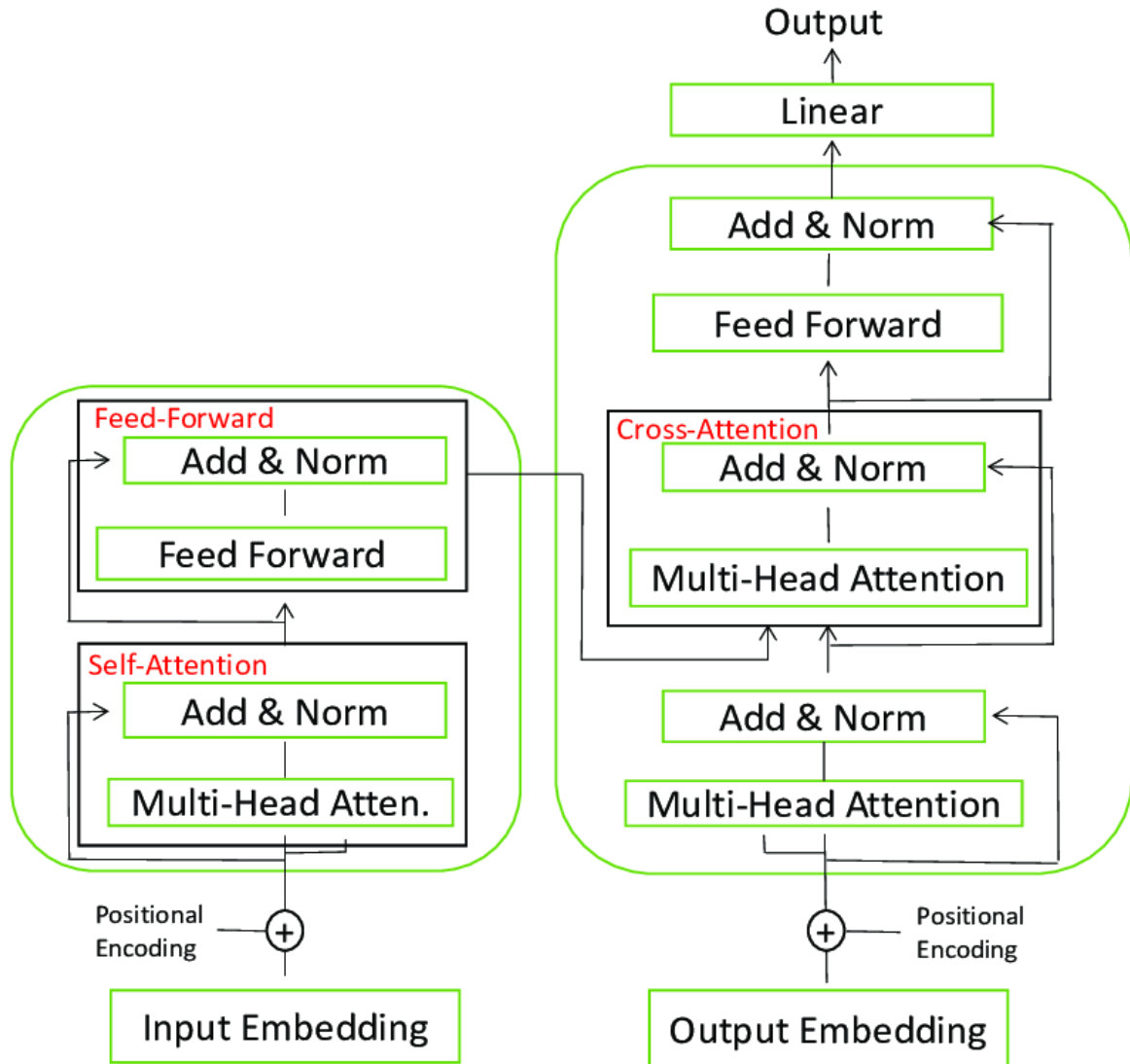


Figure 3.2: Architecture of Transformer [1]

- **Time Embeddings:**

The first step in implementing Transformer is to encode the hidden concept of time in stock prices into the model. When processing time-series data, time is an important consideration. When processing time-series or sequential data using a transformer, sequences are transmitted all at once, making it challenging to discern temporal or sequential dependencies. Transformers used with natural language data often use positional encoding to determine word order for the model. Positional en-

coding represents a word’s value and position in a phrase, allowing the transformer to understand sentence structure and word dependencies.

Similarly, a transformer requires time to process our stock prices. Without time embeddings, our transformer would not understand the temporal order of our stock prices. Stock prices from 2020 can have the same impact on future price predictions as those from 1990. And, of course, this would be absurd.

- **Time2Vector:**

To overcome a transformer’s temporal indifferences, we will use the Time2Vec technique presented in the study [21]. The paper proposes Time2Vec, a model-agnostic vector representation of time. Vector representations, similar to embedding layers, can enhance neural network performance.

To overcome a transformer’s temporal indifferences, we will use the Time2Vec technique presented in the study [21]. The paper proposes Time2Vec, a model-agnostic vector representation of time. Vector representations, similar to embedding layers, can enhance neural network performance. When it comes to the paper’s core principles, there are two major points to examine.

The authors identified that a meaningful depiction of time requires both periodic and non-periodic patterns. Weather patterns, such as seasonal changes, are an example of periodicity. A time representation should be invariant to time rescaling, meaning it does not change with different time increments (days, hours, or seconds) or extended time horizons. The mathematical definition below combines periodic and non-periodic patterns with invariance to temporal rescaling.

$$t2v(\tau)[i] = \begin{cases} \omega_i\tau + \phi_i & \text{if } i = 0 \\ F(\omega_i\tau + \phi_i) & \text{if } 1 \leq i \leq k \end{cases} \quad [22]$$

The expression $\omega_i\tau + \phi_i$ represents the non-periodic/linear component, and $F(\omega_i\tau + \phi_i)$ represents the periodic feature of the time vector.

Let's examine the general dynamics of how various non-linear functions of the time vector (Time2vec) affect an LSTM network's accuracy. It is evident that the ReLU function has the lowest performance; in comparison, All other non-linear functions are subpar compared to the sine function. The ReLU function is not time rescaling invariant, which explains why the results are so unsatisfactory. The better the performance, the stronger the invariant of a function against time rescaling.

- **Time2Vector Keras implementation:**

Now that we have covered the theoretical operation of the periodic and non-periodic components of our time vector, let's put them into practice using code. We will define the time vector as a Keras layer so that it may be readily integrated into any type of neural network design. The two sub functions in our custom Time2Vector Layer are `def build():` and `def call():`. In the `def construct()` function, we start two matrices for ω and two for ϕ since we require an ω and ϕ matrix for both the periodical (sin) and non-periodical (linear) features.

- **Transformer:**

After establishing the concept of time and implementing a time vector, the following step is to introduce the Transformer. A Transformer is a neural network that uses self-attention to increase prediction accuracy by focusing on relevant time-series data. The self-attention mechanism includes both a single-head and multi-head attention layer. The self-attention process connects all time-series stages simultaneously, creating long-term dependence understandings. The Transformer design parallelized all processes, accelerating learning.

- **Combining data and time features: Feeding the Transformer**

After implementing Time Embeddings, we will use the time vector along with IBM's pricing and volume attributes as input for our Transformer. The Time2Vector layer uses IBM pricing and volume characteristics to calculate both non-periodic and periodic time features. In the model stage, the time features are concatenated with price and volume characteristics to create a matrix with the shape (32, 128, 7).

- **Single-Head Attention:**

The first single-head attention layer receives the IBM time-series and newly calculated time characteristics as initial input. Single-head attention layer accepts three inputs in total: query, key, and value. We consider each Query, Key, and Value input to represent IBM's pricing, volume, and time attributes. Query, Key, and Value inputs undergo distinct linear transformations across dense layers. The deep layers with 96 output cells were a personal architectural decision.

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad [2] \quad (3.5)$$

Following the initial linear transformation, we will compute the attention score and weights. When predicting a future stock price, the attention weights control how much emphasis is given on individual timeseries stages. To calculate attention weights, the dot-product of linearly processed Query and Key inputs is used. The transformed Key input is transposed to allow for dot-product multiplication.

To prevent explosive gradients, divide the dot-product by the previous dense layers' dimension size(96). The split dot-product is applied to the softmax function, resulting in weights that total up. The final phase involves multiplying the softmax matrix, which sets the focus of each time step, with the modified v matrix to complete the single-head attention mechanism.

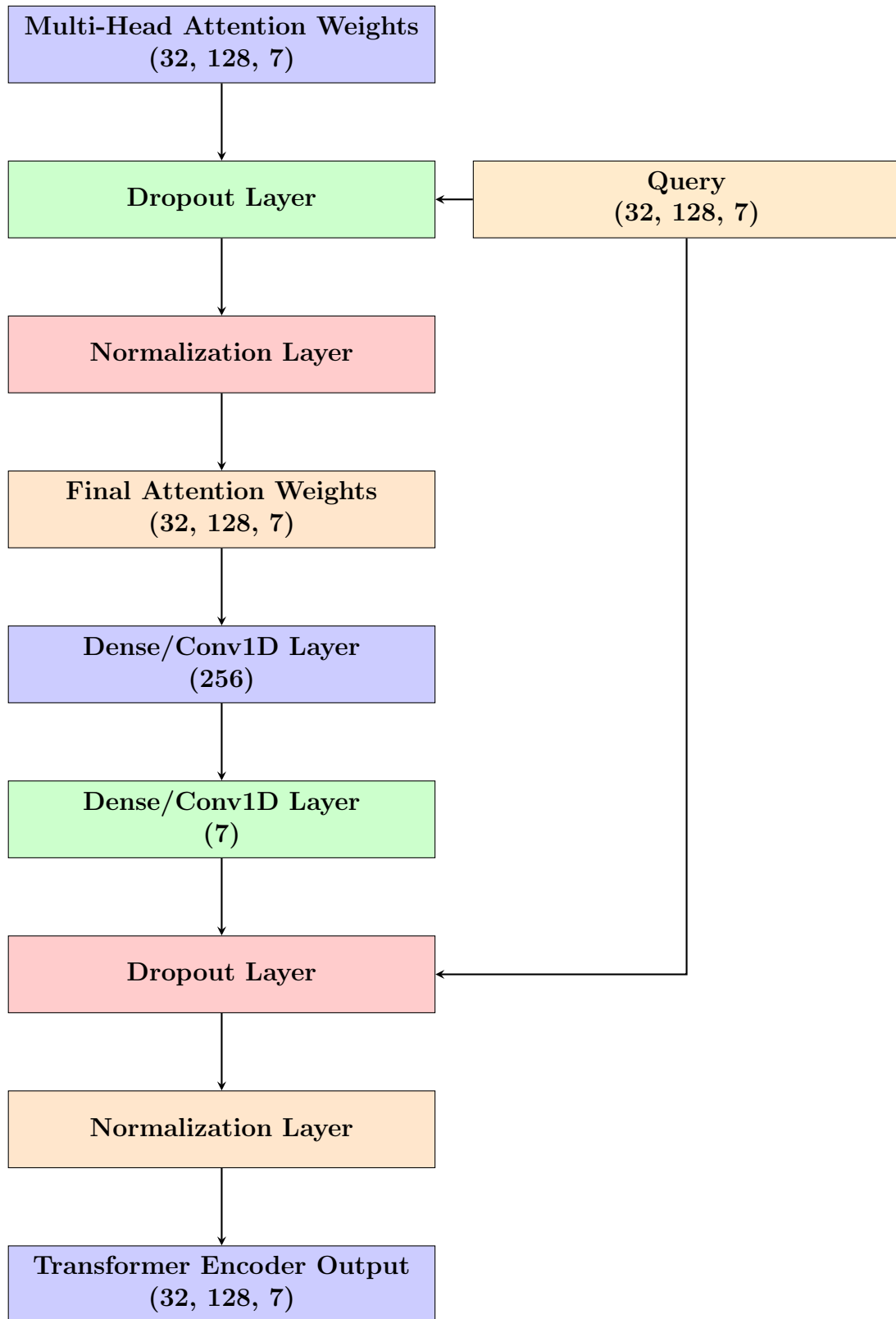


Figure 3.3: Transformer Encoder Layer

- **Multi-Head Attention:**

The paper "Attention Is All You Need" aims to develop self-attention mechanisms [2]. advocated the use of multi-head attention. A multi-head attention layer combines the attention weights of multiple single-head layers before applying a non-linear transformation with a dense layer. The picture below depicts the concatenation of three single-head layers.

The output of n single-head layers enables the encoding of many independent transformations into the model. The model can focus on numerous time series steps simultaneously. Adding more attention heads improves a model's capacity to identify long-distance dependencies [23].

- **Transformer Encoder Layer:**

The single- and multi-head attention processes (self-attention) are combined into a transformer encoder layer. Each encoder layer contains two sublayers: selfattention and feedforward. The feedforward sublayer is composed of two thick layers with ReLU activation in between.

To replace dense layers, use 1-dimensional convolutional layers with a kernel size and stride of 1. The arithmetic for a dense layer and a convolutional layer with the stated design is same.

After each sublayer, a dropout layer is added to both sublayer outputs to create a residual connection, which includes the initial query input. After each sublayer, a normalization layer is added after the residual link to stabilize and expedite the training process.

To conclude, we initialize the time embedding layer and three Transformer encoder layers. After initialization, we add a regression head to the final transformer layer and start training.

3.2.1 Experiments

To show how well Transformer performs in stock market prediction, I do a number of back-testing trials for the major global stock market indexes.

- Data Collection:

This research simplifies the 1 terabyte stock dataset by utilizing the International Business Machines Corporation (IBM), Johnson Johnson (JNJ), Procter Gamble Co. (PG), Chevron Corporation (CVX), and JPMorgan Chase Co. (JPM) stock price histories as an example. However, the technique in this post can be simply applied to a much larger dataset. The IBM dataset spans 15,000+ trading days, beginning on January 2, 1962, and ending on December 31, 2023. Each training day includes the IBM stock's open, high, low, and close prices, as well as the trading volume (OHLCV).

- Data processing:

A daily closing price one-dimensional time series serves as the observed data for each index. The entire dataset was first divided into training and testing sets. The first 80% of the data, or the training set, is employed to train the parameters of the model. Model performance is assessed using the final 20% of data as the testing set. The original data, including the training and testing sets, are normalized in order to lower dataset volatility and provide a robust model.

- Hyper-parameters setting:

I do a great deal of tests on the training set in order to identify the ideal hyper-parameters beforehand. For the mini-batch training, a batch size of 32 is used. The loss function that we use to compare the actual and forecasted values is the mean squared error, mean absolute error and mean absolute percentage error. For training models, the Adam optimizer with a learning rate of 0.0001 is employed. To ensure the training process's convergence, we fixed the number of epochs at 35. They show notable declining patterns and finally converge within 35 epochs. We use the dropout approach, with a 0.1 dropout rate set for each sub-layer, to prevent the overfitting issue.

I employ a 128-character sequence length and a batch size of 32. With a feed-forward network dimension (ff_dim) of 256 and 12 attention heads (n_heads), the dimensions for the keys (d_k) and values (d_v) are both set to 256.

- Transformer:

Single Attention

For a given input, it computes the query, key, and value matrices. The attention mechanism makes use of these matrices to concentrate on relevant sections of the input sequence by computing the dot product of the query and key matrices and utilizing a softmax function to get attention weights. These weights are then used to scale the value matrix so that important data is highlighted.

Multi Attention

Multiple SingleAttention heads are initialized. To combine different attention perspectives, the input is processed individually by each head, and their outputs are concatenated and linearly modified. The model can capture many features of temporal relationships in the data thanks to its multi-head method.

Transformer Encoder

Position-wise feed-forward neural networks are placed after a multi-head attention mechanism in each TransformerEncoder layer. Layer normalization and residual connections are used to improve and stabilize learning. The encoder builds a thorough grasp of the temporal patterns in the data by processing the input through many feed-forward and attention layers.

Time2Vector Layer

The concatenation and addition of both linear and periodic characteristics to the input sequence gives the model explicit time-related information that facilitates the better identification and exploitation of temporal patterns.

- Evaluation criteria:

Prediction accuracy and net value analysis are two methods used to assess a model's performance. We compute the prediction errors and compare the expected values with the real data in the testing set to determine the prediction accuracy. To assess the performance, three typical markers for prediction errors are used: Mean Absolute Error (MAE), Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE).

Chapter 4

Result Analysis

We're going to discuss Section 4 of the Result and Analysis here. In Section 4.1, I share the results of my experiments, and in Section 4.2, I explain my analysis of these results.

I used historical stock price information for five different stocks from the New York Stock Exchange (NYSE) for my study. From the time the stock was introduced until December 31, 2023, the information was accessible. The chosen equities provide a diverse and extensive dataset for analysis because they span several economic sectors. Procter Gamble Co. (PG), Johnson Johnson (JNJ), Chevron Corporation (CVX), International Business Machines Corporation (IBM), and JP Morgan Chase Co. (JPM) are the equities that are part of the dataset.

We analyze each of these stocks performance using Open-High-Low-Close-Volume (OHLCV) data and the output from our Transformer model with temporal embedding in the "Results Analysis" section. The primary objective of the study is to ascertain the accuracy of the stock price estimates; key performance indicators for the model are Mean Squared Error (MSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). Through an analysis of the projected returns for every company, we offer incisive perspectives on the efficacy of the Transformer model in various businesses and pinpoint certain domains where more enhancements in prediction precision might be achievable.

4.1 Implementation Screenshots

	A	B	C	D	E	F	G
1	Date	Open	High	Low	Close	Adj Close	Volume
2	02-01-1962	7.374124	7.374124	7.291268	7.291268	1.541712	407940
3	03-01-1962	7.291268	7.355003	7.291268	7.355003	1.555188	305955
4	04-01-1962	7.355003	7.355003	7.278521	7.281708	1.539689	274575
5	05-01-1962	7.272148	7.272148	7.125558	7.138305	1.509366	384405
6	08-01-1962	7.131931	7.131931	6.9471	7.004461	1.481066	572685
7	09-01-1962	7.036329	7.176546	7.036329	7.087317	1.498587	517770
8	10-01-1962	7.100064	7.131931	7.100064	7.100064	1.501282	313800
9	11-01-1962	7.119184	7.176546	7.119184	7.176546	1.517453	337335
10	12-01-1962	7.189293	7.24028	7.189293	7.189293	1.520148	462855
11	15-01-1962	7.214786	7.237094	7.214786	7.22116	1.526885	266730
12	16-01-1962	7.214786	7.214786	7.144678	7.144678	1.510715	266730
13	17-01-1962	7.112811	7.112811	7.010835	7.029955	1.486457	439320
14	18-01-1962	7.049076	7.119184	7.049076	7.049076	1.490499	392250
15	19-01-1962	7.049076	7.068196	6.972594	7.055449	1.491848	423630
16	22-01-1962	7.055449	7.125558	7.036329	7.042702	1.489152	290265
17	23-01-1962	7.039516	7.049076	6.959847	6.972594	1.474328	392250
18	24-01-1962	6.972594	7.010835	6.93754	7.010835	1.482413	431475
19	25-01-1962	7.010835	7.068196	6.921606	6.934353	1.466242	337335
20	26-01-1962	6.934353	6.959847	6.896112	6.899299	1.45883	258885
21	29-01-1962	6.899299	6.9471	6.781389	6.781389	1.433898	611910
22	30-01-1962	6.781389	6.81963	6.695347	6.695347	1.415705	776655
23	31-01-1962	6.762269	6.908859	6.762269	6.908859	1.460853	800190
24	01-02-1962	6.978967	7.087317	6.978967	7.068196	1.494543	674670
25	02-02-1962	7.068196	7.112811	7.036329	7.112811	1.503977	533460
26	05-02-1962	7.112811	7.112811	6.985341	7.023582	1.485108	329490
27	06-02-1962	7.023582	7.036329	6.998088	7.029955	1.48666	274575

Figure 4.1: Data Collection of IBM

The total number of trading days, or more accurately, more than 15 000 trading days, is shown in Figure 4.2. My data splitting strategy, which uses 80% of the training data and 20% of the testing data, can be seen in Figure 4.3.

The graph of the actual versus projected cose price for the training data set is shown in fig. 4.4, and the graph of the test data set is presented in fig. 4.5.

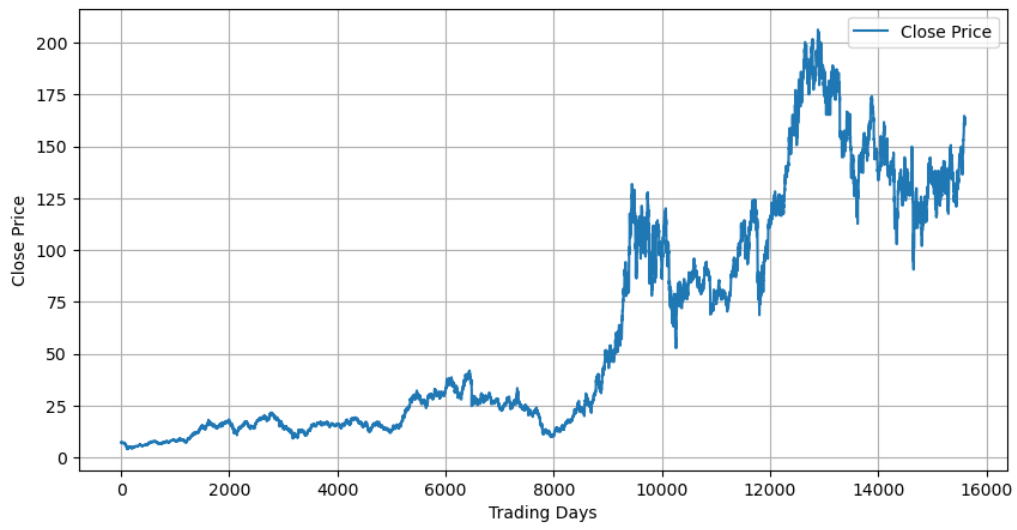


Figure 4.2: Data Collection

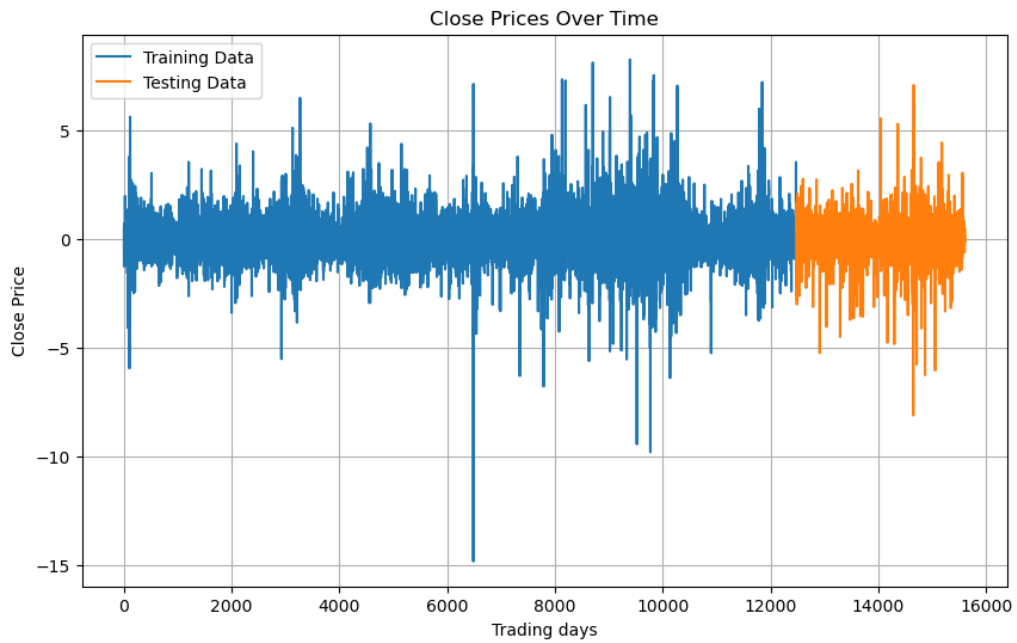


Figure 4.3: Data Separation

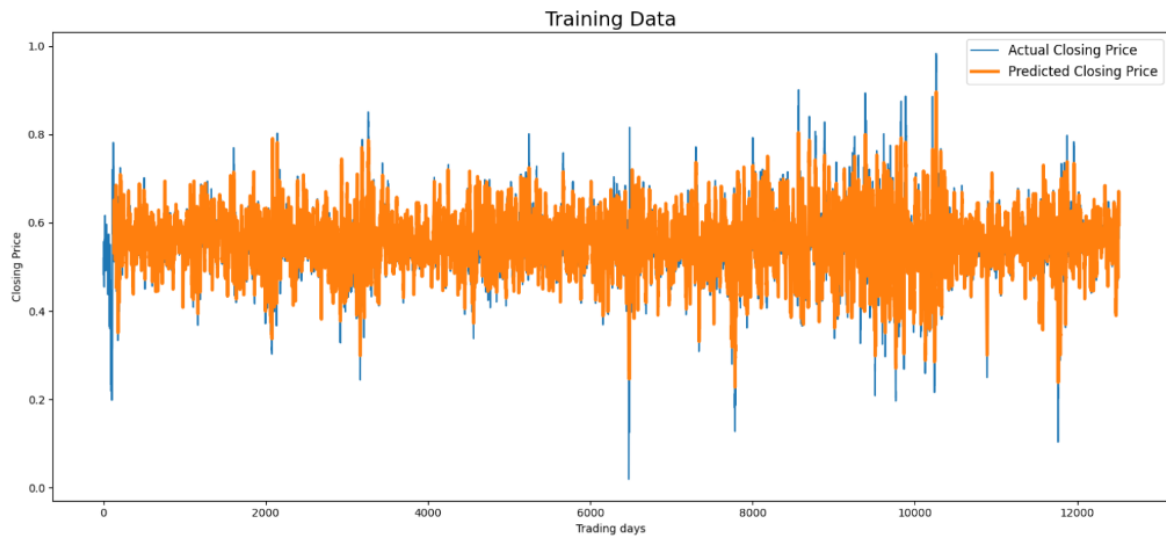


Figure 4.4: Actual vs. Predicted price for Training Data



Figure 4.5: Actual vs. Predicted price for Testing Data

Symbol	Sector	MAE	MSE	MAPE
IBM	Information Technology	0.0246	0.0014	2.5732
JNJ	Healthcare	0.0283	0.002	2.8037
PG	Consumer	0.0301	0.0029	2.6948
CVX	Energy 2	0.0295	0.0025	2.8751
JPM	Financial	0.0314	0.0032	2.9651

Table 4.1: Experiment Result

The table presents a thorough analysis of the performance indicators for five equities that are listed on the New York Stock Exchange (NYSE) and are diversified across many industries. The financial industry’s JP Morgan Chase Co. (JPM), the healthcare industry’s Johnson Johnson (JNJ), the energy industry’s Chevron Corporation (CVX), the information technology industry’s International Business Machines Corporation (IBM), and the consumer goods industry’s Procter Gamble Co. (PG) are a few of these stocks. The assessment measures that are employed include Mean Absolute Error (MAPE), Mean Squared Error (MSE), and Mean Absolute Percentage Error (MAPE). These metrics offer useful information about how accurate our stock price forecasts are for a variety of industries.

With an MAE of 0.0246 and an MSE of 0.0014, IBM beat the other stocks in the analysis, demonstrating its strong performance in the information technology business. Its very low MAPE of 2.5732%, which demonstrated accurate forecasts of IBM stock prices, gave more support for this. JP Morgan Chase Co. (JPM) had the lowest accuracy metrics, matching the other corporations with a MAPE of 2.9651%, MSE of 0.0032, and MAE of 0.0314. These data suggest that, while our methodology is effective in a number of industries, forecasting stock prices in the financial industry is more challenging. This issue might stem from the industry’s complex dynamics and responsiveness to external market factors.

4.2 Analysis

There are 35 epochs in all during the training procedure. Following training, it is evident that our transform model is only forecasting a flat line with a center point that is between each day's fluctuations in the stock price. Even a transformer model is only able to forecast the linear trend of a stock's development when it just uses the IBM stock history. determining that there is just sufficient explanatory value for a linear trend forecast in the historical price and volume data of a stock. Nevertheless, the outcomes are significantly different when the dataset is upscaled to thousands of stock tickers (1 Terabyte dataset).

From past stock prices and volumes, non-linear stock projections cannot be extracted by even the most sophisticated model designs. However, the model is able to produce noticeably improved predictions (green line) when a simple moving average smoothing technique is applied to the data (window size=10). Rather of forecasting the IBM stock's linear trend, the model can also forecast the stock's ups and downs. We may still conclude that there are outlier problems with the model, though, because close examination reveals that it still has a significant prediction delta on days with excessive daily change rates. The smoothing of the moving average impact results in an increase in performance; this may be accomplished even in the absence of a moving average.

Chapter 5

Conclusion and Future Plan

In this part, we'll look back at our trip so far. This part (part 5.1) talks about what we learned from all of our study. It's the first thing we look at in Section 5. After that, in Section 5.2, we talk about what comes next, like our plans for the future. This part talks about where we're going in SMP and what we've learned so far.

5.1 Conclusion

For the purpose of this study, we used a Transformer model with temporal embedding to predict stock prices for five different NYSE index stocks. Our objective was to use the broad temporal trends observed in the market data by using historical data that dates back to the inception of each stock and concludes on December 31, 2023. The model's capacity to capture intricate correlations across time and provide accurate forecasts proved the value of the Transformer architecture in financial time series prediction.

Mean Absolute Error (MAPE), Mean Squared Error (MSE), and Mean Absolute Percentage Error (MAPE) were the metrics we used for assessment. Strong performance was demonstrated by the low MSE and MAE values in the findings, which revealed that real stock prices and our model's forecasts were quite close to one another. On the other hand, the relative measure of prediction accuracy, or MAPE, was around 3%. This little disparity draws attention to areas that might want improvement, especially in terms of fine-tuning the model's sensitivity to more subtle fluctuations in stock prices—a must for precise percentage-based evaluations.

We have observed differing levels of accuracy about the various evaluation metrics when examining the performance of selected diversified stocks on the NYSE. In IBM, JNJ, PG, CVX and JPM the mean absolute error values show a relatively low level of errors highlighting the effectiveness of prediction models to identify trends. Similarly, these findings are corroborated by the mean square error values of the MSE, which are generally small deviations from the actual values. The mean absolute percentage error MAPE indicates slight differences, which suggests that there is room for improvement in the measure of comparative errors. Despite these differences, the predictability approach in various market segments is shown to be robust by a consistent performance across industries with IBM from IT, JNJ from Healthcare, PG from Consumer, CVX from Energy and JPM from Finance.

Finally, research have demonstrated that utilizing Transformer models with temporal embedding to anticipate stock prices for NYSE indexes is a successful strategy. Even though the MSE and MAE values were determined to be acceptable, the MAPE shows that more improvements in prediction accuracy are necessary, particularly in relation to relative error measures.

5.2 Future Plan

In order to improve the precision of our stock price forecasts, we want to reduce the Mean Absolute Percentage Error (MAPE) in the future. We plan to examine several approaches in order to achieve that goal. Initially, we will use sophisticated feature engineering approaches to extract more interesting and pertinent qualities from the historical stock data. This may entail taking into consideration other market indicators, financial news sentiment analysis, and macroeconomic variables that affect stock prices. In addition, we will test several time embedding strategies to see which ones better capture the seasonal patterns and temporal dynamism present in stock market data.

I adjust the hyperparameters and architecture of our Transformer model. Experimenting with different layers, attention mechanisms, and optimization algorithms is necessary to make the model more responsive to even the tiniest changes in price. To lessen pre-

diction bias and variation, ensemble approaches which combine predictions from many models will also be taken into consideration. Our goal is to drastically reduce the MAPE and offer more precise and trustworthy stock price projections by concentrating on these strategies. This focused work will improve our model's performance and add important knowledge to the field of financial time series prediction.

Future work will focus on refining predictive models to specifically target improvements in reducing the Mean Absolute Percentage Error (MAPE) for NYSE stock price forecasts. To better capture the underlying patterns and dynamics of stock market movements, this will involve exploring advanced feature engineering techniques tailored to this end. In addition, opportunities to mitigate errors related to MAPE could be offered by integrating alternative machine learning algorithms and ensembling methods.

Bibliography

- [1] C. Li and G. Qian, “Stock price prediction using a frequency decomposition based gru transformer neural network,” *Applied Sciences*, vol. 13, no. 1, p. 222, 2022.
- [2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [3] N. Sirimevan, I. U. H. Mamalgaha, C. Jayasekara, Y. S. Mayuran, and C. Jayawardena, “Stock market prediction using machine learning techniques,” in *2019 International Conference on Advancements in Computing (ICAC)*, pp. 192–197, 2019.
- [4] S. Singh, T. K. Madan, J. Kumar, and A. K. Singh, “Stock market forecasting using machine learning: Today and tomorrow,” in *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, vol. 1, pp. 738–745, 2019.
- [5] F. Ferdiansyah, S. H. Othman, R. Zahilah Raja Md Radzi, D. Stiawan, Y. Sazaki, and U. Ependi, “A lstm-method for bitcoin price prediction: A case study yahoo finance stock market,” in *2019 International Conference on Electrical Engineering and Computer Science (ICECOS)*, pp. 206–210, 2019.
- [6] M. Misra, A. P. Yadav, and H. Kaur, “Stock market prediction using machine learning algorithms: A classification study,” in *2018 International Conference on Recent Innovations in Electrical, Electronics Communication Engineering (ICRIEECE)*, pp. 2475–2478, 2018.
- [7] N. Parida, B. K. Balabantaray, R. Nayak, and J. K. Rout, “A deep learning based approach to stock market price prediction using technical indicators,” in *2023*

- International Conference on Advances in Intelligent Computing and Applications (AICAPS)*, pp. 1–7, IEEE, 2023.
- [8] H. Momaya, V. Patel, and V. Momaya, “Forecasting of stock trend and price using machine intelligence lstm and gru models,” in *2023 2nd International Conference on Vision Towards Emerging Trends in Communication and Networking Technologies (ViTECoN)*, pp. 1–6, IEEE, 2023.
- [9] P. Ganigi, B. Yamini, *et al.*, “Monitoring and analysis of stock data with deployable web-based application,” in *2023 2nd International Conference on Vision Towards Emerging Trends in Communication and Networking Technologies (ViTECoN)*, pp. 1–5, IEEE, 2023.
- [10] N. Nemati, H. Farahani, and S. R. Kheradpisheh, “Stock market prediction by combining cnns trained on multiple time frames,” in *2023 5th International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pp. 1–8, IEEE, 2023.
- [11] I. Yunianto, T. Haryanto, M. M. Mutoffar, Y. Fadhillah, N. E. Putria, *et al.*, “Comparison of stock price predictions using support vector regression and recurrent neural network methods,” in *2023 International Conference on Computer Science, Information Technology and Engineering (ICCoSITE)*, pp. 927–932, IEEE, 2023.
- [12] S. Roy and S. Tanveer, “Stock price forecasting using deepnet,” in *2023 International Conference on Computational Intelligence, Communication Technology and Networking (CICTN)*, pp. 418–420, IEEE, 2023.
- [13] P. Gupta, N. Paharia, S. K. Gupta, and R. S. Jadon, “Stock price prediction using modified bpso for feature selection with rnn variants on top tech companies,” in *2023 World Conference on Communication & Computing (WCONF)*, pp. 1–7, IEEE, 2023.
- [14] M. Nabipour, P. Nayyeri, H. Jabani, S. S., and A. Mosavi, “Predicting stock market trends using machine learning and deep learning algorithms via continuous and binary data; a comparative analysis,” *IEEE Access*, vol. 8, pp. 150199–150212, 2020.

- [15] Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, and L. Sun, “Transformers in time series: A survey,” *arXiv preprint arXiv:2202.07125*, 2022.
- [16] P. Lv, Y. Shu, J. Xu, and Q. Wu, “Modal decomposition-based hybrid model for stock index prediction,” *Expert Systems with Applications*, vol. 202, p. 117252, 2022.
- [17] S. Siami-Namini, N. Tavakoli, and A. S. Namin, “The performance of lstm and bilstm in forecasting time series,” in *2019 IEEE International conference on big data (Big Data)*, pp. 3285–3292, IEEE, 2019.
- [18] A. Köksal and A. Özgür, “Twitter dataset and evaluation of transformers for turkish sentiment analysis,” in *2021 29th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, IEEE, 2021.
- [19] D. Chicco, M. J. Warrens, and G. Jurman, “The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation,” *Peerj computer science*, vol. 7, p. e623, 2021.
- [20] J. J. Carey and M. F. Delaney, “T-scores and z-scores,” *Clinical reviews in bone and mineral metabolism*, vol. 8, pp. 113–121, 2010.
- [21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [22] S. M. Kazemi, R. Goel, S. Eghbali, J. Ramanan, J. Sahota, S. Thakur, S. Wu, C. Smyth, P. Poupart, and M. Brubaker, “Time2vec: Learning a vector representation of time,” *arXiv preprint arXiv:1907.05321*, 2019.
- [23] G. Tang, M. Müller, A. Rios, and R. Sennrich, “Why self-attention? a targeted evaluation of neural machine translation architectures,” *arXiv preprint arXiv:1808.08946*, 2018.

ORIGINALITY REPORT

7%

SIMILARITY INDEX

4%

INTERNET SOURCES

3%

PUBLICATIONS

3%

STUDENT PAPERS

PRIMARY SOURCES

1 "Advances in Artificial Intelligence and Security", Springer Science and Business Media LLC, 2021
Publication 1%

2 www.ijeast.com
Internet Source <1%

3 www.isindexing.com
Internet Source <1%

4 Submitted to Nanyang Technological University
Student Paper <1%

5 ikee.lib.auth.gr
Internet Source <1%

6 Submitted to Liverpool John Moores University
Student Paper <1%

7 "Data Science and Analytics", Springer Science and Business Media LLC, 2020
Publication <1%

github.com