FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Assignment of bachelor's thesis

Title:	Improving stock price prediction using media analysis
Student:	Kostiantyn Romanov
Supervisor:	Ing. Mgr. Pavla Vozárová, Ph.D., M.A.
Study program:	Informatics
Branch / specialization:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	until the end of summer semester 2022/2023

Instructions

- Study and then describe the basic models for stock price prediction.
- Analyze which media are most relevant to stock market investors. Implement a tool that allows you to automatically download from these media information about selected companies and to perform their basic classification. Also, explore the possibilities of using media databases.
- Search for the possibility of obtaining data on the development of stock prices of companies traded on different stock markets.
- Based on the data you obtain, examine whether it is possible to use media information to improve the prediction of stock prices of selected companies.

Electronically approved by Ing. Karel Klouda, Ph.D. on 17 June 2021 in Prague.

Bachelor's thesis

IMPROVING STOCK PRICE PREDICTION USING MEDIA ANALYSIS

Kostiantyn Romanov

Faculty of Information Technology
Department of Applied Mathematics
Supervisor: Ing. Mgr. Pavla Vozárová, Ph.D. M.A.
January 3, 2023

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis: Romanov Kostiantyn. *Improving stock price prediction using media analysis.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2023.

Contents

List of Figures

List of Tables

I express my gratitude to the supervisor of this thesis, Ing. Mgr. Pavla Vozárová, Ph.D., M.A. for much needed guidance and patience. I also wish to thank my friends and family for their unconditional support, and especially my mother, who despite going through the hardships of a refugee life, has found strength to always be there for me on this journey.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60 (1) of the Act.

In Prague on January 3, 2023 .....

Abstract

Stock price prediction using media analysis has been an active area of research over the last decades. Previous studies analyze the impact of media sentiment on a day-to-day scale, limited by the problem of data availability. In this thesis, I opt to compile a dataset of stock prices and news headlines to study the effect of news sentiment on an intraday scale, where it is hypothesized to be most pronounced. I employ the recently emerged transformer architecture, showing great potential for sequence processing tasks. The proposed model combines the FinBERT model for sentiment classification with the Informer-based model for time-series prediction. This facilitates the analysis of the predictive capabilities of the produced technical indicators and sentiment features over various prediction windows. Although no significant impact of media sentiment was found in a short-term prediction, supporting the claims of the effective market hypothesis, a long-term prediction is shown to benefit from the addition of sentiment features.

Keywords natural language processing, news mining, sentiment analysis, stock price prediction, time-series prediction, transformer

Abstrakt

Předpovídání cen akcií pomocí mediální analýzy je v posledních desetiletích aktivní oblastí výzkumu. Předchozí studie analyzovaly vliv mediálního sentimentu v každodenním měřítku, byly však omezeny problémem dostupnosti dat. V této práci jsem se rozhodl sestavit dataset o cenách akcií a novinových titulcích, abych mohl studovat vliv zpravodajského sentimentu v mezidenním měřítku, kde se podle předpokladu projevuje nejvýrazněji. Využívám nedávno vzniklou architekturu transformerů, která vykazuje velký potenciál pro úlohy zpracování sekvencí. Navrhovaný model kombinuje model FinBERT pro klasifikaci sentimentu s modelem založeným na Informeru pro predikci časových řad. To usnadňuje analýzu predikčních schopností vytvořených technických indikátorů a příznaků sentimentu v různých predikčních oknech. Ačkoli v krátkodobé predikci nebyl zjištěn žádný významný vliv mediálního sentimentu, což podporuje tvrzení teorie efektivních trhů, ukazuje se, že dlouhodobá predikce má z přidání příznaků sentimentu prospěch.

Klíčová slova analýza sentimentu, dolování z novinek, predikce časových řad, předpověď ceny akcií, transformer, zpracování přirozeného jazyka

Abbreviations

EMH	Efficient Marker Hypothesis
AMH	Adaptive Market Hypothesis
NLP	Natural Language Processing
BoW	Bag of Words
TF-IDF	Term Frequency-Inverse Document Frequency
KNN	K-Nearest Neighbors
SVM	Support Vector Machine
RNN	Recurrent neural network
LSTM	Long Short-Term Memory
DL	Deep Learning
TSF	Time Series Forecast
MSE	Mean Square Error

Introduction

Never fall in love with a stock, because it will never love you back [1].

Stock price prediction has been an active topic in the scientific community for at least half a century. Stock prices are often viewed as a leading indicator of economic conditions, so a better understanding of the predictability of stock markets can help economists and policymakers make more accurate predictions about the direction of the economy. Research on the predictability of the stock market can help shed light on the efficiency of financial markets, which is the degree to which prices reflect all available information. A more efficient market is less likely to be affected by irrational behavior, which can lead to more stable prices and fewer opportunities for arbitrage. Learning and modeling the complex processes that influence price formation can allow investors to make more informed decisions and can potentially lead to better returns on their investments and risk aversion.

The emergence and widespread popularity of the Internet since the mid-1990s has had a major impact on financial markets and investment. The level of connectivity that the Internet provided allowed investors to buy and sell securities easier, reducing the costs of trading and increasing liquidity in the financial markets. The World Wide Web has made it much easier for investors to access a wide range of information about potential investments, including financial statements, news articles, and analysts' reports. The widespread access to media allowed for greater transparency and significantly increased the information flow actively influencing the investor sentiment.

Studying media sentiment when predicting stock market movements can be useful because it can help incorporate important external factors that can affect stock prices into the prediction process. Research into the extent to which media sentiment can improve stock price prediction has been lively in recent decades, with a myriad of methods for sentiment analysis developed. Sentiment analysis deals with extracting and classifying sentiment from textual data, such as a tweet or a news article, to quantify it and enable studying of its effects. As investors are prone to biases and reactive behavior influenced by information they gather through the news, using sentiment analysis for stock price prediction can be beneficial and deserves further analysis.

However, both sentiment analysis and time-series forecasting require the processing of inherently sequential data, where past information has a profound impact on the current state, which was found to be challenging to solve with existing methods. Additionally, the low availability of publicly accessible data on financial media articles and stock prices on an intraday scale has presented obstacles to research in the area.

In this paper, I attempt to address the mentioned problems and study the effect of media sentiment on stock price predictability. My motivation is recent significant advances in the area of natural language processing, namely the introduction of transformer architecture. It allows for a much more sophisticated language representation and, therefore, a higher quality sentiment

extraction. It has also been successfully applied in time-series forecasting domain, showing a great potential to move forward the research into the stock price prediction using sentiment analysis.

0.1 Goals of the thesis

The aim of the thesis is to propose a model for stock price development prediction utilizing media sentiment analysis and determine whether sentiment information improves prediction quality. The goals can be listed as follows:

1. Describe the theoretical background for the stock market forecast. Characterize the issue from the point of view of the effective market hypothesis and behavioral finance.
2. Explain how and why sentiment analysis can be beneficial for stock price prediction. Determine which media are relevant to investors. Provide a historical review of sentiment analysis techniques and list previous research that has focused on the matter.
3. Provide a review of methods applied in stock price prediction.
4. Propose a model for the prediction of stock price development, incorporating media sentiment analysis. Explore the possibility of using sentiment analysis in combination with technical analysis to improve results.
5. Implement software tools that perform media scraping and analysis to predict stock price developments based on the proposed model. Consider using media databases.
6. Based on the outcomes achieved, examine whether it is possible to use media information to improve the quality of prediction of stock price development for selected companies.

0.2 Structure of the thesis

The thesis is structured as follows. Firstly, I lay out the theoretical foundation for financial markets predictability in Chapter 1. I describe several approaches to explain market efficiency and present methods for stock price development analysis. I define the concept of investor sentiment, list approaches to its measurement, and the role of media its formation in Chapter 2. Then, I introduce the text classification pipeline and discuss the historical approaches to sentiment extraction and their limitations in Chapter 3. I present the transformer architecture and why it is suitable for sequence transduction tasks in Chapter 4. I give a brief overview of the history of time-series forecasting in Chapter 5. Then, I propose a model for stock price prediction on an intraday scale using media analysis based on the transformer architecture in Chapter 6. I evaluate the model on the stock prices of selected companies in Chapter 7 and analyze the benefits of using media sentiment for the stock price forecast. Finally, I conclude this paper in Chapter 8, where I discuss the fulfilment of the goals of the thesis and propose areas for future research.

Chapter 1

Financial markets prediction: Background

Just because water likes to find its own level does not mean that the ocean is flat [2].

This paper attempts to answer the question of whether media information, available to investors, can be used to improve the quality of stock market prediction through the application of modeling methods and artificial intelligence. This is an interdisciplinary problem that connects the fields of linguistics (extracting information from unstructured text sources), machine learning (providing computational methods for analysis and modeling), and behavioral finance (interpreting information within economic theory). In this chapter, I will describe the economic background for market prediction and theoretical approaches to tackle the problem.

1.1 Efficient Market Hypothesis

The behavior of financial markets is commonly explained by the efficient market hypothesis. Originally introduced by Farma [3], it states that a security's price development can be characterized as a random walk, meaning that an individual price change is independent and markets are, therefore, completely random and unpredictable or *information efficient*. This means that regular investments with a return rate exceeding that of a market average are impossible, making the unsophisticated "buy and hold" strategy the most viable. This would render any kind of market analysis useless, as all information available at a given moment is reflected in the share price and the past history of the price development cannot be used to forecast the future.

Empirical evidence, however, suggests that the theory does not always hold, as price changes shall not be strictly independent. Recognizing this fact, Farma introduces various levels of market efficiency in a later paper [4]. Depending on information availability, markets may be more strongly or weakly efficient: lack of public access to relevant market information, as well as insider trading, would reduce the efficiency. Therefore, the EMH is more applicable in markets with free flow of information, which is not always achievable in practice.

1.2 Behavioral finance

The EMH is based on the notion of an effective arbitrage mechanism, which means that if a security price fails to incorporate relevant information, a significant incentive is created to trade on this information. A strong version of the EMH considers the market adjustment to be near

instantaneous. Studies on the matter, however, show this to be a continuous process of various speed and accuracy [5].

Behavioral finance describes this phenomenon as a result of market agents not being fully rational. Investors are susceptible to various forms of psychological biases, such as rationalization, self-attribution, hindsight bias, confirmatory bias, etc. [6] The theory considers the price of the security to be a purely perceived value, and as a result, price discovery is a never-ending process, and the current price can be at best considered a flawed proxy for the intrinsic value of a security [2]. This finding has put an end to the notion of an a priori efficient market.

The theory of investor sentiment establishes the link between investor attitude towards markets (optimistic or pessimistic) and their behavior. Taking into consideration the impact of market agents' irrationality on price formations, a greater role has been attributed to the media. Actively effecting investors' attitude, it not only reports, but also has a significant impact on market dynamics.

1.3 Adaptive Market Hypothesis

The relevant critique of EMH by proponents of behavioral finance theory prompted a lively discussion in the research community. Many articles have been dedicated to argue the extent to which financial markets can be considered effective [7]. The introduction of the adaptive market hypothesis managed to reconcile the community, leading to a compromise [8]. The theory applies the principles of evolution, competition, and adaptation to financial markets, implying that the degree of market efficiency is related to the number of competitors in the market, their adaptability, and available opportunities for profit. Decision biases attributed by behaviorists to irrationality are, in fact, consistent with an evolutionary model of individuals adapting to a changing environment through basic heuristics [8]. Therefore, the AMH creates a theoretical framework, where classical models of financial economics can co-exist with behavioral finance.

Urquhart and McGroarty [9] have conducted an extensive investigation of the Adaptive Market Hypothesis. Their study examines the stock return predictability in the S&P500, FTSE100, NIKKEI225 and EURO STOXX 50 and attempts to establish the relationship between the level of returns predictability and market conditions. The results show that returns go through periods of dependence and independence, consistent with the AMH. Overall, it argues for adaptive markets providing a more suitable description of stock returns behavior than the EMH.

1.4 Markets predictability

The most important implication of the points mentioned above is that the predictability of financial markets is theoretically substantiated and practically evident in available research. The degree to which financial markets are predictable generally depends on the efficiency of information absorption. Emerging markets, such as those in Southeast Asia or Africa, have been shown to express weak-form efficiency, in contrast to markets in developed economies [10, 11]. Studies also indicated that short-term variants of technical trading rules have better predictive capacity than long-term variants [11].

1.5 Fundamental and technical analysis

Regardless of the degree to which one believes that financial markets can be predicted, it has generally been established that market analysis can be beneficial to deepen the understanding of various market dynamics and their causes. Broadly speaking, numerous approaches at attempting to model market behavior can be generalized into two camps: those of technical and fundamental analysis.

Technical analysis is the process of analyzing a security's past prices in an attempt to determine its future value. The rationale for technical analysis lies in the belief that market movements tend to repeat themselves, that the future can be found in the past [12]. Technical analysts discover visual patterns and repetitions of graph movements utilizing mathematical modeling and numerous pattern recognition techniques. The uncovered regularities in market development are typically named and serve as the foundation for technical predictive methodology.

Common techniques include the application of various graph overlays, such as moving average rules, Bollinger bands, support and resistance price levels, etc. Analysts use relative strength rules, filter rules, and trading range breakout rules [13]. In statistical analysis, a variety of autoregressive modeling methods are employed [14]. Recent advances in artificial intelligence have stimulated research on the application of machine learning-based models.

The simplicity and straightforwardness of this approach to market forecasting have ensured its wide popularity among market participants. However, the research community has shown more caution in recognizing its effectiveness. Numerous studies attempted to test the theory in practice, reaching mixed results, putting the predictive power of such methods in question [11]. One of the main drawbacks of technical analysis is that it does not concern itself with uncovering the root causes of the established patterns, stating nothing more than that the patterns exist.

While technical approaches are hardly capable of identifying the reasons behind market behavior, it is the primary task of their main alternative, *fundamental analysis*. Fundamentalists' goal is to determine factors relevant to uncovering the intrinsic value of a given security. Trading strategies are then developed depending on how the fundamental evaluation compares to the market price.

The data considered to be required for the successful valuation of a security can be wide-ranging. In evaluating a stock price, company-specific information such as changes in inventories, gross margins, selling expenses, etc. might prove relevant [15]. Similarly, one could analyze general economic performance, market indices, data concerning government activities, monetary policy, effective tax rates, as well as labor force productivity, all of which might have an effect on intrinsic value not yet incorporated into the market price [16].

Methods for fundamental analysis can be generalized into *quantitative* and *qualitative*. The former focuses on analyzing the financial statements to make investment decisions. Such statements include balance sheets, income statements, as well as cash flow statements. Qualitative analysis tries to evaluate competitive advantage, management strategies, and business model, as well as the prospects of relevant industries [17].

Some studies managed to form a portfolio based on fundamental factors, claiming to have shown abnormal return rates [15]. The benefit of fundamental analysis has also been attributed to the success of short-sellers [18]. Nevertheless, a significant problem of fundamental analysis is that it is highly challenging to generalize the process of determining the factors relevant to the specific security's value. Hence, such methods were rarely successfully automated [16].

It has been common for market participants to resort to both technical and fundamental approaches to market analysis. Multiple researchers have supported the idea that utilization of both investment techniques would lead to more successful investing decisions for traders, proving the complementary nature of fundamental and technical analysis [1]. Fundamental data, however, is often relayed in an unstructured, textual form, remaining to be a challenge to make the best use of it efficiently through computing. While mathematical and statistical methods are well capable of modeling dependencies within numerical data, extracting meaning from unstructured text is no trivial task.

Chapter 2

Investor sentiment

Animal spirits [19].

The previous chapter laid out the theoretical foundation for financial markets prediction. Both technical and fundamental analysts base their approaches on the irrational nature of investors' decision-making. In this chapter, I will explore how researchers define this irrationality, what are the approaches to its measurement, and what it is influenced by.

2.1 Investor sentiment

Investor sentiment can be described as one's belief about future market developments, cash flows, and investment risks, not necessarily backed by the facts at hand. The general attitude of investors towards a particular market is referred to as *market sentiment*. It is commonly assumed that rising prices indicate a "bullish" investor sentiment, while falling prices, to the contrary, a "bearish" one. Popular among average investors, this idea has not been consistently supported by research on the matter, trying to uncover the complex relationship of investor opinion and market behavior [16].

To explain the influence that investor sentiment has on stock market developments, researchers turn to behavioral finance. Some have made an assumption that it is costly and risky to bet against predominant sentimental investors. As a result, arbitrage forces generated by rational investors are not as impactful in pushing prices to their fundamental values, as EMH would suggest [20]. The assumption can be supported by events in recent stock market history, such as the dot-com bubble in the late 1990s and the US housing bubble of 2007-2008.

2.2 Approaches to sentiment measurement

Having firmly established that investor sentiment has a measurable impact on the stock market, it remains a challenge to accurately measure and quantify this effect. The widespread approach is to compile investor sentiment indices through the use of *sentiment proxies*. Such proxies can be divided into *direct* and *indirect*, depending on the calculation and data source.

Direct proxies include investor surveys and questioners. Multiple studies attempted to predict the stock market based on investors' own opinion, with mixed results [21]. Consumer Confidence Index, even though not directly related to prices of securities, has shown some correlation with small stock returns as well [22].

Indirect proxies use the corresponding objective data already available in the financial markets and elsewhere. One such proxy is data on retail investor trades, as unprofessional and younger

investors are more susceptible to be subject to sentiment [23]. Trading volume, or liquidity in general, can also be viewed as a proxy. It has been shown that optimistic investors are more likely to trade, thus adding liquidity [24]. Some papers have also attempted to connect stock prices to changes in human emotions. For example, market returns have been shown to be lower in the fall and winter, which has been attributed to the seasonal affective disorder [25]. Losses in the international major football matches have also been associated with poor next-day returns in the losing country [26].

None of the proxies are, however, without a flaw. Some of them contain idiosyncratic components that have no relation to investor sentiment [20]. Some are additionally reflecting economic fundamentals to a certain extent. Another significant drawback is data availability: much of the data required to comprise a sentiment index is either lacking or not granular enough for short-term forecast, where sentiment effect has been shown to be most pronounced 1.4.

2.3 Role of media in sentiment formation

The role of media in the process of market adjustment has always been undeniable: in the end, the free flow of information is a requirement for market efficiency. However, the appearance and accessibility of the Internet have arguably multiplied the impact of media on market behavior. Investors are now able to receive the latest news about price developments almost instantly, they share their sentiment on multiple social media platforms, they communicate freely, influencing each other's opinion. So, how does the media affect investor sentiment?

The impact of media on investors' decision-making has been a hot topic among researchers for at least the past two decades. Antweiler and Frank [27] classified chat room messages as "buy", "hold", or "sell" signals. They indicated the existence of a relation between message activity and return volatility, although no statistically significant effect on stock returns was found. Tetlock [28] studied the interactions between the media and the stock market using a Wall Street Journal column. He found that high media pessimism predicts a decrease in market prices with a later reversion to fundamentals.

Financial news can potentially serve as a proxy for investor sentiment, as investors may react to new information reported in the news. For example, if a company announces positive earnings or a new product launch, investors may become more optimistic about the company's prospects and be more likely to buy its stock. On the other hand, if a company announces negative earnings or is facing legal or regulatory issues, investors may become more cautious and less likely to buy its stock. Multiple studies have looked at the impact of financial news on stock prices with inconsistent results. Schumaker et al. [29] built a financial news article prediction system that extracts the author's tone from news articles. They discovered that market traders tend to behave in a contrarian manner, e.g., see good news, sell; see bad news, buy. Khan et. al. [30] used classification algorithms on social media and financial news data and discovered their beneficial impact on stock price prediction accuracy. To the contrary, a study by Thomas Renault [31] has concluded that while investor sentiment and stock returns are highly correlated, investor sentiment derived from messages sent on social media was not found to help in predicting large capitalization stocks return at a daily frequency.

Media information is typically available in the form of unstructured textual data, which is notoriously difficult to analyze. However, recent advances in NLP caused a wave of research on the impact of media on the stock market. In the next chapter, I will discuss the methods for information extraction from textual sources and how they can be applied to the stock market.

Chapter 3

Sentiment analysis

You shall know a word by the company it keeps [32].

Natural language processing is an area of research that explores how computational power can be utilized to understand and manipulate text and speech. NLP researchers attempt to gain insight into how human beings understand and use language to develop methods that allow computer systems to understand and manipulate natural language to perform various tasks. NLP has its foundation in such disciplines as linguistics, psychology, mathematics, artificial intelligence, etc. Its typical applications include machine translation, text processing and summarization, cross-language information retrieval, speech recognition, etc. [33]

Another popular application of NLP is *sentiment analysis*, also sometimes referred to as *opinion mining*. Sentiment analysis is the computational study of people's opinions, emotions, and attitudes towards a certain entity or topic preserved in text. Sentiment classification methods typically label a document associated with an opinion as positive or negative.

In the last decades there has been a boom in the research into sentiment analysis techniques, as huge amounts of textual data became more and more available thought social media and news platforms. Large volumes of digitized opinionated data is shared online on the Internet in forms of tweets, comments, reviews, discussions, blogs, news feeds, etc. The analysis of this data has proved useful and spread from computer science to marketing, finance, political science, health, and communications.

3.1 Sentiment analysis tasks

The common way to study sentiment analysis is on three levels of granularity: document, sentence, and aspect level.

*Document-leve*l sentiment classification analyses and classifies a document imbued with an opinion as overall positive or negative. Examples of such documents are product reviews, customer complaints, or financial reports. This type of analysis considers the document as a whole expressing an attitude towards a single known entity from a single opinion holder [34].

Sentence-level sentiment analysis categorizes individual sentences in a document. It is very similar to document analysis, as a sentence can be considered a short document. Unlike a document, thought, which is considered to express an opinion, not every sentence has to be opinionated. Determining whether or not a sentence expresses an attitude is called subjectivity classification. Then the resulting opinionated sentences are again categorized by sentiment polarity. Thus, sentence level analysis can by considered a three-class classification problem, with negative, neutral, and positive classes [35].

Compared with the previous two, *aspect (or aspect-based)* sentiment analysis is more fine-grained. It attempts to extract and summarize individuals' opinions on a certain entity and aspects of such entity, also referred to as targets. For example, in a movie review, it aims to gather positive and negative attitudes towards different features of the movie, although the overall sentiment could be positive or negative. Therefore, aspect level sentiment analysis could be divided into subtasks such as aspect extraction, entity extraction, and aspect sentiment classification [34].

Other sentiment analysis tasks include emotion analysis, where the focus is on extracting expressed emotions, such as joy or anger, rather than positive or negative sentiment. Sarcasm detection has also been an important but challenging aspect of sentiment analysis. Much research has been aimed at multilingual sentiment analysis problems etc. [35]

In the next section, I will briefly describe the traditional approaches to sentiment analysis and then go into more details on the latest developments in the field.

3.2 Traditional approaches

Sentiment analysis is a subtask of text classification, which can be defined as follows. The input pipeline is comprised of some raw textual dataset, where each instance is a document, paragraph, or sentence. Each instance is then processed by the selected algorithm and classified into one of the sentiment classes studied. The typical text classification pipeline includes text preprocessing, feature extraction, optional dimensionality reduction, classification, and evaluation.

The original form of natural language is not suitable for serving as input for computational models. It contains many unnecessary words, misspellings, slang, etc. In most algorithms, unnecessary features and noise can have a negative effect on performance. Therefore, a number of text preprocessing methods are utilized to clean up text data sets, allowing for a more efficient text featurization [36]. These methods include *tokenization*, *stop words removal*, *capitalization removal*, *spelling correction*, *stemming*, and *lemmatization*, etc. It is worth noting that different preprocessing steps are required by different algorithms and their use is not always beneficial.

Feature extraction is the process of transforming preprocessed textual data into numerical features that can be processed by the algorithms. A number of feature extraction and representation techniques were introduced in an attempt to preserve the document's meaning and focus. Most notable include *n-gram modeling*, *bag-of-words document representation*, and *Term Frequency-Inverse Document Frequency*. BoW models in particular encode each word in a vocabulary as a one-hot-encoded vector, and a document vector is then formed as a sum of word vectors.

The created feature representations are then processed using classification algorithms. *Naive Bayes Classifier*, *K-Nearest Neighbors*, *Support Vector Machine* are among the most popular in the research community and have been widely applied to the problem of sentiment classification [36].

The feature representations discussed inevitably lead to high-dimensional vector models with an increase in vocabulary size. Sparse document representations are highly difficult to model with increasing time complexity and memory consumption. It is also challenging for computational models to extract meaningful dependencies from a large representational space.

3.3 Word embedding

The limitations of previous approaches to word representation motivated further research in distributional semantics. Following the principle that linguistic items with similar distributions tend to have similar meanings, Begio et al. [37] investigated the possibility of distributed representation of words. Such representations, also called word embeddings, attempt to capture the characteristics of the word's immediate context. It commonly involves the transformation of

words in a vocabulary to a vector of real numbers: the original high-dimensional vector space (e.g., one-hot encoding) is embedded in a lower-dimensional dense vector space. Therefore, word embedding is a vector of real numbers, where each dimension represents the latent feature of a word.

A surprising characteristic of such word representations is the explicitly encoded patterns and similarities in language, many of which can be expressed as linear translations [38]. As an example, adding word vectors for "man" and "royal", would result in a vector for a semantic composite of the words – "king".

Word embeddings are commonly learned using a shallow neural network, as in a *word2vec* system, introduced by Mikolov et al. [38] They proposed *Continuous Bag-of-Words* and *skip-gram* language models. The first one strives to predict a word based on the embedding of neighboring words, while the other one predicts the surrounding context based on the embedding of the current word. *Global Vectors for Word Representation* is another popular approach [39]. The model utilizes global matrix factorization and local context window methods and is trained only on nonzero elements in the word-word co-occurrence matrix.

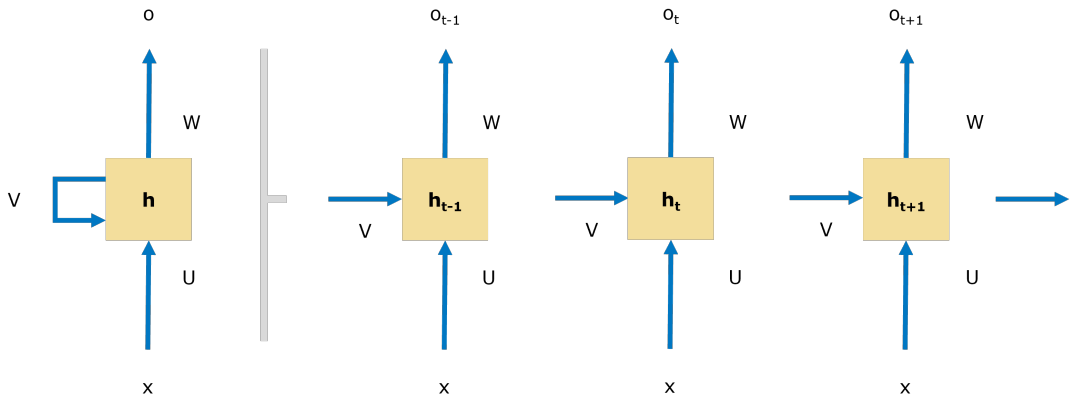
Dimensionality reduction, as well as the aforementioned properties of word embeddings, propelled their popularity and caused the emergence of novel methods for sentiment analysis, in particular using deep neural networks.

3.4 Deep learning

Basic deep neural networks are formed of multiple layers consisting of nodes (neurons) that every single layer only receives information from the previous layer and provides it to the next one. The input layer constructed by the feature extraction method connects the input feature space with the first hidden layer. The number of nodes in the output layer is equal to the number of classes in a multiclass classification and only one for binary classification. The neural network is trained using a backpropagation algorithm, common choices of an activation function are *sigmoid*, *ReLU*, and *tanh*. A *Softmax* function is used for the output layer in multiclass classification.

Deep learning has been widely used in various NLP applications. Most popular models include Convolutional, Recursive, and Recurrent Neural Networks. I will focus on the latter one, as its architecture is naturally suited for token-level sequential labeling tasks.

Figure 3.1 A simple RNN network unfolded through time



3.4.1 RNN

The recurrent neural network models units in sequence, utilizing the memory of the previous unit when moving to the next. This is especially useful in NLP applications, where units are symbols, words, or whole sentences. Sequential processing of words in a sentence allows RNNs to capture a wider range of semantic information, as previous words have an impact on the meaning of the following *("dog" vs. "hot dog")*. Such networks can take a variable-length input. These characteristics make RNNs in combination with word embeddings a very powerful tool for modeling complex language representations [40].

A basic RNN architecture is shown in Figure 3.1 unfolded through time to better illustrate the process of "remembering". In the figure, x_t represents the input to the RNN at time step t and h_t is the hidden state at the same step. The number of steps is given by the input length. h_t is calculated as follows:

$$\mathbf{h}_t = f(U\mathbf{x_t} + W\mathbf{h_{t-1}}) \tag{3.1}$$

Therefore, h_t incorporates information regarding the current input, as well as the hidden state of the previous time step. The function f is an activation function and U, V, W account for weights that are shared over time. The hidden state of the RNN accumulates the information from previous states, serving as the network's *memory cell*.

Theoretically, RNNs should be capable of taking input of unlimited length, in practice, however, such networks can only use information from a few time steps back. Otherwise, they are susceptible to the problems of vanishing or exploding gradients [41].

3.4.2 LSTM

The long-short-term memory network is a type of RNN, which can learn long-term dependencies, solving the problem of the vanishing gradient. LSTM employs additional gates: input, forget and output gates, to regulate what information is allowed to pass into further states.

3.4.3 Bidirectional RNN

Traditional RNNs often face the problem of bias towards previous words, when in language later words may be more influential. Addressing this problem is a bidirectional RNN, which looks at both the left and right context [42]. Bidirectional RNN is comprised of two RNNs, stacked on top of one another: the first is processing input in original order, the second in reverse order. The calculation of output is performed on the basis of the hidden states of both RNNs.

3.4.4 Limitations

RNNs have achieved great success in sequential data processing. They are, however, held back by several limitations. Although the problems of the vanishing and exploding gradient have been mitigated by more complex models, other are inherently intertwined with the prime strengths of the networks. Naturally suited for sequential processing, RNNs require the previous state to compute the current one. This makes it impossible to parallelize the network training process, making it computationally heavy, slow, and complex.

Word embeddings, commonly used as input to neural networks, are also bound by drawbacks. The main one is their inability to represent multiple meanings of a single word, or properly capture the context to represent multi-word phrases.

In the next section, I will discuss the Transformer model, which aims to solve these problems.

Chapter 4

Transformers

Attention is all you need [43].

The limitations of RNN apply generally to the sequence-to-sequence models and prompted further research in the area. The recent transformer network, based on the encoder-decoder architecture, drops the recurrence completely and instead relies entirely on the mechanism of attention. In this chapter, I will discuss the concepts which led to appearance of transformers, explain the architecture of the model, and how it can be applied to NLP tasks including sentiment analysis in the financial domain.

4.1 Encoder-Decoder architecture

The encoder-decoder architecture lies at the core of all successful RNN models for sequence transduction problems, particularly machine translation. The encoder processes the variable-length input (source sentence) and constructs a fixed-length internal feature representation. Depending on this representation, the decoder then generates a variable length output (target sentence) [44].

The *encoder* is an RNN that processes each symbol of the input sequence sequentially, updating the hidden states with each new symbol read. After reading the end of the sequence, the hidden state of the RNN is a representation of the whole input sequence.

The *decoder* is another RNN that is trained to generate the output sequence by predicting the next symbol given the current hidden state. This hidden state is calculated based on the output so far, previous hidden state of the decoder, and on the representation of the input sequence [45].

The encoder and the decoder are two separate units, which means that they could be trained to best perform their respective task. This architecture with various improvements became a popular approach to solving problems like automatic translation, text summarization, and speech recognition. One possible way to improve the performance of the architecture is the introduction of an attention mechanism.

4.2 Attention

One of the problems of the traditional encoder-decoder model is that the encoder treats every piece of information equally, even though not all of it might be relevant to the task. This problem is especially pronounced in the cases of long and information-rich input. An intuitive example is the summarization problem. It is unrealistic to expect a fixed-size vector to preserve all the information in a text input, which length is potentially very long [46].

In most sequence-to-sequence tasks, a certain relation exists between the input sequence and the output, meaning that each token generation step is in some way related to a particular part of the input. This assumption is the motivation behind *attention mechanism*. Attention attempts to resolve the problem by allowing the decoder to refer back to a specific part of the input text. Specifically, during the decoding process, the decoder is conditioned on the last hidden state, the output generated so far, and a "context" vector dependent on the input hidden states.

With an attention mechanism, the encoder no longer encodes the full input into a fixed-size vector. Instead, the decoder is allowed to "attend" to different parts of the source text at each step of the output generation. Each decoder output word now depends on a weighted combination of all the input hidden states, not just the last one. These weights are fully learned by the model in the training process. Additionally, they are typically normalized to form a distribution over the input states.

The attention mechanism proved to be a highly beneficial modification to existing models across multiple domains. As researchers focused more on its applications, a new architecture was introduced to maximize its benefits.

4.3 Transformer architecture

The *transformer architecture* was introduced in a paper by Vaswani et al. [43] The main contribution of the paper is the decision to rid of recurrence units entirely, instead relying fully on an attention mechanism to model dependencies between input and output. This allows for parallelization of much of computation required to train the model, eliminating one of the most significant drawbacks of RNNs. The transformer network introduced in the paper follows the typical encoder-decoder structure. Both an encoder and a decoder contain a stack of two core blocks: an attention block and a feed-forward network. I will first explain the details of an attention mechanism.

4.3.1 Self-Attention

Self-attention is the fundamental operation of a transformer. It is a sequence-to-sequence operation: a sequence of vectors $x_1, x_2, ..., x_L$ is taken as input and a sequence of vectors $y_1, y_2, ..., y_L$ is produced as output. All vectors have dimension d, which I will refer to as a model dimension. To produce an output vector y_i, the self-attention takes a weighted average over all input vectors, the simplest option being a dot-product:

$$y_t = \sum_{j=1}^{L} w_{ij} x_j, \tag{4.1}$$

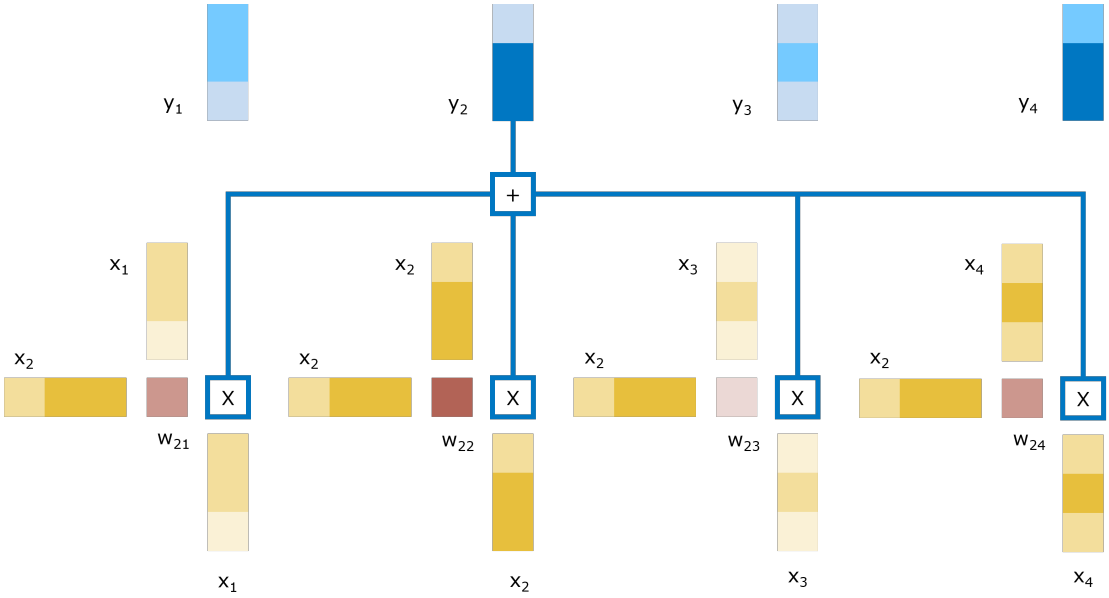
$$\text{where} \quad w_{ij} = \frac{\exp(w'_{ij})}{\sum_{l=1}^{L} \exp(w'_{il})} = softmax(w'_{ij}), \tag{4.2}$$

$$w'_{ij} = x_i x_j^T. \tag{4.3}$$

This is the only operation in the transformer architecture that propagates information between input vectors.

The vector x_i is a learned embedding vector for i^{th} word in the input sequence. Since the embeddings are learned, the relation between two given words is completely dependent on the task. To provide a general example, the definite article *the* is not typically relevant to the interpretation of words in a sentence, therefore its embedding is likely to have low or negative dot product with other words. On the contrary, interpreting the meaning of the word *come* is heavily dependent on the nouns in a sentence as well as the prepositions that potentially follow it, so it is likely to have high dot-products with words like *student* or *across*.

Figure 4.1 An illustration of the self-attention process. Adapted from [47]



4.3.1.1 Queries, Keys and Values

The mechanism of self-attention, as described above, is entirely parameterless (not considering the embedding layer). To give the self-attention layer some controllable parameters, the concepts of *queries*, *keys* and *values* are introduced. Each vector x_i from the input sequence is used in three different contexts:

1. It is a part of the computation for its own output y_i
2. It is a part of the computation for the output of every j^{th} output y_j
3. It is a part of the weighted sum to compute y_j once the weights have been established.

The roles x_i plays in these contexts are referred to as *query*, *key* and *value* respectively. To perform the roles, new vectors are derived from x_i applying linear transformations, as follows:

$$q_i = W_q x_i, \quad k_i = W_k x_i, \quad v_i = W_v x_i \tag{4.4}$$

$$w'_{ij} = q_i k_j^T, \tag{4.5}$$

$$w_{ij} = softmax(w'_{ij}), \tag{4.6}$$

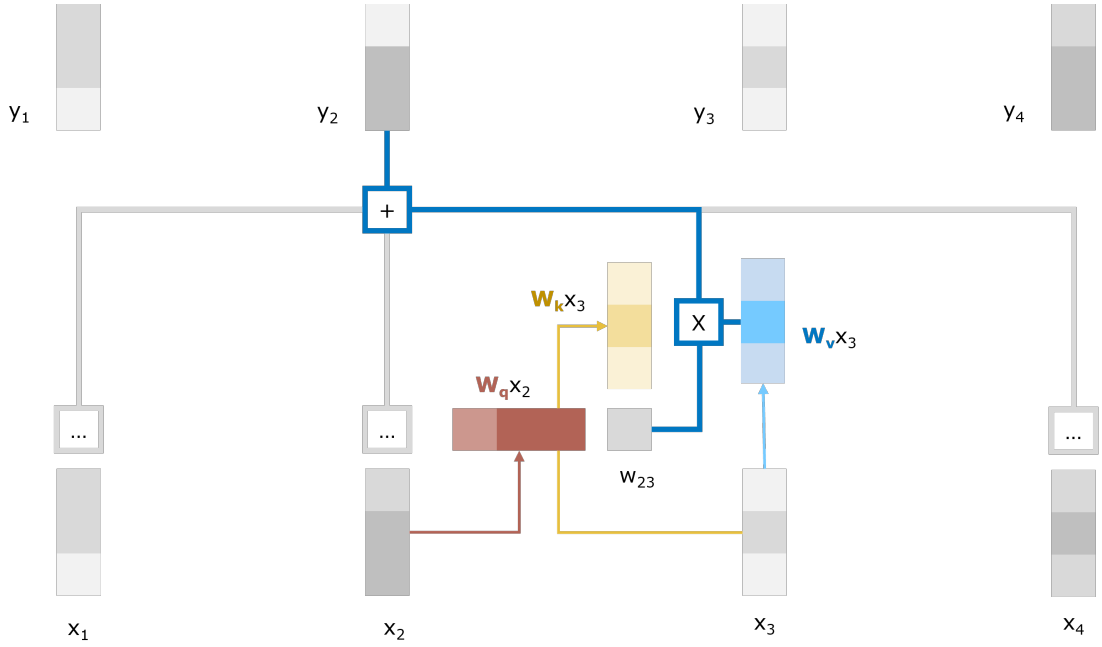
$$y_t = \sum_{j=1}^{L} w_{ij} v_j, \tag{4.7}$$

where W_q, W_k, W_v are $d \times d$ matrices of learned parameters. The transformations are illustrated in Figure 4.2.

4.3.1.2 Dot product scaled

The average value of the dot product increases with the embedding dimension d. The softmax function proved to be sensitive to large inputs, negatively impacting the learning process [43].

Figure 4.2 Query, key, value martices. Adapted from [47]



To mitigate this effect, the dot product is scaled by the amount that the increase in dimension increases the average length of vectors:

$$w'_{ij} = \frac{q_i k_j^T}{\sqrt{d}} \tag{4.8}$$

Hence, the whole self-attention operation can be derived as follows:

$$\mathcal{A}(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}})V, \tag{4.9}$$

where $Q \in \mathbb{R}^{L_Q \times d}, K \in \mathbb{R}^{L_K \times d}, V \in \mathbb{R}^{L_V \times d}$ are matrices of query, key and value vectors respectively.

4.3.1.3 Multi-head attention

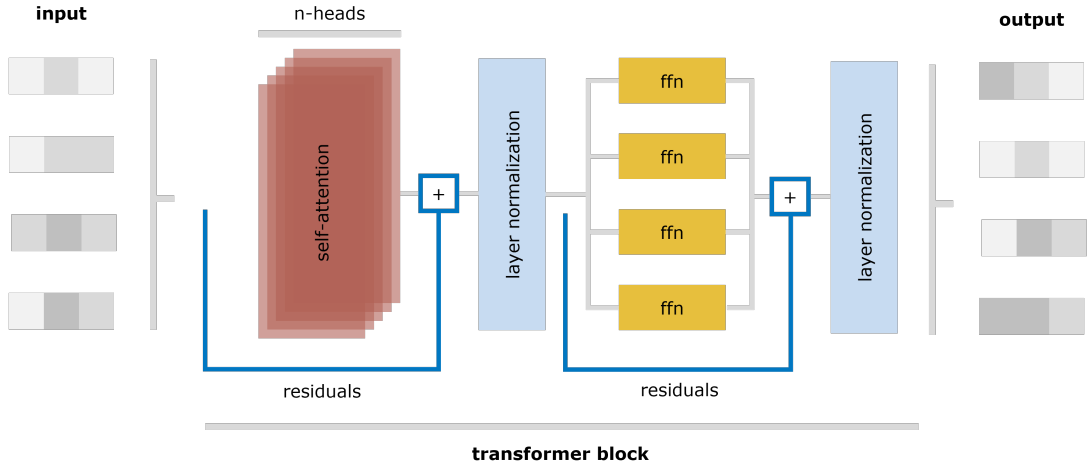
Up to this point the self-attention mechanism fails to account for the fact that a word might have various meanings. To address this issue, several self-attention blocks can be combined, each with different matrices $W_q^r, \quad W_k^r, \quad W_v^r$. These blocks are referred to as *attention heads*.

Given an input x_i each attention head computes a different vector y_i^r. These are later concatenated and subjected to a linear transformation with learned weights to reduce the dimension back to d.

4.3.2 Transformer block

The transformer networks typically consist of a number of the so-called *transformer blocks* stacked together. A generic transformer block can be seen in Figure 4.3. It contains, in sequence, a self-attention layer, layer normalization, a position-wise fully-connected feed-forward layer, meaning

Figure 4.3 Typical transformer block. Adapted from [47]



that it is applied to each input vector separately and identically, and another layer normalization. Residual connections are added around both layers before normalization. Performing normalization and residual connections has been shown to significantly improve the accuracy of the model [48].

4.3.2.1 Input

As I have mentioned above, the sequential inputs of the model first go through a learned embedding layer. Word embeddings are used to represent the words. However, until now, the whole network has been permutation-invariant, which is not an intuitive choice when dealing with sequential information. Information about the position of words in a sentence needs to be passed to the network. This is achieved by producing another vector that incorporates positional information and adding it to the word embedding vector. There are commonly two approaches to this:

- The learned *positional embeddings* are an equivalent of word embeddings but trained to represent position in a sentence. The main drawback of using positional embeddings is the fact that sequences of all expected length must be seen during training, otherwise the relevant embeddings simply do not get trained.
- *Positional encoding* is a certain function that assigns positions to real number vectors used instead of learned embeddings. The benefit of such approach is that the choice of a well-suited function enables the networks to deal with sequences longer than those seen during training. The choice of the function is a complicated hyperparameter.

4.3.2.2 Output: classification

Transformers are inherently sequence transduction models, which means that they output another sequence. The common technique to perform classification with such models is applying global average pooling to the output sequence. Then the produced vector is mapped to the class vector and the softmax function is applied to get the probabilities.

4.3.2.3 Output: auto-regression

Transformers may also be used as autoregressive models. I will consider a task of predicting the next character at each point in a sequence as an example. The target output is the input sequence shifted one symbol to the left. Given a transformer described above, the task is trivial: as the output depends on the entire input sequence, predicting the next character requires simply retrieving it from the input.

Using the self-attention mechanism as an autoregressive model requires preventing leftward information flow up to and including current position. This can be achieved by applying a *mask* to the matrix of dot products, before applying the softmax function. Illegal connections are thus set to $-\infty$.

4.4 BERT

BERT is one example of a contemporary model based on transformer architecture. Introduced in [49], it stands for Bidirectional Encoder Representations from Transformers. It is a state-of-the-art model for various language tasks such as question answering, language inference, and sentiment classification.

The training process of BERT consists of two steps: *pre-training* and *fine-tuning*. During pre-training, the model is trained on a large corpora of data over various pre-training tasks. Fine-tuning is a process of retraining the model initialized with pre-trained parameters on a set of labeled data dependent on the downstream task, thus fine-tuning the parameters. This approach is highly common for NLP models. It utilizes transfer learning and makes the model accessible, as the computational heavy process of learning language representations can only be performed once during pre-training.

Learned commonly applicable representations of words have become an integral part of modern NLP models in the form of pre-trained word embeddings. BERT is a generalization of that approach using a masked language model to enable pre-trained deep bidirectional representations. The following sections provide more details.

4.4.1 Model architecture

BERT's architecture is unified across different tasks with minimal differences between the pre-trained and final downstream architecture. It is a multi-layer bidirectional transformer encoder. The model parameters are the number of layers (or transformer blocks) L, the hidden size H and the number of self-attention heads A. The original paper [49] describes two models $\text{BERT}_{\text{base}}$(L=12, H=768, A=12: 110M parameters) and BERTlarge(L=24, H=1024, A=16: 340M parameters).

4.4.2 Input and Output Representations

To enable BERT to perform a variety of downstream tasks, model's input representation is able to represent both a single and a pair of sentences. *Sentence* in this context means a continuous text span, not necessarily a linguistic sentence. A *sequence* refers to the input token sequence, thus a single sentence or a pair of sentences packed together.

BERT employs *WordPiece* tokenization, which is a subword segmentation algorithm. It attempts to solve the issues faced by the word-level and character-level tokenization, such as wast vocabulary and different meaning of similar words, as well as large length of sequences and meaningless individual tokens respectively. Its general approach is to split rarely occurring words into meaningful sub-tokens. For example, a word *car* will not be split, while a word *cars* will be split

into *car* and *s*. This enables the model to draw connections between similar words and reduces the size of the vocabulary.

Every input sequence in BERT stats with a special classification token $[CLS]$. Classification tasks use the final hidden state of this token as the aggregate sequence representation. Differentiating between sentences in a sequence is achieved using a special $[SEP]$ token. Additionally, a learned embedding is added to every token, indicating whether it belongs to the first or second sentence.

Therefore, an input representation for a given token is constructed by adding the token, segment, and position embeddings.

4.4.3 Pre-training tasks

The pre-training corpus is formed of the BookCorpus (800M words) and English Wikipedia (2,500M words), not including markup. Two unsupervised tasks are used for BERT's pre-training:

- *Masked LM* is BERT's take on solving the problem of bidirectional conditioning typical for transformer-based NLP models. It is a process of randomly masking some percentage of the input tokens and then predicting them. The final hidden representations of the masked tokens are fed into an output softmax layer over the vocabulary.
- *Next Sentence Prediction* is performed with the objective of teaching the model to understand the sentence relation. The dataset can be trivially assembled from the corpus itself by choosing two sentences A and B, with sentence A actually following sentence B in 50% of cases, otherwise being chosen randomly. As mentioned above, the $[CLS]$ is used for next sentence prediction.

The pre-training procedure requires a large amount of computational power. Training of $BERT_{\text{base}}$ was performed on 4 Cloud TPUs in Pod configuration (16 TPU chips total). Training of $BERT_{\text{large}}$ was performed on 16 Cloud TPUs (64 TPU chips total). Each pre-training took 4 days to complete [49].

4.4.4 Fine-tuning

Fine-tuning can be easily achieved by replacing the inputs and outputs of the model with the appropriate ones and fine-tuning the parameters end-to-end. At the input, the aforementioned sentences A and B may represent:

- sentence pairs in paraphrasing task,
- hypothesis-premise pairs in language entailment,
- question-passage pairs in question answering,
- a text-$\varnothing$ pair in text classification or sequence tagging.

At the output, the token representations are used for token-level tasks and the $[CLS]$ token - for classification.

Fine-tuning is a vastly less resource-heavy task when compared to pre-training. The original paper states that all of the results can be replicated in at most 1 hour on a single Cloud TPU.

The effectiveness of BERT propelled it to the forefront of NLP research and led to a number of *BERT-like* models, tailor made for specific domains. The next section describes such a model for the financial domain.

4.5 FinBERT

The need for financial domain-specific NLP models arises as general-purpose models have been shown to lack the ability to properly process specialized language in a financial context. When considering the financial sentiment analysis task, a severe shortage of labeled data is another significant problem. Therefore, a transfer learning model like BERT seems to provide an elegant solution, as it can be further pre-trained on domain-specific language and fine-tuned on a few labeled examples.

Such a model was introduced in [50] and is referred to as FinBERT. It was shown to achieve state-of-the-art results for two financial sentiment analysis datasets and outperform the general-purpose model.

4.5.1 Further pre-training

FinBERT is further pre-trained on a financial corpus in an attempt to teach the model financial context-specific representations. The paper uses a subset of Reuters' TRC2 news article dataset, which includes 46,143 documents with more than 29M words relevant to the financial domain.

4.5.2 Sentiment classification

The process of fine-tuning FinBERT for text classification is identical to the original model described above. The classifier network is trained on the labeled sentiment dataset. The main sentiment analysis dataset used in the paper is Financial PhraseBank [51]. It consists of 4845 English sentences randomly selected from financial news articles, annotated by 16 people with background in finance and business. FiQA Sentiment [52] is another dataset used for the model's fine-tuning, consisting of 1,174 financial headlines, each assigned a sentiment score.

4.5.3 Model performance

FinBERT showed superior results when compared to other pre-trained language models achieving state-of-the-art, even though its performance was only marginally better than that of general-purpose BERT model. More importantly, the model, fine-tuned on from around 500 to 3000 labeled examples in different training configurations, managed to disprove the notion of NLP models requiring a vast amount of data to be effective.

Chapter 5

Time-series prediction

Stock price prediction is at its core a time-series forecasting problem, as stock prices tend to change over time in a systematic way. Time-series forecasting involves using historical data to make predictions about future events. Various models have been introduced by machine learning researches, with deep learning applications being the latest trend. In this chapter, I will give an overview of ML and DL methods for time series prediction and their applications in the financial domain.

5.1 Statistical approaches

Time-series forecasting and their application in finance have been studied extensively for years. Before the rise of popularity of machine learning techniques, statistical methods were generally applied to analyze and predict stocks. Typical methods of statistical analysis are Auto-Regressive Moving Average (ARMA), the Auto-Regressive Integrated Moving Average (ARIMA) etc. [53]

Autoregressive moving average (ARMA) models and autoregressive integrated moving average (ARIMA) models are statistical models that can be used to analyze and forecast time series data.

An ARMA model is a combination of an autoregressive (AR) model and a moving average (MA) model. An AR model is a type of model that uses past values of a time series to predict future values, while an MA model is a type of model that uses past errors in prediction to forecast future values. The AR and MA components of an ARMA model can be combined in different ways to form different models.

An ARIMA model is similar to an ARMA model, but it also includes differencing of the data to make it stationary (i.e., to remove trend and seasonality). This is done by taking the difference between consecutive observations in the time series, which can help to stabilize the variance and remove any trends that may be present.

To use an ARMA or ARIMA model for stock price prediction, one would start by fitting the model to historical stock price data. Once the model has been trained, you can use it to make predictions about future stock prices. It is important to note that these models are based on statistical assumptions, and their accuracy will depend on the underlying patterns in the data.

Ariyo et al.[54] presented an ARIMA-based stock price predictive model. Analyzing data from the New York Stock Exchange (NYSE) and Nigeria Stock Exchange (NSE), they concluded that the ARIMA model had a strong potential for short-term prediction. However, Rathnayaka et al. [55] presented an empirical study of stock price prediction models that compared the performance of ARIMA models with that of artificial neural networks. Their study showed that the forecasting using the artificial neural networks method has higher accuracy value than the results with the ARIMA method.

5.2 Machine learning

Machine learning has been widely studied for its ability to predict the stock market. Time series forecasting involves using historical data to make predictions about future events, and machine learning algorithms are well-suited to this task as they can learn patterns and relationships in data. Machine learning techniques usually require training an algorithm of choice to automatically assign the input data to the given output data. Typical tasks are classification, in case of stock trend prediction, and regression for stock price forecast. Several algorithms have been used for these purposes. Ballings et al. [56] provided a comparison review of the most popular ones, from Random Forest to Support Vector Machines, Logistic Regression, and Neural Networks. For long-term stock forecast Random Forest was the best performing algorithm, as the study concluded.

Throughout the literature, financial time series forecasting is mostly considered a regression problem. However, there is also a significant number of studies, particularly on trend prediction, that use classification models to tackle financial forecasting problems [57].

Different kinds of deep learning models were utilized for stock price prediction, such as Deep Multilayer Perceptron (DMLP), RNN, LSTM, CNN, Restricted Boltzmann Machines (RBMs), DBN, Autoencoder (AE), and DRL [58]. Selvin et. al [59] explores the use of LSTM, RNN and CNN-sliding window models to predict future stock prices on a short-term basis. Similarly, Nikou et. al. [60] proposed a DNN model based on LSTM for stock price forecasting, achieving higher prediction accuracy than other machine learning methods.

5.3 Transformer-based approaches

The transformer architecture, which was originally developed for natural language processing tasks, has demonstrated high potential for stock price prediction, as it is particularly well-suited for processing sequential data, such as time series. This is achieved through the use of self-attention mechanisms, which allow the model to selectively weight different input elements and consider their dependencies with respect to one another. In contrast, many other deep learning models, such as CNNs and RNNs, are limited in their ability to capture long-term dependencies because they rely on fixed-size kernels to process the data. This can make it difficult for these models to effectively capture complex dependencies in time-series data.

Using a transformer model for stock price prediction requires preparing the data in a way that is compatible with the model's input requirements. This typically involves converting stock price data into a sequence of numerical values and possibly also incorporating additional features (e.g., company financial statements, economic indicators) as input to the model [61]. During training, the model will learn patterns and relationships in the data and can then be used to make predictions about future stock prices. The accuracy of the predictions will depend on the quality of the training data and the suitability of the model for the specific forecasting task.

The application of transformer architecture for stock price prediction is a relatively novel area of study. Multiple models adapting the NLP transformers for time-series prediction were proposed. Li et al. [62] introduced a transformer model, which compared favourably with the state-of-the-art on both synthetic and real-world data. Zerveas et al. [63] introduced a transformer-based framework for learning multivariate time-series representation, drawing inspiration from the BERT model. Another transformer network was presented for dynamic spatiotemporal forecasting [64]. Recently, an Informer model was developed for long sequence time-series forecasting [65]. Several transformer-based models have been applied to stock price prediction in recent years, all demonstrating an increase in accuracy compared to other models [66, 67, 68].

Chapter 6

The proposed model

This section provides a description of the proposed model for stock price prediction using sentiment analysis of media articles. Having discussed the basic theory behind sentiment extraction and time series forecasting, I will focus on the model's architecture and implementation details. The model analyses stock prices and financial news titles to predict future stock prices for a given company. As such, it must be able to perform two tasks:

1. Extract and quantify general and company-specific market sentiment from financial news articles.
2. Use the produced sentiment features, historical share prices and derivative technical indicators in stock price forecasting for a given company.

Thus, the model consists of two steps: sentiment extraction and time-series forecasting. First step is performed using a FinBERT model. Second step requires utilization of an Informer-based transformer model [65]. The basics of the model architecture is illustrated in Figure 6.1 In the sections below I discuss the model's input and go into the details of model's architecture.

Figure 6.1 The basics of the model architecture

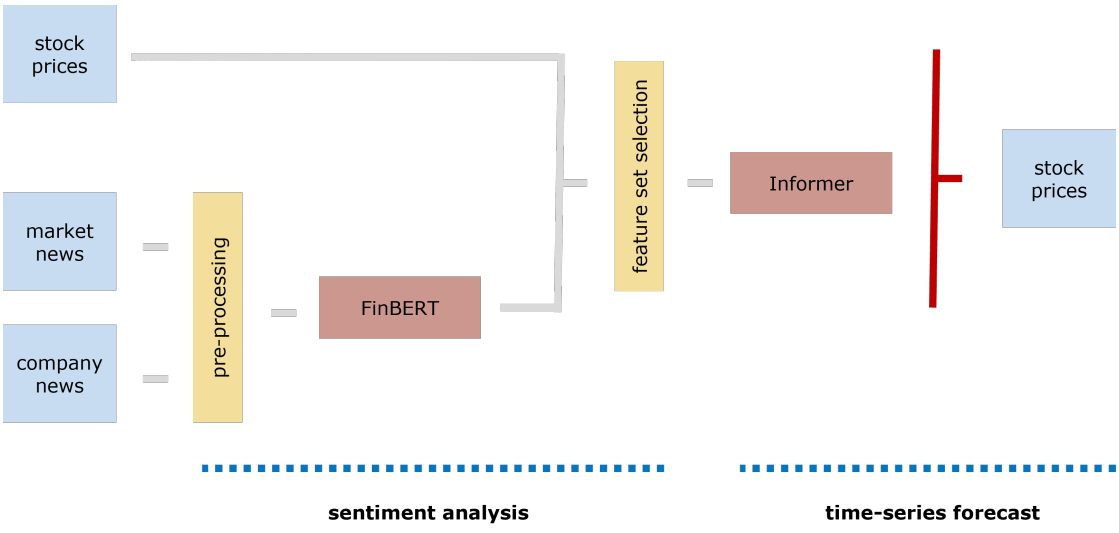


Table 6.1 General market news: exemplar instances

Date of issue	Headline	Company tickers
2022-10-29 12:47:11	Britain denies Russian claims that its navy personnel blew up Nord Stream gas pipeline	
2022-11-18 20:45:47	Vaccinating the unvaxxed is key to end pandemic: Former FDA official	PFE, JNJ, MRNA, NVAX, BNTX
2022-12-08 16:33:30	FOREX-Dollar eases against euro as investors weigh rates outlook	EUR=X, EURUSD=X

Table 6.2 Company-specific news: exemplar instances

Date of issue	Headline	Company tickers
2022-10-29 12:00:25	Meta's Problems Can Be Fixed. Don't Get Your Hopes Up That They Will.	META
2022-10-31 09:56:05	Apple's China supply chain tested as Covid hits iPhone assembler Foxconn	AAPL
2022-12-08 16:33:30	Down 36% From Its High, Is Alphabet Stock a Screaming Buy Right Now?	GOOGL

6.1 Model's input and data acquisition

To enable the model to accurately predict future stock price development for a selected company incorporating sentiment analysis I provide it with relevant textual information and historical data. This includes financial news and historical stock prices. Financial news are further divided into two categories: general market news, thought to be relevant to the stock market at large, and company-specific news, which are considered to only have impact on a given company. General market news may include reactions to macroeconomic activity, analysis of financial crises, reports on geopolitical developments, science and breaking news etc. Company-specific news, though, are focused on the selected company's performance, earning reports, public relation issues, top management changes etc. The model is capable of extracting the relevant sentiment information from the provided news articles and use it in the time-series prediction task.

With that said, I gather three datasets to serve as the model's input: *general market news* dataset, *company-specific news* dataset and *company stock prices* dataset. An instance of *general market news* and *company-specific news* datasets represents a single news article. Its features are a *headline*, *date of issue* and relation to a specific company - a *ticker*. I decide to use only a headline as an article's representation over a summary or a full content, as it is a standard practice among researchers due to headlines generally conveying the sentiment of a whole article [69]. In addition, a headline's length is more suitable for FinBERT model, used for sentiment extraction, as it is fine-tuned on a sentence-long textual data [50].

The features of a *company stock prices* dataset instance are a *timestamp*, a relevant *company ticker* and a *stock price* at the respective timestamp.

The exemplar instances of each dataset are displayed in Tables 6.1, 6.2, 6.3.

Table 6.3 Company stock prices: exemplar instances

Timestamp	Stock price	Company ticker
2022-10-28 20:00:04	99.20	META
2022-11-01 13:59:34	294.81	NFLX
2022-12-16 21:12:05	3852.36	^GSPC

The theory suggests that the impact of investor sentiment on stock prices is most pronounced in a short window of opportunity after the news release and before the market has a change to adjust to new information. To have a better chance at detecting this effect, I gather and examine the data on the intraday scale, which introduced challenges to the data availability and collection process. When tasked with gathering financial news data, there are two main options for researchers: either using an existing dataset, or compiling a brand new one through the web scrapping techniques. Financial news articles are often subject to copyright, therefore the publicly-available ready-made datasets are sparse [70, 71]. One example of such a dataset is Reuters-21578 text categorization collection data set [72], which is a collection of documents that appeared on Reuters newswire in 1987. The documents were assembled and indexed with categories including Business and Finance. This dataset is comprised of a large number of instances and have the benefit of spanning longer time frames. However, the data it contains is not dense enough to be useful in an intraday context. Similarly, the historical share prices are commonly provided on a day-to-day basic with no intraday price development dataset publicly available to my best knowledge. Therefore, a web scrapping technique had to be utilized to facilitate the model with the data required.

All three datasets were gathered using web scrapping from *Yahoo! Finance* website from October 28, 2022 till December 16, 2022 with a 15-minute interval. This follows the findings of previous studies, which emphasised the importance of informational noise reduction over the size of the news dataset [29, 73]. Therefore, a single news source or, in this case, an aggregator is commonly preferred. *General market news* dataset was scrapped from the *Latest Financial and Business News* page on the website, which provides such news categories as *World*, *U.S.*, *Politics*, *Technology*, *Business*, *Sports*, *Entertainment*, *Science*, etc. I consider the articles gathered from this page to be a representation of the information flow an average investor is exposed to on a daily basis, as the *Yahoo! Finance* website is an 8^{th} most popular media publisher in the U.S. with 245.3 million monthly visitors and *Latest Financial and Business News* page is among the most commonly visited [74]. The gathered articles intentionally include topics outside of the financial domain, as those were shown to influence investor sentiment as well, as described in Section 2.2. Thus, I assume the *general market news* dataset to be an adequate proxy for general market investor sentiment. Similarly, company-specific news articles as well as current stock prices are gathered from the company page news tab on the *Yahoo! Finance* website. As these news include information on the selected company developments, I consider them to serve as a suitable company-related investor sentiment proxy.

I choose six tickers META, AAPL, AMZN, NFLX, also referred to as FAANG, and ^GSPC (S&P500) traded on NYSE as a case study in this paper. The FAANG companies are among the largest traded on stock exchanges in the US and belong to the same industry. This allows to test whether the finding of the paper are consistent across the big-tech industry. I additionally conduct a comparative analysis with the market at large, represented by the S&P500 index. The selection of the FAANG companies and NYSE allows for the abundance of media information relevant to an individual company or the whole market at the desired granularity, enabling the study of shorter time intervals. The potential drawback of selecting companies traded on NYSE is a shorter window of opportunity to detect the media impact, due to a widely-reported higher market efficiency in developed economies. I attempt to analyse this effect by considering various prediction length, as described in Chapter 7.

6.2 Task 1: sentiment extraction

The gathered datasets serve as an input to the proposed model. In this section, I describe the process of extracting *sentiment labels* from the scrapped news headlines, serving as a proxy for investor opinion. These labels are then mapped to the company stock prices and aggregated to form *sentiment scores* of the final dataset used in the time-series forecasting task.

Table 6.4 News headlines sentiment scores

Title	Sentiment label
Strong Dollar Seen Hurting US Outlook and Even Tilting Fed Path	negative
China to speed up vaccinations, build more designated COVID hospitals	positive
Canada mints special black-ringed 'toonie' coin in memory of Queen Elizabeth	neutral

6.2.1 Sentiment classification

Before performing sentiment analysis, gathered news headlines are properly pre-processed. The following limitations are imposed:

- News headlines must consist of more than 30 symbols.
- News headlines must be unique, meaning that republishing of the same story is not allowed. However, the same headline can be used in relation to different companies.
- General market news headlines must not intersect with company-specific news headlines. If a headline is present in both datasets, it is removed from the general market news dataset. This results in a separate general market news dataset for each company.
- *Yahoo! Finance* publishes news titles with special terms included, such as "UPDATE 1-" etc. Such terms are irrelevant to the content of a news article and should be removed from the headline.

I decided to use the FinBERT model, described in Section4.5, for the purpose of analyzing the sentiment expressed in the news headlines. With BERT-like models holding the state-of-the-art in sentiment analysis tasks across their respective domains, the choice I faced was to either use a model already suitable for the financial domain, like FinBERT, or opt for a vanilla BERT model, which would need to be additionally fine-tuned. The latter option would require the preparation of task-specific data, meaning assigning sentiment labels to at least some part of the gathered financial news dataset. This was deemed not practical both due to the lack of necessary expertise to assess the financial sentiment of an article and required time resources. Some research was conducted in the area of automatic labelling techniques based on security's price development [75], but I determined it to by outside of the scope of this paper. Otherwise, an already labeled dataset relevant to the financial domain could be used for fine-tuning. To the best of my knowledge, the only publicly available labeled sentiment classification datasets in the financial domain are the Financial PhraseBank and FiQA, which are already used for the FinBERT pre-training. With this in mind, FinBERT model was selected for sentiment classification.

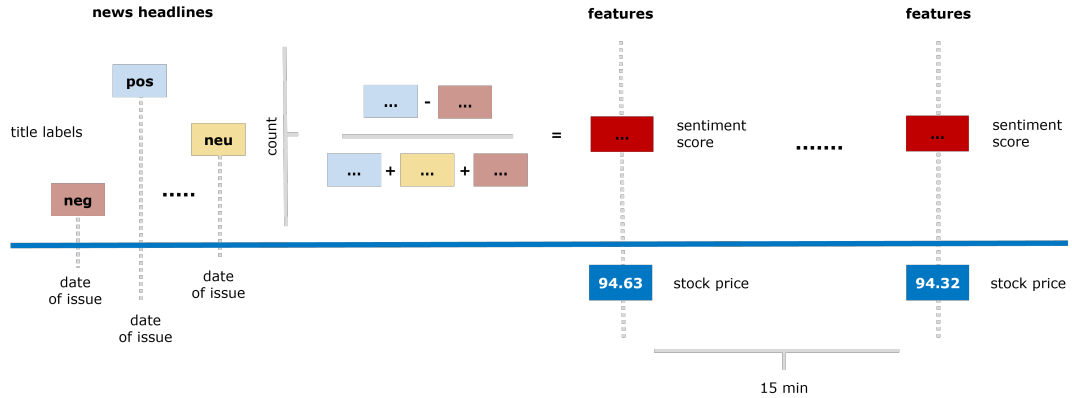
WordPiece algorithm is applied to transform pre-processed headlines into tokens acceptable by the FinBERT model. FinBERT processes the input tokens and classifies a given headline into a *negative*, *neutral* or *positive* sentiment category. This category is used to form the *sentiment label* feature for each news headline. An example of the sentiment labels produced can be seen in Table 6.4.

6.2.2 Sentiment scores

Future stock price prediction is achieved by analysing a time-series of historical stock prices. To study the effect of financial news sentiment, I formulate additional **sentiment scores**, representing the compound sentiment of the news articles in a given time interval. After assigning each news headline a sentiment label, the headlines are placed on the same timeline as company's stock prices via the date of issue and are aggregated based on which 15 minute time interval they fall into. The process is illustrated in Figure 6.2

Figure 6.2 Process of sentiment scores calculation

News headlines with dates of issue falling in a 15 minute interval before current stock price timestamp are aggregated and their sentiment labels are used to produce sentiment scores. Sentiment score is calculated using a total count of negative, neutral, and positive sentiment labels, as shown in Equation 6.1. The same procedure is applied to each 15 minute interval in stock prices dataset.



Several strategies can be utilized to measure the aggregated headlines' sentiment. Previous research primarily focuses on building sentiment features based on the count of news falling in each of the studied categories. The sentiment is commonly considered to be represented by the category with highest number of news in it [76]. Otherwise, a proportion of news in each class is thought to reflect sentiment [77, 78]. Similarly, I calculate sentiment score using the total count of aggregated negative, neutral, and positive sentiment labels, as a difference between the positive and negative news, divided by the total number of news, as shown below:

$$sentiment\ score = \frac{Pos_{count} - Neg_{count}}{Pos_{count} + Neu_{count} + Neg_{count}} \tag{6.1}$$

The strategy is applied to general market and company-specific news separately, forming two groups of sentiment scores. Additionally, to model the effect of continuous sentiment, I form further sentiment features by applying a moving average filter over the sentiment scores with periods 15 and 30. The periods were chosen to correspond with the prediction lengths, discussed in Chapter 7. All produced features and their labels are listed in Table 6.6. An example of the produced dataset's instances is provided in Table 7.5.

In an attempt to further improve the predictive capabilities of the model, the feature set is expanded to include technical indicators. Previous studies have demonstrated their beneficial impact on stock price prediction [61]. As such, additional technical features are computed: "returns", "moving average (period 15)" and "moving average (period 30)".

The analysis of the feature set and best features selection is discussed in Chapter 7.

6.3 Task 2: time-series forecasting

To perform the task of stock price prediction given the produced feature set, I utilize a transformer-based model for multivariate time-series forecasting. The architecture is inspired by Informer, which is a model adapting transformers for the time-series forecasting problem [65]. Below, I describe a modified version of original Informer, but I will be referring to its by the same name.

Table 6.5 An example of instances in the AAPL dataset

The illustrated dataset is used for stock price prediction in Task 2. To produce this dataset, data relevant to AAPL ticker was extracted from the stock prices, general market news and company-specific news dataset. Then the aggregation strategy was applied to the form sentiment scores.

date	**price**	**return**	**ma15**	**ma30**	**general score**	**company score**	...	**company score ma30**
2022-11-01 14:45:00	152.99	-1.26	153.56	153.46	0	0		-0.14
2022-11-01 15:00:00	150.74	-2.25	153.36	153.29	0	0		-0.13
2022-11-01 15:15:00	150.75	0.01	153.15	153.20	-0.11	-1		-0.13

Table 6.6 A list of produced features and their types

Feature label	**Feature**	**Feature type**
timestamp	timestamp	date
price	stock price	technical
return	return	
ma15	stock price ma period 15	
ma30	stock price ma period 30	
general score	general-market sentiment score	sentiment
company score	company-specific sentiment score	
general ma15	general-market sentiment score ma period 15	
company ma30	company-specific sentiment score ma period 30	

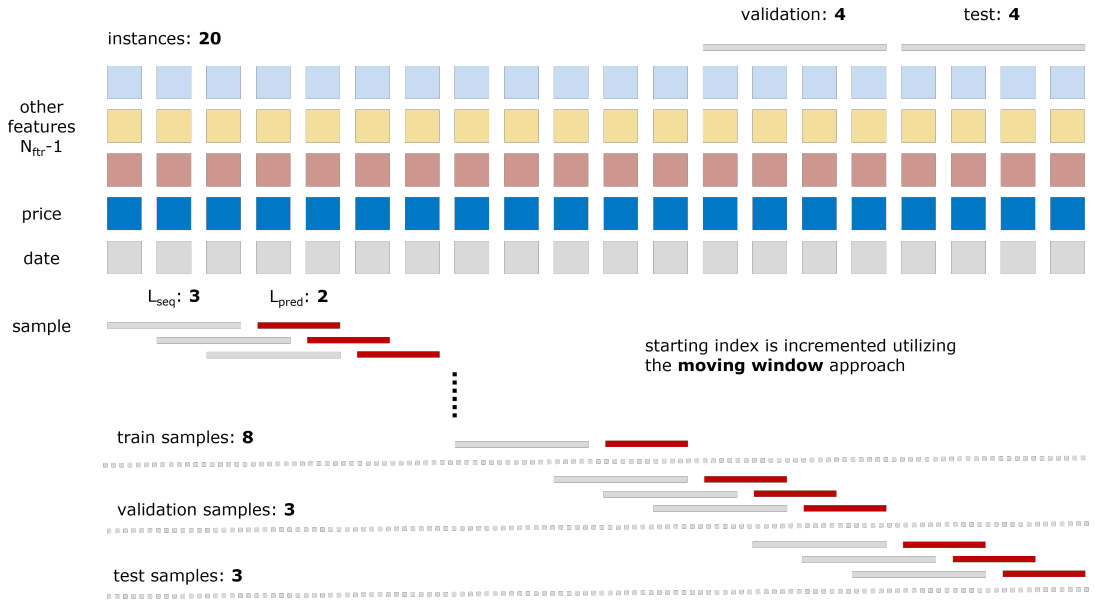
6.3.1 Informer input

The feature set produced in the previous section can be viewed as a multivariate time-series. The whole dataset is split into a training, validation and testing sets. The training set includes first 60% of the data and is used to train the model's parameters. Further 20% of the data is used as a validation set to evaluate the model during hyperparameter tuning and to mitigate over-fitting to the train data using an early stopping mechanism. Test set is comprised of the last 20% of input time-series and is used to evaluate the model.

Informer takes L_{seq} consequent instances with N_{ftr} number of features (date is not included) from the training set and predicts L_{pred} stock prices directly following it. Utilizing the moving window approach, the starting index of the input sequence is constantly incremented, until all the training data was seen by the model. The validation and testing procedures are carried out in the same way. The process is illustrated in Figure 6.3.

Figure 6.3 Data split and moving window

Provided a dataset of length 20 with $N_{\text{ftr}} = 4$ (date not included), a 16/4/4 train/validation/test split is performed. Setting $L_{\text{seq}} = 3$ and $L_{\text{pred}} = 2$ results in 8 train, 3 validation and 3 test samples.



A single instance consists of a timestamp (date) and N_{ftr} number of features, including *price*, which is considered to be a *target* variable. In order to be processable by the model, the inputs are first standardized by removing the mean and are scaled to unit variance. Then, inputs must be embedded into a vector space of dimension D_{model}. In a similar fashion to other transformer-based models, this includes value and positional embeddings with an addition of date-time embedding.

6.3.1.1 Value embedding

The *value embedding* is a single dimension convolution with kernel size 3 and *leaky relu* as an activation function. The weights are initialized using a normal distribution in accordance with Kaiming He initialization. [79] With input representation for i-th sequence sample $\mathcal{X}^i \in \mathbb{R}^{(L_{\text{seq}}+L_{\text{pred}})\times N_{\text{ftr}}}$, value embedding is:

$$\mathbf{val}(\mathcal{X}^i) = \mathbf{conv1d}(\mathcal{X}^i) \tag{6.2}$$

6.3.1.2 Positional embedding

A simple *positional learned embedding* is used to represent the relative positions of data points in a sequence. The use of positional embeddings in favor of positional encoding was made due to their generally better performance, as well as a stable lengths of inputs given by parameters L_{seq} and L_{pred} [43]. Therefore, it is guaranteed that the model has learned all possible positional embeddings during the training phase, unlike NLP transformers, where input length is variable.

$$\mathbf{pos}(\mathcal{X}^i) = \mathbf{embed}(\mathcal{X}^i) \tag{6.3}$$

6.3.1.3 Date-time embedding

Another piece of information which must be embedded into an input representation vector space is the timestamp. Date-time embeddings face a highly similar problem to that of positional embedding of variable length sequences, as the model will inevitably encounter the timestamps outside of those in the training set. Taking that into consideration, the embedding must be able to learn and recognise the periodical nature of date-time. While various methods of timestamp encoding has recently emerged, one of the most promising approaches is the *Time2Vec* representation [80].

With the goals of achieving periodicity, invariance to time rescaling and model architecture independence, Time2Vec embeds a given scalar notion of time t into a vector of size $k+1$ as follows:

$$\mathbf{t2v}(t)[i] = \begin{cases} \omega_i t + \varphi_i, & \text{if } i = 0. \\ \mathcal{F}(\omega_i t + \varphi_i), & \text{if } 1 \leq i \leq k, \end{cases} \tag{6.4}$$

where $\mathbf{t2v}(t)[i]$ is the i^{th} element of $\mathbf{t2v}(t)$, $\mathcal{F}$ is a periodic activation function, ω_i and φ_i are learnable parameters. $\mathcal{F}$ is chosen to be the cosine function.

Time2Vec application implies the need to convert a timestamp into a D_{time}-feature representation. The model achieves this by decomposing date-time into 5 features: *day of year*, *day of month*, *day of week*, *hour of day* and *minute of hour*. Each of the features are scaled for their values to occupy a range from $-\frac{1}{2}$ to $\frac{1}{2}$. To provide an example, *day of month* feature is calculated as follows:

$$\textit{day of month (scaled)} = \frac{\textit{day of month} - 1}{30} - \frac{1}{2}, \tag{6.5}$$

with other features derived analogously. The collection of these features is then passed to the **t2v** function and thus projected to the $k+1 = D_{\text{model}}$ dimension representation.

6.3.1.4 Input representation

Input representation for i-th sequence sample $\mathcal{X}^i \in \mathbb{R}^{(L_{\text{seq}}+L_{\text{pred}})\times N_{\text{ftr}}}$ and respective input timestamps $\mathcal{X}^i_{\text{time}} \in \mathbb{R}^{(L_{\text{seq}}+L_{\text{pred}})\times D_{\text{time}}}$ can be thus composed as a vector addition of value, positional and timestamp embeddings:

$$\mathcal{X}^i_{en} = \mathbf{val}(\mathcal{X}^i) + \mathbf{pos}(\mathcal{X}^i) + \mathbf{t2v}(\mathcal{X}^i_{\text{time}}), \tag{6.6}$$

where $\mathcal{X}^i_{en} \in \mathbb{R}^{(L_{\text{seq}}+L_{\text{pred}})\times D_{\text{model}}}$. Based on the performance of the model it has been observed that different types of embeddings appear not to significantly interfere with numerical information of each other, seemingly occupying different, roughly orthogonal, subspaces in $\mathbb{R}^{D_{model}}$,

which is backed by other studies [63]. This condition is much easier to satisfy with higher model dimension.

6.3.2 Informer architecture

The modified Informer model follows the basic principles of transformer architecture. It is composed of N_{layer} sequential encoder layers, which are in essence a typical transformer block described in Section 4.3. Each encoder layer consists of a N_{heads}-head self-attention mechanism, followed by residual connections, layer normalization, a position-wise feed-forward neural network and another application of residual connections and layer normalization.

Original Informer [65] takes after the earliest introduced transformers [43] in incorporating encoder-decoder architecture. In such configuration, encoder is tasked with with representing the whole sequence while decoder unpacks it to the desired target sequence. Additionally, decoder can access input sequence in an autoregressive manner, referred to as *teacher forcing*. Further research into transformer architecture demonstrated that a simple stack of transformer blocks is sufficient enough to achieve state-of-the-art in sequential tasks. BERT model described in Section 4.4 as well as the GPT-2 transformer model have entirely dispensed with the encoder-decoder architecture [49, 81]. Similarly, the proposed model does not incorporate a decoder, which is sometimes referred to as an *encoder-only architecture*.

The encoder's input is a multivariate sequence of $L_{\text{input}} = L_{\text{seq}} + L_{\text{target}}$ length, meaning that encoder has access to the target sequence it is tasked to predict. The goal is to provide the encoder with the target timestamps, but prevent it from attending to the target features. To achieve this a zero-mask $\mathcal{X}_0 \in \mathbb{R}^{L_{\text{pred}} \times N_{\text{ftr}}}$ is applied over the target features, effectively preventing the encoder from accessing them. In addition to rejecting the dynamic decoding of the vanilla encoder-decoder architecture this allows the model to produce outputs in one forward procedure, which is highly beneficial to the model's performance [65].

The encoder's output is a sequence of L_{target} predicted stock prices, which is compared to the real values to evaluate the model. The evaluation metric MSE $= \frac{1}{n}\sum_{i=1}^{n}(y - \hat{y})^2$ is used to evaluate the results, following the original Informer architecture [65]. Informer's model architecture is shown in Figure 6.4

Figure 6.4 Informer's architecture and zero-mask

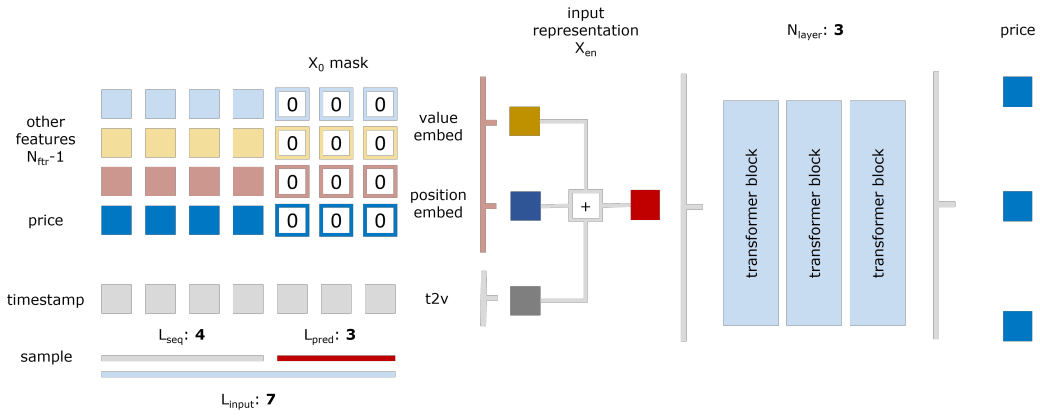


Table 6.7 Informer's parameters

Parameter's name	Description	Default value
root_path	root path of the input dataset	./data/stock
data_path	input dataset filename	AAPL.csv
features	forecasting task, options: [M, S, MS]	MS
ftr_num	number of features (not including timestamp)	10
d_out	dimensions of informer outputs	1
target	target feature in S or MS task	price
freq	frequency for time features encoding	15t
seq_len	input sequence length	30
pred_len	prediction sequence length	15
itr	number of iterations	10
train_epochs	number of train epochs	10
batch_size	batch size of train data	6
patience	early stopping patience	5
learning_rate	starting optimizer learning rate	0.0001
loss	loss function	mse
lradj	strategy of learning rate adjustment	type1
inverse	inverse output data	false
d_model	hidden dimension of the model	256
n_heads	number of attention heads	4
e_layers	number of encoder layers	4
d_ff	dimension of fead-forward neural network	1024
embed	time features embedding, options: [fixed, t2v]	t2v
activation	activation function	relu
padding	padding type	0
dropout	dropout rate	0.05
output_attention	output attention in encoder	false
predict	whether to predict unseen future data	false
num_workers	data loader num workers	0
use_gpu	whether to use gpu	true
gpu	gpu id	0
use_multi_gpu	whether to use multiple gpus	true
devices	device ids of multiple gpus	0

6.3.3 Prediction

As Informer model requires target timestamps to be passed as part of the model's inputs, predicting values outside of the available dataset implies the need for construction of a sequence of target timestamps. Considering the intraday nature of the studied datasets with 15 minutes frequency, this means correctly choosing the trading hours in the desired time interval. The model utilizes the NYSE trading calendar as well as the US national holidays calendar to accurately build the target timestamps. These timestamps with addition of zero-mask $\mathcal{X}_0$ are concatenated to the provided sequence $\mathcal{X}^i$ together forming the model's input. The output prices are then calculated with respect to the produced timestamps.

6.4 Implementation details

To facilitate the construction of the input datasets through the web scrapping technique, an open source Scrapy framework was used. Three Scrapy spiders were implemented to gather the data that form general news, company news and company prices datasets. A Unix bash job was scheduled to run on a created instance of Amazon Elastic Compute Cloud to scrap the data from the *Yahoo! Finance* website with 15 minutes intervals. An Amazon Simple Storage Service bucket was used to store the gathered data.

Python 3.9 and Jupyter Notebook were used to implement and present the model itself. The FinBERT model for news headlines sentiment analysis was accessed via the HuggingFace provided libraries, such as AutoModelForSequenceClassification, AutoTokenizer, AutoConfig, TextClassificationPipeline. The Informer model was implemented using the PyTorch library.

Chapter 7
Experiments

This chapter presents the application of the model described above on the gathered financial news and stock price data. I describe the datasets, explain the process of hyperparameter tuning and the hardware setup used for the experiments. Finally, I test whether the addition of sentiment features derived from the financial media improves the quality of the model's predictions by conducting a comparative analysis of various feature sets. The consistency of the findings is tested for several big-tech stocks as well as the S&P500 index, serving as a proxy for stock market in general.

7.1 Data analysis

As mentioned in Section 6.1, three datasets were gathered to facilitate the experiments: the general-market news dataset, the company-specific news dataset as well as the historical stock prices dataset. While the first one is a collection of financial news articles considered to have an impact on the stock market at large, the latter two are company related. Overall, six tickers were chosen: META, AAPL, GOOGL, NFLX, AMZN, ^GSPC. The data was gathered in the period from October 28, 2022 till December 16, 2022 with an interval of 15 minutes.

Table 7.1 shows the total number of instances in gathered datasets per ticker as well as the number of instances after pre-processing steps.

The development of the analysed ticker stock prices and train/validation/test split is presented in Figure 7.1.

In the sentiment extraction process, described in 6.2, each news headline is assigned a classification label. The distribution of these labels for general news as well as company-specific news is shown in Figure 7.2.

Table 7.1 Number of gathered data instances per ticker before and after pre-processing

	General news		Company news		Stock Price	
Ticker	**Total**	**After pre-processing**	**Total**	**After pre-processing**	**Total**	**After pre-processing**
META	13418	10741	601	511	889	873
AAPL	13418	10642	1057	876	897	878
GOOGL	13418	10743	664	592	903	881
NFLX	13418	10780	348	304	892	876
AMZN	13418	10717	1053	945	900	884
^GSPC	13418	10201	3392	2471	964	937

Figure 7.1 The development of the studied stock prices and a train/val/test split

Only business days and working hours are displayed to remove gaps in the plot

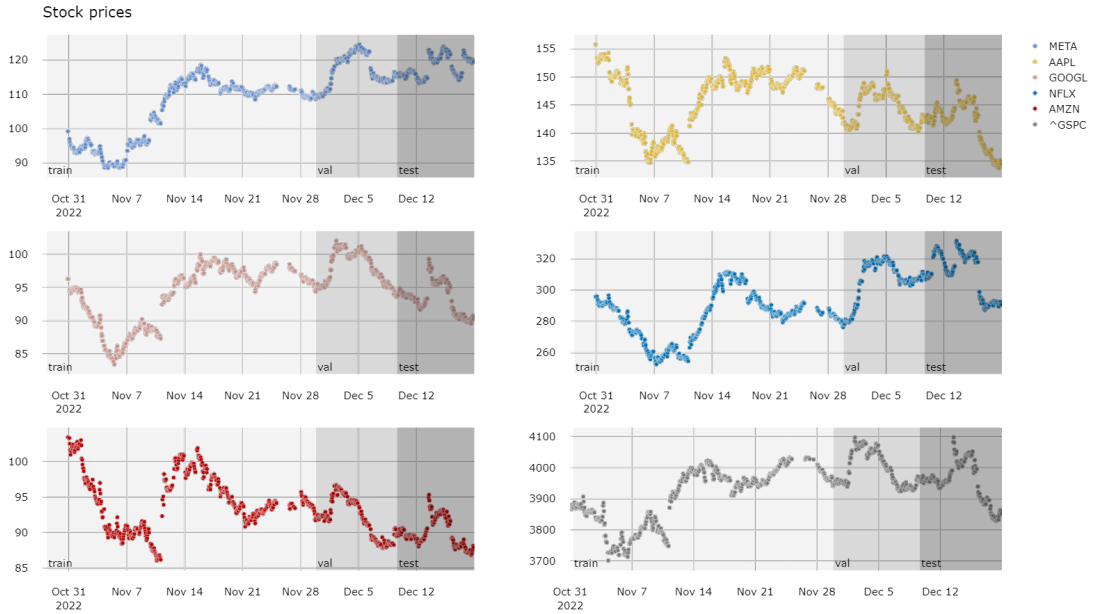


Table 7.2 A list of hyperparameters used for model tuning and the best-performing combination

Parameter	Range of values	Best value
batch_size	6, 12, 24	6
d_model	256, 512	256
n-heads	4, 6, 10	4
e_layers	4, 6, 10	4
activation	gelu, relu	relu

7.2 Hyperparameter tuning

In order to tune the model's parameters an exhaustive grid search was conducted over the parameter combinations presented in Table 7.2. For this purpose, I compile a dataset, which consists of the following features: *date*, *price*, *ma15*, *general score ma15*, *company score ma15*. The validation split of the dataset is used to evaluate the model during hyperparameter tuning. The model is optimized using the Adam algorithm. The starting learning rate is decaying two times smaller after every epoch. The total number of epochs is set to 10 with an application of early stopping with patience 5. The evaluation metric MSE $= \frac{1}{n}\sum_{i=1}^{n}(y-\hat{y})^2$ is used to evaluate the results.

Best model parameters are listed in Table 7.2.

Device specifications used for the model's training and all the experiments are listed in Table 7.3. The hyperparameter tuning procedure required around 4 hours to be completed.

Figure 7.2 Distribution of sentiment labels for general news and each ticker

The size of the ticker-specific graphs is proportional to the number of news headlines.

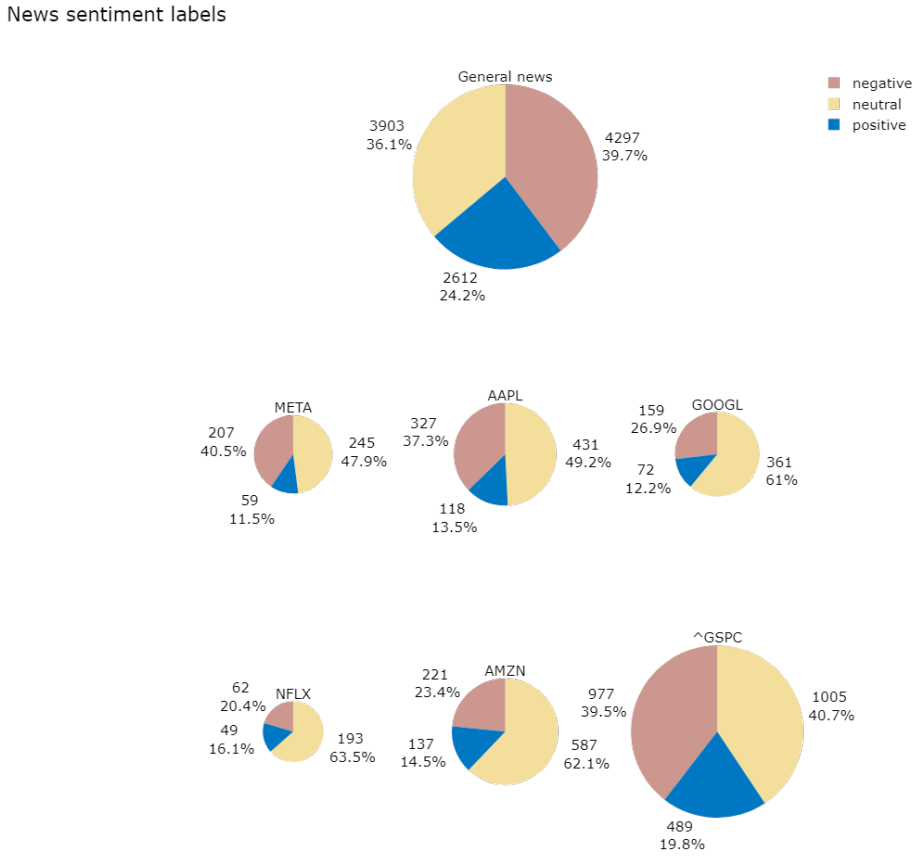


Table 7.3 Device specifications

Parameter	Value
Processor	11th Gen Intel(R) Core(TM) i7-11700F @ 2.50GHz 2.50 GHz
Installed RAM	16.0 GB (15.8 GB usable)
System type	64-bit operating system, x64-based processor
GPU 0	NVIDIA GeForce RTX 3060 Ti
DirectX version	12
GPU Memory	15.9 GB

7.3 Feature set analysis

To determine whether information derived from media articles has an impact on the stock market and whether it can be used to improve the stock price forecasting, I utilize the proposed model to perform news headlines sentiment analysis. The produced sentiment and technical features, listed in Table 6.6, are then grouped together to form different *feature sets*. These feature sets include price, technical indicators, sentiment scores or a combination of those. Instead of studying all possible feature sets that can be gathered via various combinations of available features, I select only a fraction of those due to the limited computational resources. The feature sets are formed in a way to allow for the study of media impact on stock price forecasting. This means that I evaluate and conduct comparative analysis of the performance of the model, when future prices are predicted based on:

- the current price only,
- the current price and technical indicators,
- the current price and sentiment scores,
- the current price and both technical and sentiment scores.

I will be referring to these features sets as *technical-based*, *sentiment-based* or *combined*. A full list of tested feature sets is provided in Table 7.4. It includes a label, which I will be using to identify the feature set, a list of features it consists of and a feature set type.

Additionally, I study the development of media impact in time by analysing 3 different prediction lengths. Finally, I perform the process for all 6 tickers mentioned to determine whether the findings are consistent among different companies in the same industry and with the market in general.

The model was trained on the datasets containing selected features with $L_{\text{seq}} = 30$, which roughly corresponds to stock price developments over a single day. This sequence length was chosen to enable training with relatively high prediction length, without sacrificing model's performance due to the lack of available samples. Three prediction lengths $L_{\text{pred}} = 15$, $L_{\text{pred}} = 30$ and $L_{\text{pred}} = 75$ were analysed to provide insight into the impact of various features over variable prediction lengths. The full results of the experiments are presented in Tables B.1, B.2, B.3, where each row corresponds to the selected feature set. The MSE metric was used to evaluate the model for each of the selected feature sets for each ticker. The *Average* column contains an average of min-max normalised MSE values among FAANG companies for a given feature set. The S&P500 was not included as the average MSE value is meant to represent the performance of a given feature set across the big-tech industry. Min-max normalisation is performed due to the fact that the MSE values for the different companies may be on different scales, and normalizing the values will ensure that they are comparable. The rows are sorted based on the value of *Average* column in ascending order. The *Best* column provides a number of FAANG companies for which the selected feature set was best performing. The top-5 best-performing feature sets for each prediction length are presented in Table 7.5. The total time required to train and evaluate the model to conduct the experiments is about 12 hours.

Table 7.4 A list of tested feature sets

Feature set label	Features	Feature set type
fs-price	price	technical-based
fs-return	price	
fs-ma15	price, ma15	
fs-ma30	price, ma30	
fs-general-score	price, general score	sentiment-based
fs-company-score	price, company score	
fs-general-company-score	price, general score, company score	
fs-general-ma15	price, general score ma15	
fs-company-ma15	price, company score ma15	
fs-general-company-ma15	price, general score ma15, company score ma15	
fs-general-ma30	price, general score ma30	
fs-company-ma30	price, company score ma30	
fs-general-company-ma30	price, general score ma30, company score ma30	
fs-return-general-company-score	price, return, general score, company score	combined
fs-ma15-general-company-ma15	price, ma15, general score ma15, company score ma15	
fs-ma30-general-company-ma30	price, ma30, general score ma30, company score ma30	

Table 7.5 Feature set analysis, top-5 results for each prediction length

Prediction length	Feature set	META	AAPL	GOOGL	NFLX	AMZN	Average	Best	^GSPC
15	fs-return-general-company-score	0.199	0.243	0.212	0.462	0.221	0.064	1	0.182
	fs-price	0.165	0.263	0.242	0.456	0.270	0.100	2	0.196
	fs-return	0.235	0.261	0.268	0.588	0.253	0.225	0	0.178
	fs-ma15	0.250	0.341	0.276	0.501	0.241	0.242	0	0.187
	fs-ma30	0.284	0.344	0.284	0.576	0.247	0.311	0	0.201
30	fs-price	0.265	0.686	0.459	0.648	0.343	0.061	0	0.357
	fs-return-general-company-score	0.261	0.399	0.414	0.912	0.340	0.067	3	0.254
	fs-return	0.307	0.503	0.461	0.825	0.374	0.116	0	0.310
	fs-ma15	0.367	0.670	0.516	0.635	0.334	0.151	2	0.296
	fs-general-score	0.525	0.839	0.456	0.726	0.503	0.271	0	0.513
75	fs-general-company-ma30	0.891	0.619	0.372	2.080	0.555	0.252	3	0.458
	fs-price	0.837	1.286	0.646	1.548	1.006	0.309	0	0.505
	fs-general-score	0.792	1.809	0.769	1.400	0.865	0.321	0	0.544
	fs-return-general-company-score	0.531	1.262	0.719	1.304	2.943	0.328	2	0.570
	fs-ma30-general-company-ma30	0.963	0.693	0.431	2.196	0.684	0.328	0	0.487

The shortest prediction window $L_{\text{pred}} = 15$ is arguably the most interesting one, as the impact of financial news sentiment is considered to be most prominent shortly after news release, as explained in 1.4. With stock market taking some time to adjust to new information, the addition of sentiment features is hypothesised to improve the prediction quality of the model. The results, indeed, indicate some improvement in the model's performance when sentiment features are added. For example, *fs-return-general-company-score* feature set is found to be best performing across FAANG companies with a lower average MSE score than *fs-return*, as shown in 7.5. However, when comparing the error scores of feature sets based on *ma15* and *ma30* technical indicators, it becomes apparent that the addition of sentiment features worsens the results. Overall, for $L_{\text{pred}} = \frac{1}{2}L_{\text{seq}}$ it is evident that technical-based feature sets achieve a better performance than sentiment-based ones, occupying the top-5 results in the table. Interestingly, *fs-price* acquired the lowest MSE score among all technical-based feature sets, supporting the claims of the EMH about highly efficient information absorption in the developed markets. Feature sets including *return* indicator appear to possess higher prediction capabilities, which might be explained by the repeating patterns in day-to-day return developments. Sentiment feature sets based on company news seem to outperform the general news based ones, and feature sets containing both seeing higher MSE values. Sentiment scores demonstrated comparable performance with their smoothed counterparts, e.g. *fs-company-score* vs. *fs-company-ma15*. Overall, incorporating media sentiment in predicting short-term price developments is found to bring insignificant improvements to stock price forecasting if any at all. It is worth mentioning, thought, that the difference in error scores for the best performing feature sets is too marginal to be considered significant, preventing me from drawing definitive conclusions from the results. I discuss the per-ticker results further below.

When considering a prediction window $L_{\text{pred}} = L_{\text{seq}} = 30$, the industry-average results are similar, refer to Table 7.5. The top-6 is occupied by the same feature sets with *fs-price* being best performing, with, again, minor differences in average MSE score. Sentiment-based feature sets tend to be outperformed by the technical-based or combined.

Interestingly, in case of a longer prediction window $L_{\text{pred}} = \frac{5}{2}L_{\text{seq}} = 75$, the prediction ability of technical-based feature sets tends to fall behind sentiment-based ones, as shown in Table B.3. For example, *fs-ma15* has a higher average MSE score than *fs-general-ma15*, or *fs-ma30* – higher than *fs-general-ma30* The *fs-general-company-ma30* feature set is found to be best performing, achieving best results for AAPL, GOOGL, and AMZN tickers. These results differ from the findings for shorter prediction lengths and point to the model's ability to extract market trends from news sentiment, which can be beneficial for a longer-term prediction. It could also be explained by the potential time-lag required for conscious investors to process the article, form their sentiment, and act on it. Nevertheless, *fs-price* feature set still demonstrates one of the lowest MSE scores.

I further analyse the performance of each feature set per specific company. The per-ticker average MSE value is calculated by taking the average of the normalised error scores for each prediction length. I list the top-5 best performing feature sets per each company with respective error scores in Table 7.6. The complete per-ticker results are provided in Table B.4. Similarly to per prediction length results, an *Average* column is added, which is calculated as an average normalised MSE score of the FAANG tickers, representing the performance of a feature set over all prediction lengths and big-tech tickers. The rows are sorted in ascending order based on this value. *Best* column shows the number of FAANG ticker, for which selected feature set is the best performing.

Generally, it is apparent there is little consistency among the produced results: for each FAANG company, there is a different selection of best performing feature sets. One similarity most of the FAANG tickers share: technical-based indicators, such as *fs-price* and *fs-return*, tend to perform well, with the exception of AMZN, where combined and sentiment-based feature sets occupy the top-5 spots. In particular, the *fs-price* and *fs-return-general-company-score* feature sets are among the top-5 best performing for META, AAPL, GOOGL and NFLX. When it comes

to sentiment-based feature sets, META is the only ticker demonstrating lower MSE scores for company-specific news, when compared to general-market or a combination of both. For all companies, other then NFLX, smoothing the sentiment scores with a moving average filter has led to improved performance.

ˆGSPC ticker was chosen to serve as a baseline representing the stock market at large. When comparing the FAANG companies' results to the ones for the ˆGSPC, one thing becomes evident. The sentiment-based feature sets demonstrate a higher average MSE score in predicting ˆGSPC stock when compared to technical-based feature sets or combined. These results could be explained by the investor sentiment impact being less pronounced when applied to indices compared to individual companies. This outcome is also backed by the notion that developed markets are generally more efficient in information absorption.

The performed experiments lead me to conclude the following:

- Sentiment-based feature sets showed generally weaker performance than price or technical-based feature sets for $L_{\text{pred}} = \frac{1}{2}L_{\text{seq}} = 15$ and $L_{\text{pred}} = L_{\text{seq}} = 30$ prediction lengths. When considering a longer prediction window of $L_{\text{pred}} = \frac{5}{2}L_{\text{seq}} = 75$, sentiment-based features, to the contrary, demonstrated better performance.
- I explain higher average short-term predictive power of price only and technical-based feature sets with widely reported high information efficiency of the developed stock markets.
- I attribute lowest average MSE scores of sentiment-based feature sets for longer-term prediction with the model's ability to extract information about future stock price trends from investor sentiment.
- The results were found to be inconsistent across FAANG companies. The differences in error scores among best performing feature sets are to marginal to be considered significant. Both technical and sentiment-based feature sets are among top 5 for each of the FAANG companies.
- The complementary nature of technical and sentiment-based indicators was not generally established. While *fs-return-general-company-score* on average performs better than *fs-return*, as can be seen in Table B.4, the same is not true for other combined feature sets.
- Smoothing the sentiment scores with a moving average filter generally improved the results for most of the FAANG companies. Judging from the average MSE scores for all prediction lengths, presented in Table B.4, moving average sentiment scores with period 15 yielded better results, supporting the claim made in Section 1.4
- I did not manage to determine the complementary nature of company and general market news sentiment, as evident from Tables 7.6, B.4, with both showing comparable predictive power.
- General market represented by the S&P500 index was shown to be highly information efficient on average across all analysed prediction lengths.

Overall, though some feature sets that included financial news sentiment showed high predictive power, especially over longer prediction window, I cannot definitively conclude that media sentiment improves stock price prediction on the intraday scale. This outcome goes along with the EMH stating that developed markets are efficient at absorbing information. A better performance of sentiment analysis for long-term prediction demonstrates that financial news could be considered a source of some fundamental information, the extraction of which has the potential to improve long-term prediction. This contradicts the approach of technical analysis, relying solely on information incorporated in past prices. This outcome is supported by other researches, who studied investor sentiment derived from social media on a day-to-day basic and did not found significant improvement in model's predictive capabilities [31, 75]. However, the topic remains actively studied and the research community has not yet reached a consensus, so future work is advised.

Table 7.6 Top-5 best performing feature sets per ticker

Ticker	Feature set	MSE score
META	fs-return-general-company-score	0.048
	fs-return	0.154
	fs-company-ma30	0.201
	fs-price	0.222
	fs-company-ma15	0.279
AAPL	fs-general-ma15	0.051
	fs-return-general-company-score	0.080
	fs-return	0.117
	fs-price	0.144
	fs-general-ma30	0.158
GOOGL	fs-return-general-company-score	0.158
	fs-price	0.222
	fs-general-company-ma15	0.276
	fs-ma30-general-company-ma30	0.320
	fs-company-ma30	0.346
NFLX	fs-price	0.056
	fs-general-score	0.090
	fs-ma15	0.094
	fs-return-general-company-score	0.115
	fs-ma30	0.174
AMZN	fs-ma30-general-company-ma30	0.049
	fs-general-ma30	0.098
	fs-general-ma15	0.101
	fs-general-company-ma15	0.101
	fs-ma15-general-company-ma15	0.109
^GSPC	fs-return	0.162
	fs-return-general-company-score	0.187
	fs-price	0.204
	fs-ma15	0.273
	fs-ma30	0.303

Chapter 8
Conclusions

In conclusion of this paper, I will review the goals and evaluate the degree of their fulfilment. I will mention the results achieved in Chapter 7, list the major hurdles I faced in working on this thesis, and suggest areas of future research.

8.1 Achieved goals

The main objective of the thesis was to propose a model for stock price development prediction utilizing media sentiment analysis and to determine whether media information could be used to improve the quality of the prediction. This objective was achieved successfully. I managed to propose and implement a model that performs sentiment classification of financial news headlines and uses it in stock price forecasts. The experiments I conducted were aimed at determining whether the extracted sentiment information aided in achieving higher prediction quality with various prediction lengths considered.

To facilitate the development of the model with the necessary theoretical background, various views on market predictability were discussed. I defined the concept of investor sentiment, looked into its potential proxies, and discussed the role of various media in its formation. I reviewed the methods for sentiment analysis and time-series prediction with a focus on the transformer architecture and described the studies utilizing these models in stock price prediction.

I faced challenges during the data acquisition process, as no publicly-available media and stock prices database was deemed sufficient to enable the study of media effect on stock price prediction on the intraday scale. Therefore, I implemented a software tool allowing for automatic web scrapping of the data required by the model.

The outcome of the experiments demonstrate that while news media sentiment analysis shoes better performance for long-term prediction across all FAANG tickers, incorporating it for a short-term stock price forecast does not yield any improvement in the results.

8.2 Suggestion for future research

One of the biggest challenges of research in stock market predictability is the lack of publicly available data, especially on the intraday scale. In this thesis I opted to gather the required datasets using web scrapping over a relatively short time interval, which imposed limitations on the length of studied sequences. I suggest the findings to be tested on a larger set of data, allowing for a wider range of sequence/prediction lengths, aiding in determining the impact of media sentiment more precisely. Additionally, it would be beneficial to analyse stock markets in

developing economies, as media sentiment effect is commonly reported to be more pronounces there.

The need to analyse a larger set of data inevitably leads to an increase in required computational resources. With transformer architecture this problem is most apparent, as the number of computations required in standard dot-product attention growth exponentially with the increase in processed sequences. Thus, to enable the analysis of longer prediction windows, an alternative to the vanilla attention mechanism should be used. Most recent attempts at adapting transformers for time-series forecasting propose such alternatives, drastically reducing the memory and time complexity of the proposed models [65, 82]. This could be highly beneficial for detecting the impact of investor sentiment on stock prices, as it would allow the study of continuous prediction length intervals, potentially leading to discovering the precise time interval where the sentiment effect is most prominent. Future studies could focus on applying and evaluating these models in stock price prediction.

Lastly, a more fine-grained approach to sentiment extraction is advised. With most research dedicated to document or sentence-level sentiment classification, analyzing media sentiment on an aspect level is neglected. With recent advances in such areas of NLP as text summarization and comprehension, brought by the transformer networks, I expect more complex approaches to sentiment analysis to emerge.

Appendix A

Usage instructions

A.1 Setting up the environment

Before running the Scrapy spiders or Informer model, a proper `conda` environment must be created. To set up the environment and activate, enter the `src` directory and run the following commands:

- `conda env create -f environment.yml`
- `conda activate bp`

A.2 Running Scrapy spiders

To run a Scrapy spider, enter the `src/bp-scrapper` directory and run one of the following commands:

- `scrapy crawl news_sp`
  `-o <path to output_news.jl file>`
  `-s JOBDIR=<path to job state directory>`
- `scrapy crawl ticker_news_sp`
  `-o <path to output_ticker_news.jl file>`
  `-s JOBDIR=<path to job state directory>`
- `scrapy crawl price_sp`
  `-o <path to output_price.jl file>`
  `-s JOBDIR=<path to job state directory>`

Important: the spiders are set up to parse the US version of *Yahoo! Finance* website, so ensure that you run them from the US IP address or using a VPN.

A.3 Informer showcase

The Informer model source code is located in the `src/bp-informer` directory. The proposed model as described in Chapter 6, in particular data pre-processing, financial news headlines sentiment analysis, sentiment score calculation, the composition of sentiment features and feature sets, Informer hyperparameter tuning, and the conduction of the experiments are showcased in

the `data-processor.ipynb` Jupyter notebook. You may also refer to it for an example of Informer usage.

`run_informer.ipynb` is a file providing an API for Informer's training and testing. Run the `main` function to train and test the model using the `default_args`. For the description of the arguments please refer to Table 6.7. To test the model with a different configuration consider editing `default_args`.

The results of the experiments described in this paper can be found in the `bp-informer/results` directory containing the saved results of the experiments performed for 15, 30, and 75 prediction lengths in CSV format.

Appendix B

Experiment results

Here I provide the full experiment results data. The following tables are listed:

- Table B.1 showcases model results for $L_{\text{seq}} = 30$ and $L_{\text{pred}} = 15$
- Table B.2 showcases model results for $L_{\text{seq}} = 30$ and $L_{\text{pred}} = 30$
- Table B.3 showcases model results for $L_{\text{seq}} = 30$ and $L_{\text{pred}} = 75$
- Table B.4 showcases model average model results across all tested prediction lengths.

Each row of the mentioned tables represents one of the tested feature sets. Further I provide column description:

- The *Feature set* column contains the label of the feature set.
- Columns *META*, *AAPL*, *GOOGL*, *NFLX*, *AMZN* and *^GSPC* contain the MSE score of the model for a given ticker. In case of Table B.4 the value represent an average of min-max normalized values across all prediction lengths for a respective ticker.
- The *Average* column is an average of the min-max normalized MSE scores across all FAANG tickers. The ^GSPC MSE score is not included in the calculation.
- The *Best* column contains a number of tickers, for which the respective feature set had the lowest MSE score.

Table B.1 Feature set analysis, prediction length 15

Prediction length	Feature set	META	AAPL	GOOGL	NFLX	AMZN	Average	Best	^GSPC
15	fs-return-general-company-score	0.199	0.243	0.212	0.462	0.221	0.064	1	0.182
	fs-price	0.165	0.263	0.242	0.456	0.270	0.100	2	0.196
	fs-return	0.235	0.261	0.268	0.588	0.253	0.225	0	0.178
	fs-ma15	0.250	0.341	0.276	0.501	0.241	0.242	0	0.187
	fs-ma30	0.284	0.344	0.284	0.576	0.247	0.311	0	0.201
	fs-general-score	0.336	0.329	0.307	0.512	0.271	0.363	0	0.279
	fs-company-ma15	0.223	0.413	0.288	0.662	0.342	0.375	0	0.420
	fs-company-score	0.271	0.313	0.341	0.692	0.284	0.410	0	0.231
	fs-company-ma30	0.227	0.382	0.214	0.686	0.558	0.410	0	0.320
	fs-ma15-general-company-ma15	0.297	0.488	0.317	0.777	0.210	0.471	0	0.241
	fs-general-company-ma15	0.324	0.512	0.250	0.956	0.228	0.505	0	0.322
	fs-general-company-score	0.268	0.477	0.321	0.597	0.452	0.511	0	0.266
	fs-general-ma15	0.389	0.209	0.397	0.748	0.250	0.529	1	0.297
	fs-ma30-general-company-ma30	0.333	0.686	0.280	0.811	0.187	0.544	1	0.271
	fs-general-ma30	0.401	0.312	0.346	0.805	0.277	0.563	0	0.259
	fs-general-company-ma30	0.310	0.419	0.336	1.010	0.373	0.645	0	0.271

Table B.2 Feature set analysis, prediction length 30

Prediction length	Feature set	META	AAPL	GOOGL	NFLX	AMZN	Average	Best	^GSPC
30	fs-price	0.265	0.686	0.459	0.648	0.343	0.061	0	0.357
	fs-return-general-company-score	0.261	0.399	0.414	0.912	0.340	0.067	3	0.254
	fs-return	0.307	0.503	0.461	0.825	0.374	0.116	0	0.310
	fs-ma15	0.367	0.670	0.516	0.635	0.334	0.151	2	0.296
	fs-general-score	0.525	0.839	0.456	0.726	0.503	0.271	0	0.513
	fs-ma30	0.390	0.806	0.608	0.725	0.514	0.274	0	0.314
	fs-general-company-score	0.487	1.092	0.506	0.900	0.645	0.360	0	0.473
	fs-company-ma15	0.340	1.083	0.671	1.203	0.417	0.411	0	0.627
	fs-general-ma15	0.588	0.499	0.675	0.946	0.434	0.447	0	0.536
	fs-ma15-general-company-ma15	0.486	1.276	0.615	1.059	0.597	0.474	0	0.497
	fs-company-score	0.477	1.228	0.712	0.968	0.508	0.487	0	0.473
	fs-general-company-ma15	0.462	1.253	0.581	1.465	0.514	0.525	0	0.581
	fs-ma30-general-company-ma30	0.543	1.663	0.592	1.155	0.494	0.545	0	0.473
	fs-general-ma30	0.571	0.893	0.762	1.086	0.411	0.557	0	0.531
	fs-general-company-ma30	0.510	1.200	0.645	1.312	0.678	0.568	0	0.556
	fs-company-ma30	0.355	2.392	0.638	0.805	2.067	0.627	0	0.493

Table B.3 Feature set analysis, prediction length 75

Prediction length	Feature set	META	AAPL	GOOGL	NFLX	AMZN	Average	Best	^GSPC
75	fs-general-company-ma30	0.891	0.619	0.372	2.080	0.555	0.252	3	0.458
	fs-price	0.837	1.286	0.646	1.548	1.006	0.309	0	0.505
	fs-general-score	0.792	1.809	0.769	1.400	0.865	0.321	0	0.544
	fs-return-general-company-score	0.531	1.262	0.719	1.304	2.943	0.328	2	0.570
	fs-ma30-general-company-ma30	0.963	0.693	0.431	2.196	0.684	0.328	0	0.487
	fs-return	0.543	1.345	0.896	2.176	1.223	0.352	0	0.522
	fs-general-ma15	0.931	1.009	0.484	2.265	0.737	0.359	0	0.500
	fs-ma15	0.828	1.340	0.677	1.622	1.464	0.364	0	0.599
	fs-general-ma30	0.915	0.659	0.476	2.677	0.572	0.369	0	0.540
	fs-ma30	0.829	1.941	0.776	1.619	1.195	0.400	0	0.595
	fs-general-company-score	0.998	1.213	0.644	1.607	1.290	0.405	0	0.675
	fs-general-company-ma15	0.839	1.144	0.477	2.892	0.770	0.406	0	0.514
	fs-company-ma15	0.694	2.143	0.750	2.076	1.479	0.427	0	0.471
	fs-company-ma30	0.556	4.452	0.652	1.849	1.835	0.463	0	0.445
	fs-ma15-general-company-ma15	0.960	1.026	0.604	2.712	0.826	0.468	0	0.464
	fs-company-score	0.877	1.461	1.103	1.555	2.393	0.578	0	0.526

Table B.4 Feature set analysis, average MSE values

Feature set	META	AAPL	GOOGL	NFLX	AMZN	Average	Best	^GSPC
fs-return-general-company-score	0.048	0.080	0.158	0.115	0.365	0.100	2	0.187
fs-price	0.222	0.144	0.222	0.056	0.139	0.106	1	0.204
fs-return	0.154	0.117	0.385	0.339	0.160	0.197	0	0.162
fs-ma15	0.440	0.200	0.352	0.094	0.175	0.225	0	0.273
fs-general-score	0.697	0.261	0.392	0.090	0.151	0.304	0	0.514
fs-ma30	0.512	0.277	0.500	0.174	0.178	0.323	0	0.303
fs-company-ma15	0.279	0.389	0.555	0.514	0.284	0.418	0	0.704
fs-general-company-score	0.709	0.355	0.409	0.255	0.401	0.436	0	0.650
fs-general-ma15	0.935	0.051	0.634	0.502	0.101	0.448	1	0.496
fs-general-company-ma15	0.649	0.400	0.276	0.968	0.101	0.478	0	0.591
fs-ma30-general-company-ma30	0.833	0.551	0.320	0.610	0.049	0.481	1	0.385
fs-ma15-general-company-ma15	0.722	0.377	0.488	0.659	0.109	0.483	0	0.331
fs-general-company-ma30	0.716	0.281	0.445	0.768	0.233	0.498	0	0.417
fs-general-ma30	0.923	0.158	0.622	0.679	0.098	0.510	0	0.497
fs-company-score	0.617	0.285	0.851	0.328	0.377	0.534	0	0.386
fs-company-ma30	0.201	0.788	0.346	0.321	0.845	0.547	0	0.409

Bibliography

1. DRAKOPOULOU, Veliota. A review of fundamental and technical stock analysis techniques. *Journal of Stock & Forex Trading*. 2016, vol. 5.
2. LEE, Charles M.C. Market efficiency and accounting research: a discussion of 'capital market research in accounting' by S.P. Kothari. *Journal of Accounting and Economics*. 2001, vol. 31, no. 1, pp. 233–253. ISSN 0165-4101. Available from DOI: https://doi.org/10.1016/S0165-4101(01)00038-6.
3. FAMA, Eugene F. Random Walks in Stock Market Prices. *Financial Analysts Journal*. 1965, vol. 21, no. 5, pp. 55–59. Available from DOI: 10.2469/faj.v21.n5.55.
4. FAMA, Eugene F. Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance* [online]. 1970, vol. 25, no. 2, pp. 383–417 [visited on 2022-11-13]. ISSN 00221082, ISSN 15406261. Available from: http://www.jstor.org/stable/2325486.
5. NASEER, Mehwish; BIN TARIQ, Dr, et al. The efficient market hypothesis: A critical review of the literature. *The IUP Journal of Financial Risk Management*. 2015, vol. 12, no. 4, pp. 48–63.
6. DANIEL, Kent; HIRSHLEIFER, David; TEOH, Siew Hong. Investor psychology in capital markets: evidence and policy implications. *Journal of Monetary Economics*. 2002, vol. 49, no. 1, pp. 139–209. ISSN 0304-3932. Available from DOI: https://doi.org/10.1016/S0304-3932(01)00091-5.
7. MALKIEL, Burton G. The Efficient Market Hypothesis and Its Critics. *Journal of Economic Perspectives*. 2003, vol. 17, no. 1, pp. 59–82. Available from DOI: 10.1257/089533003321164958.
8. LO, Andrew W. Reconciling efficient markets with behavioral finance: the adaptive markets hypothesis. *Journal of investment consulting*. 2005, vol. 7, no. 2, pp. 21–44.
9. URQUHART, Andrew; MCGROARTY, Frank. Are stock markets really efficient? Evidence of the adaptive market hypothesis. *International Review of Financial Analysis*. 2016, vol. 47, pp. 39–49. ISSN 1057-5219. Available from DOI: https://doi.org/10.1016/j.irfa.2016.06.011.
10. APPIAH-KUSI, Joe; MENYAH, Kojo. Return predictability in African stock markets. *Review of Financial Economics*. 2003, vol. 12, no. 3, pp. 247–270. ISSN 1058-3300. Available from DOI: https://doi.org/10.1016/S1058-3300(02)00073-3.
11. YU, Hao; NARTEA, Gilbert V.; GAN, Christopher; YAO, Lee J. Predictive ability and profitability of simple technical trading rules: Recent evidence from Southeast Asian stock markets. *International Review of Economics & Finance*. 2013, vol. 25, pp. 356–371. ISSN 1059-0560. Available from DOI: https://doi.org/10.1016/j.iref.2012.07.016.
12. ACHELIS, Steven B. *Technical Analysis from A to Z*. McGraw Hill New York, 2001.

13. CHONG, Terence Tai-Leung; NG, Wing-Kam. Technical analysis and the London stock exchange: testing the MACD and RSI rules using the FT30. *Applied Economics Letters*. 2008, vol. 15, no. 14, pp. 1111–1114.

14. ARIYO, Adebiyi A.; ADEWUMI, Adewumi O.; AYO, Charles K. Stock Price Prediction Using the ARIMA Model. In: *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*. 2014, pp. 106–112. Available from DOI: 10.1109/UKSim.2014.67.

15. ABARBANELL, Jeffery S.; BUSHEE, Brian J. Abnormal Returns to a Fundamental Analysis Strategy. *The Accounting Review* [online]. 1998, vol. 73, no. 1, pp. 19–45 [visited on 2022-11-15]. ISSN 00014826. Available from: http://www.jstor.org/stable/248340.

16. KHADJEH NASSIRTOUSSI, Arman; AGHABOZORGI, Saeed; YING WAH, Teh; NGO, David Chek Ling. Text mining for market prediction: A systematic review. *Expert Systems with Applications*. 2014, vol. 41, no. 16, pp. 7653–7670. ISSN 0957-4174. Available from DOI: https://doi.org/10.1016/j.eswa.2014.06.009.

17. THOMSETT, Michael C. *Getting started in fundamental analysis*. John Wiley & Sons, 2006.

18. DECHOW, Patricia M; HUTTON, Amy P; MEULBROEK, Lisa; SLOAN, Richard G. Short-sellers, fundamental analysis, and stock returns. *Journal of Financial Economics*. 2001, vol. 61, no. 1, pp. 77–106. ISSN 0304-405X. Available from DOI: https://doi.org/10.1016/S0304-405X(01)00056-3.

19. KEYNES, John Maynard. The general theory of employment. *The quarterly journal of economics*. 1937, vol. 51, no. 2, pp. 209–223.

20. BAKER, Malcolm; WURGLER, Jeffrey. Investor sentiment in the stock market. *Journal of economic perspectives*. 2007, vol. 21, no. 2, pp. 129–152.

21. BROWN, Gregory W; CLIFF, Michael T. Investor sentiment and asset valuation. *The Journal of Business*. 2005, vol. 78, no. 2, pp. 405–440.

22. LEMMON, Michael; PORTNIAGUINA, Evgenia. Consumer confidence and asset prices: Some empirical evidence. *The Review of Financial Studies*. 2006, vol. 19, no. 4, pp. 1499–1529.

23. BARBER, Brad M; ODEAN, Terrance; ZHU, Ning. Do noise traders move markets? In: *EFA 2006 Zurich meetings paper*. 2006.

24. BAKER, Malcolm; STEIN, Jeremy C. Market liquidity as a sentiment indicator. *Journal of financial Markets*. 2004, vol. 7, no. 3, pp. 271–299.

25. KAMSTRA, Mark J; KRAMER, Lisa A; LEVI, Maurice D. Winter blues: A SAD stock market cycle. *American Economic Review*. 2003, vol. 93, no. 1, pp. 324–343.

26. EDMANS, Alex; GARCIA, Diego; NORLI, Øyvind. Sports sentiment and stock returns. *The Journal of finance*. 2007, vol. 62, no. 4, pp. 1967–1998.

27. ANTWEILER, Werner; FRANK, Murray Z. Is all that talk just noise? The information content of internet stock message boards. *The Journal of finance*. 2004, vol. 59, no. 3, pp. 1259–1294.

28. TETLOCK, Paul C. Giving content to investor sentiment: The role of media in the stock market. *The Journal of finance*. 2007, vol. 62, no. 3, pp. 1139–1168.

29. SCHUMAKER, Robert P; ZHANG, Yulei; HUANG, Chun-Neng; CHEN, Hsinchun. Evaluating sentiment in financial news articles. *Decision Support Systems*. 2012, vol. 53, no. 3, pp. 458–464.

30. KHAN, Wasiat; GHAZANFAR, Mustansar Ali; AZAM, Muhammad Awais; KARAMI, Amin; ALYOUBI, Khaled H; ALFAKEEH, Ahmed S. Stock market prediction using machine learning classifiers and social media, news. *Journal of Ambient Intelligence and Humanized Computing*. 2020, pp. 1–24.

31. RENAULT, Thomas. Sentiment analysis and machine learning in finance: a comparison of methods and models on one million messages. *Digital Finance*. 2020, vol. 2, no. 1, pp. 1–13.
32. FIRTH, John R. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*. 1957.
33. CHOWDHARY, KR1442. Natural language processing. *Fundamentals of artificial intelligence*. 2020, pp. 603–649.
34. ZHANG, Lei; WANG, Shuai; LIU, Bing. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*. 2018, vol. 8, no. 4, e1253.
35. LIU, Bing. Aspect Sentiment Classification. In: *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press, 2015, pp. 90–136. Available from DOI: 10.1017/CBO9781139084789.006.
36. KOWSARI, Kamran; JAFARI MEIMANDI, Kiana; HEIDARYSAFA, Mojtaba; MENDU, Sanjana; BARNES, Laura; BROWN, Donald. Text Classification Algorithms: A Survey. *Information*. 2019, vol. 10, no. 4. ISSN 2078-2489. Available from DOI: 10.3390/info10040150.
37. BENGIO, Yoshua; DUCHARME, Réjean; VINCENT, Pascal. A neural probabilistic language model. *Advances in neural information processing systems*. 2000, vol. 13.
38. MIKOLOV, Tomas; SUTSKEVER, Ilya; CHEN, Kai; CORRADO, Greg S; DEAN, Jeff. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*. 2013, vol. 26.
39. PENNINGTON, Jeffrey; SOCHER, Richard; MANNING, Christopher D. Glove: Global vectors for word representation. In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.
40. ZHOU, Guo-Bing; WU, Jianxin; ZHANG, Chen-Lin; ZHOU, Zhi-Hua. Minimal gated unit for recurrent neural networks. *International Journal of Automation and Computing*. 2016, vol. 13, no. 3, pp. 226–234.
41. BENGIO, Yoshua; SIMARD, Patrice; FRASCONI, Paolo. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*. 1994, vol. 5, no. 2, pp. 157–166.
42. SCHUSTER, Mike; PALIWAL, Kuldip K. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*. 1997, vol. 45, no. 11, pp. 2673–2681.
43. VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N; KAISER, Łukasz; POLOSUKHIN, Illia. Attention is all you need. *Advances in neural information processing systems*. 2017, vol. 30.
44. CHO, Kyunghyun; VAN MERRIËNBOER, Bart; BAHDANAU, Dzmitry; BENGIO, Yoshua. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*. 2014.
45. CHO, Kyunghyun; VAN MERRIËNBOER, Bart; GULCEHRE, Caglar; BAHDANAU, Dzmitry; BOUGARES, Fethi; SCHWENK, Holger; BENGIO, Yoshua. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*. 2014.
46. YOUNG, Tom; HAZARIKA, Devamanyu; PORIA, Soujanya; CAMBRIA, Erik. Recent Trends in Deep Learning Based Natural Language Processing [Review Article]. *IEEE Computational Intelligence Magazine*. 2018, vol. 13, no. 3, pp. 55–75. Available from DOI: 10.1109/MCI.2018.2840738.
47. BLOEM, Peter. *Transformers from scratch*. 2019. Available also from: https://peterbloem.nl/blog/transformers.

48. HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. Deep Residual Learning for Image Recognition. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. Available from DOI: 10.1109/CVPR.2016.90.
49. DEVLIN, Jacob; CHANG, Ming-Wei; LEE, Kenton; TOUTANOVA, Kristina. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. 2018.
50. ARACI, Dogu. Finbert: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*. 2019.
51. MALO, Pekka; SINHA, Ankur; KORHONEN, Pekka; WALLENIUS, Jyrki; TAKALA, Pyry. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*. 2014, vol. 65, no. 4, pp. 782–796.
52. MAIA, Macedo; HANDSCHUH, Siegfried; FREITAS, André; DAVIS, Brian; MCDERMOTT, Ross; ZARROUK, Manel; BALAHUR, Alexandra. Www'18 open challenge: financial opinion mining and question answering. In: *Companion proceedings of the the web conference 2018*. 2018, pp. 1941–1942.
53. ZHONG, Xiao; ENKE, David. Forecasting daily stock market return using dimensionality reduction. *Expert Systems with Applications*. 2017, vol. 67, pp. 126–139. ISSN 0957-4174. Available from DOI: https://doi.org/10.1016/j.eswa.2016.09.027.
54. ARIYO, Adebiyi A.; ADEWUMI, Adewumi O.; AYO, Charles K. Stock Price Prediction Using the ARIMA Model. In: *2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*. 2014, pp. 106–112. Available from DOI: 10.1109/UKSim.2014.67.
55. RATHNAYAKA, RM Kapila Tharanga; SENEVIRATNA, DMKN; JIANGUO, Wei; ARUMAWADU, Hasitha Indika. A hybrid statistical approach for stock market forecasting based on artificial neural network and ARIMA time series models. In: *2015 International Conference on Behavioral, Economic and Socio-cultural Computing (BESC)*. IEEE, 2015, pp. 54–60.
56. BALLINGS, Michel; VAN DEN POEL, Dirk; HESPEELS, Nathalie; GRYP, Ruben. Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications*. 2015, vol. 42, no. 20, pp. 7046–7056. ISSN 0957-4174. Available from DOI: https://doi.org/10.1016/j.eswa.2015.05.013.
57. LONG, Jiawei; CHEN, Zhaopeng; HE, Weibing; WU, Taiyu; REN, Jiangtao. An integrated framework of deep learning and knowledge graph for prediction of stock price trend: An application in Chinese stock exchange market. *Applied Soft Computing*. 2020, vol. 91, p. 106205.
58. BENGIO, Yoshua; LECUN, Yann; HINTON, Geoffrey. Deep learning for AI. *Communications of the ACM*. 2021, vol. 64, no. 7, pp. 58–65.
59. SELVIN, Sreelekshmy; VINAYAKUMAR, R; GOPALAKRISHNAN, EA; MENON, Vijay Krishna; SOMAN, KP. Stock price prediction using LSTM, RNN and CNN-sliding window model. In: *2017 international conference on advances in computing, communications and informatics (icacci)*. IEEE, 2017, pp. 1643–1647.
60. NIKOU, Mahla; MANSOURFAR, Gholamreza; BAGHERZADEH, Jamshid. Stock price prediction using DEEP learning algorithm and its comparison with machine learning algorithms. *Intelligent Systems in Accounting, Finance and Management*. 2019, vol. 26, no. 4, pp. 164–174.
61. SONKIYA, Priyank; BAJPAI, Vikas; BANSAL, Anukriti. Stock price prediction using BERT and GAN. *arXiv preprint arXiv:2107.09055*. 2021.
62. LI, Shiyang; JIN, Xiaoyong; XUAN, Yao; ZHOU, Xiyou; CHEN, Wenhu; WANG, Yu-Xiang; YAN, Xifeng. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*. 2019, vol. 32.

63. ZERVEAS, George; JAYARAMAN, Srideepika; PATEL, Dhaval; BHAMIDIPATY, Anuradha; EICKHOFF, Carsten. A transformer-based framework for multivariate time series representation learning. In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2021, pp. 2114–2124.

64. GRIGSBY, Jake; WANG, Zhe; QI, Yanjun. Long-range transformers for dynamic spatiotemporal forecasting. *arXiv preprint arXiv:2109.12218*. 2021.

65. ZHOU, Haoyi; ZHANG, Shanghang; PENG, Jieqi; ZHANG, Shuai; LI, Jianxin; XIONG, Hui; ZHANG, Wancai. Informer: Beyond efficient transformer for long sequence time-series forecasting. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2021, vol. 35, pp. 11106–11115. No. 12.

66. WANG, Chaojie; CHEN, Yuanyuan; ZHANG, Shuqi; ZHANG, Qiuhui. Stock market index prediction using deep Transformer model. *Expert Systems with Applications*. 2022, vol. 208, p. 118128.

67. ZHANG, Qiuyue; QIN, Chao; ZHANG, Yunfeng; BAO, Fangxun; ZHANG, Caiming; LIU, Peide. Transformer-based attention network for stock movement prediction. *Expert Systems with Applications*. 2022, vol. 202, p. 117239.

68. DING, Qianggang; WU, Sifan; SUN, Hao; GUO, Jiadong; GUO, Jian. Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction. In: *IJCAI*. 2020, pp. 4640–4646.

69. NASSIRTOUSSI, Arman Khadjeh; AGHABOZORGI, Saeed; WAH, Teh Ying; NGO, David Chek Ling. Text mining of news-headlines for FOREX market prediction: A Multi-layer Dimension Reduction Algorithm with semantics and sentiment. *Expert Systems with Applications*. 2015, vol. 42, no. 1, pp. 306–324.

70. PHILIPPE REMY, Xiao Ding. *Financial News Dataset from Bloomberg and Reuters* [`https://github.com/philipperemy/financial-news-dataset`]. GitHub, 2015.

71. ARGUNOV, Gennadiy. *Historical financial news archive* [`https://www.kaggle.com/datasets/gennadiyr/us-equities-news-data`]. Kaggle, 2019.

72. LEWIS, David D. *Reuters-21578 text categorization collection data set*. 1997.

73. FALINOUSS, Pegah. *Stock trend prediction using news articles: a text mining approach*. 2007.

74. *Finance.yahoo.com traffic analytics & market share — similarweb*. [N.d.]. Available also from: `https://www.similarweb.com/website/finance.yahoo.com/`.

75. JAGGI, Mukul; MANDAL, Priyanka; NARANG, Shreya; NASEEM, Usman; KHUSHI, Matloob. Text mining of stocktwits data for predicting stock prices. *Applied System Innovation*. 2021, vol. 4, no. 1, p. 13.

76. BATRA, Rakhi; DAUDPOTA, Sher Muhammad. Integrating StockTwits with sentiment analysis for better prediction of stock price movement. In: *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*. IEEE, 2018, pp. 1–5.

77. SOUSA, Matheus Gomes; SAKIYAMA, Kenzo; SOUZA RODRIGUES, Lucas de; MORAES, Pedro Henrique; FERNANDES, Eraldo Rezende; MATSUBARA, Edson Takashi. BERT for stock market sentiment analysis. In: *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2019, pp. 1597–1601.

78. HIEW, Joshua Zoen Git; HUANG, Xin; MOU, Hao; LI, Duan; WU, Qi; XU, Yabo. BERT-based financial sentiment index and LSTM-based stock return predictability. *arXiv preprint arXiv:1906.09024*. 2019.

79. HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

80. KAZEMI, Seyed Mehran; GOEL, Rishab; EGHBALI, Sepehr; RAMANAN, Janahan; SAHOTA, Jaspreet; THAKUR, Sanjay; WU, Stella; SMYTH, Cathal; POUPART, Pascal; BRUBAKER, Marcus. Time2vec: Learning a vector representation of time. *arXiv preprint arXiv:1907.05321*. 2019.

81. RADFORD, Alec; WU, Jeffrey; CHILD, Rewon; LUAN, David; AMODEI, Dario; SUTSKEVER, Ilya, et al. Language models are unsupervised multitask learners. *OpenAI blog*. 2019, vol. 1, no. 8, p. 9.

82. WANG, Sinong; LI, Belinda Z; KHABSA, Madian; FANG, Han; MA, Hao. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*. 2020.

Media contents

- `readme.txt` ..... media contents description
- `src`
  - `environment.yml` ..... conda environment file
  - `bp-scrapper` ..... directory containing Scrapy spiders used to gather the datasets
  - `bp-scrapper-data`
    - `news.jsonl` ..... dataset of general market news
    - `price.jsonl` ..... dataset of companies' stock prices
    - `ticker_news.jsonl` ..... dataset of company-specific news
  - `bp-informer` ..... directory containing the Informer source code
    - `data-processor.ipynb` ..... notebook showcasing the experiments
- `text`
  - `thesis.pdf` ..... text of the thesis in PDF format