UNIVERSITÄT GRAZ
UNIVERSITY OF GRAZ

Patrick Deutschmann, BSc

Sentiment for Price Prediction (SePP): Predicting Stock Performance Based on Tweets

Bachelor's Thesis

University of Graz

Institute of Banking and Finance
Head: Univ.-Prof. Dr. Andrea Schertler

Supervisor: Assoz. Prof. Mag. Dr.rer.soc.oec. Stefan Palan

Graz, November 2021

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, _____ _____

Date Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am _____ _____

Datum Unterschrift

Abstract

Stock prices are influenced by investors' beliefs, which many express on social media. As recent advances in machine learning have achieved impressive results in detecting sentiment in human language, this thesis aims to apply them to the task of stock price prediction. We present a machine learning model that takes as input tweets about certain assets, derives their sentiments and predicts whether the stock price will go up or down after that. Our model achieves a new state of the art in prediction accuracy on the StockNet data set and demonstrates that tweet sentiments have a discernible effect on stock prices.

Contents

List of Figures

1 Introduction

The Efficient Market Hypothesis (EMH) states that if a market is strong-form efficient, stock prices reflect all relevant information. This would mean that all assets are priced correctly based on the expectation of what they will be worth in the future [38]. Their development should strictly follow a random walk pattern, as only new information could cause changes. However, there have been studies in behavioural finance showing that, in practice, at least the strong form does not always hold [27, 38]. One factor that affects stock prices is market sentiment. As shown by [2], investors' attitudes have a clearly discernible effect on prices.

In the meantime, Machine Learning (ML), a field quite far away from behavioural finance, has made significant advances in understanding sentiment in human language. The advent of accurate sentiment detection algorithms paired with a steady increase of online financial commentary brought about the idea of analysing the influences of online posts on stock markets. One key question is whether algorithms could use these online comments and their sentiments to predict future prices.

This thesis project aims to analyse the influence of a particularly famous form of short, written online comments, i.e., tweets, on the stock market using ML. In particular, we will analyse the comments using Natural Language Processing (NLP), a subfield of ML, and try to make predictions of subsequent changes in stock prices.

Previous work aiming to predict stock market movements based on tweets, such as [31] and [6], uses sentiment analysis to extract the nature of the statement and draw conclusions about potential future stock developments. However, in recent years, there have been significant advances in NLP deep learning models, mostly building on the success of Transformers, initially introduced by [40], that have also dramatically improved how such models understand text sentiments, as shown in [32]. Hence, applying these models to this task could show promising results.

Building upon these new sentiment detection approaches, we construct a model wherein only Twitter data is used to predict stock price changes. We intentionally

avoid using historical market data as input so that the model is forced to focus on the sentiment alone and cannot utilise patterns in the market data, the understanding of which would likely only be beneficial in the short term.

The proposition that a model can predict stock returns based on tweet sentiments builds on the assumption that they reflect Twitter users' attitudes towards a company and, by extension, their expectation of its stock price development. Even though the data might be noisy and there might very well exist tweets where the sentiment is not reflective of the investor's expectation of a stock's development, we retain the assumption that, in aggregate, the tendencies observed in the tweet sentiments are sufficient to make predictions to a significant degree of precision.[1]

In short, our contributions are as follows:

- We introduce a new stock prediction model that takes tweets as input and predicts whether a stock will go up or down in the next time step, based on the sentiment it detects in the tweets. It achieves new state-of-the-art results for accuracy on the StockNet data set [43].
- We compare our approach to similar techniques and derive how much predictability is provided by only using tweet data rather than historical market information.
- We evaluate the real-world applicability of our method and analyse theoretical conclusions that can be drawn from the results.

This thesis is structured in five chapters. After this introduction, we present related work in the form of relevant background and existing methods in Chapter 2. We then document our methodology, including a detailed problem formulation, a description of the data, the model and the experimental results in Chapter 3. Then, in Chapter 4, we discuss our findings with respect to real-world applications, theoretical consequences and potential for future research. Finally, we conclude our work in Chapter 5.

[1] All code and hyper-parameters are available at `https://github.com/deutschmn/stock-tweets`.

2 Related Work

The following chapter presents relevant background and the current state of the art for stock market prediction.

2.1 Background

In this section, we provide preliminary information on the background relevant to this project. We introduce relevant concepts and terminology first about machine learning and then about stock markets.

2.1.1 Machine Learning (ML)

The term ML is often used synonymously with *Artificial Intelligence (AI)* and *Deep Learning (DL)*, but there exist quite fundamental differences between them. ML is a sub-field of AI, whereas DL is a sub-field of ML. While AI is the umbrella term for intelligence exhibited by machines [35], *machine learning* is defined by [18] as "the study of computer algorithms that improve automatically through experience". Algorithms that use machine learning are typically referred to as *models*. Any algorithm's behaviour depends on its *parameters*. However, the main difference between an ML model and a classical algorithm is that in ML, the parameters are *learned* from data. Therefore, ML models are also called *data-driven* approaches. Teaching a model its parameters is called *training* it. Parameters are not to be confused with *hyper-parameters*. While the former are learned, the latter are specified by hand in advance. They can, for example, determine the architecture of the model or the way it is trained.

While ML itself is a sub-field of computer science, it is also a sister discipline to statistics. As both fields are rapidly evolving, differences and similarities are debated and subject to varying opinions. Commonly considered differences between them are that statistics are more concerned with formal proofs, survey design and sampling. Machine learning, on the other hand, is more focussed

on predictive performance, tolerating a lack of theoretical backing to a certain extent [1].

Deep Learning (DL) is an ML technique that builds on deep neural networks, hence the name. It has gained increasing popularity in recent years as it achieved remarkable results in various domains [37]. A typical *feed-forward* neural network, also known as *Multi-layer Perceptron (MLP)*, consists of multiple *layers* that are separated by so-called *activation functions*. In the beginning, the input is fed in as a numerical representation and is then processed sequentially through the layers until the final prediction comes out in the end. One of the most important layers is a *linear* layer that performs a matrix multiplication of the input it receives with its weight matrix, a learnable parameter. Common activation functions include *Sigmoid*, *Rectified Linear Unit (ReLU)* and *LeakyReLU* [15]. In this project, we use a deep neural network to derive the sentiment of tweets and then predict how it influences the stock price.

Natural Language Processing (NLP), a sub-field of ML, is concerned with building models that can deal with human language. Almost all NLP models nowadays are neural networks, mostly building on the Transformer architecture [40]. These models are very large as they have millions of parameters. To achieve good results, they have to be trained on a lot of data on powerful and expensive hardware. Fortunately, large language models also exhibit the useful attribute of generalising well to new tasks. This means that the parameters of models that others have *pre-trained* can be re-used and *fine-tuned* to different but related tasks. This concept is called *transfer learning*. For instance, in this project, we use models that have been pre-trained on sentiment detection. Then, we only fine-tune them on our data set and teach them the connection to the stock price.

In a so-called *supervised* setting, which is the case for this project, training a neural network consists in feeding a model certain inputs, letting it come up with a prediction and then comparing it to the expected outcome, the so-called *target* or *label* [15]. In our case, we feed our model the tweets and compare its prediction for the stock price movement to what we have really observed. The tweets are split up into multiple tokens, which are then, in turn, converted to numeric representations that the neural network can process. This process is called *tokenisation* and depends on the model used.

The predictions are then compared to the labels, and the *error* (or *loss*) is computed based on a *loss function* that is chosen depending on the task. Common loss functions include *Mean Squared Error (MSE) loss* for regression tasks or *Binary Cross Entropy (BCE) loss* for classification tasks. The model learns from

back-propagating the error through the network, calculating the gradients with respect to the individual parameters. The step when predictions are computed, is called *forward pass*, whereas the step where the gradients from the errors are derived is called *backward pass*. Finally, a *gradient step* towards correcting the error is made, i.e., the parameters are changed a little bit. The degree to which they are adapted (the size of the gradient step) depends on the *Learning Rate (LR)*.

In order to train the model, an optimisation algorithm, such as *Adam* [21] or *AdamW* [25], is used. As the errors of the predictions are propagated through the whole model, all the way to the inputs, training is often called *end-to-end*. When training a model, it is presented the *training data* multiple times. The unit of time in which the model sees the whole training data set is called an *epoch*. Neural networks are typically trained on Graphics Processing Units (GPUs) as they are well suited for parallel processing. When multiple training samples are processed at the same time, they are called a *batch*.

2.1.2 Stock Markets

The goal of stock market prediction is to tell in advance how prices are going to develop. If it can be achieved consistently, *excess returns* can be generated, i.e., returns that exceed those of the broad market as a whole. This concept is also known as *beating the market* or *generating alpha* [10].

There are two major approaches for stock market prediction: *fundamental analysis* and *technical analysis* [26]. *Fundamental analysis* tries to evaluate a company's stock by its intrinsic value, i.e., the sum of its discounted future cash flows. It approximates these by the observable behaviour and performance of companies. *Technical analysis* on the other hand, only considers trends and patterns in the stock market, for example, stock price developments. It relies on the assumption that all factors that drive the intrinsic value are already factored into the price.

Another important factor at play is *market sentiment* [39]. The concept is not to be confused with the NLP term *sentiment analysis*, which typically refers to the sentiment of a particular textual statement. In contrast, in finance, it refers to the overall attitude of market participants towards assets. Technical indicators allow measuring it by assessing whether prices are rising or falling.

A question worth asking is whether it is even possible to beat the market. The well-known Efficient Market Hypothesis (EMH) aims to answer that by introducing the concept of market efficiency. A market is called *efficient* with

respect to an information set if the price of an asset fully reflects that information set [38]. The EMH states that financial markets are indeed efficient, meaning that generating alpha should not be possible. The qualification concerning the information set brings about weak-, semistrong- and strong-form efficiency. In a weak-form efficient market, the relevant information set is only market trading data. Hence, in such a market, investors should not be able to generate excess returns using technical analysis. In a semistrong-form efficient market, the information set is all publicly available, relevant information. Finally, in a strong-form efficient market, it is all relevant information, even including insider information. If markets are semi-strong- or strong-form efficient, fundamental analysis, too, should not work. Prices would follow random walk patterns and change only if new information came to light [29].

The EMH is supported by solid evidence, at least in the weak and semistrong form [38]. However, some results disagree with it, often due to investors' irrational behaviour, resulting in pricing irregularities and predictable patterns [27, 2]. Moreover, as [16] stresses, a market cannot be perfectly efficient, as otherwise, market participants would have no incentive to uncover mispricings. Despite that, [27] asserts that even though irregularities exist, they are unlikely to persist for a long time and are hard for investors to exploit consistently.

2.2 State of the Art

Machine learning approaches for stock market predictions can mainly be differentiated by two dimensions: (i) the data they receive as input and (ii) the architecture of the models. In terms of input data, models that perform technical analysis only use past price information to predict future prices. On the other hand, models that perform fundamental analysis also consider other factors, such as companies' performance indicators. For this project, we consider approaches that try to capture market sentiment through online news information. We restrict our analysis to those evaluated on the StockNet data set [43] for comparability reasons. With regard to the model architecture, we distinguish between classical ML approaches and modern deep learning models. An overview of the methods we compare ourselves to is depicted in Table 2.1. The results on the StockNet data set [43] are shown in Table 3.2.

As stock prediction is a time series problem, technical analysis can be performed by typical methods such as Autoregressive Integrated Moving Average (ARIMA), which is a generalisation of the Autoregressive Moving Average (ARMA) model [9]. It is a classical ML method that is limited by the data

		Architecture	
		Classical ML methods	*Deep Learning*
Input	*Past prices only*	ARIMA [9]	Adv-ALSTM [14]
	News only	RandForest [33]	*our approach* HAN [19]
	Both	-	MFNN [24] StockNet [43] SMPN [42]

Table 2.1: Categorisation of different stock prediction approaches

it can process. In this case, ARIMA cannot trivially be extended to process text information. The method presented in [33] only feeds in news information, processes it using n-grams [28] and word2vec embeddings [30] and then predicts the price movement using random forests [8]. While this shows promising results, understanding human language is a complex task that led to the development of more elaborate models.

In particular, deep learning models have produced significant advances in this area. For NLP tasks, recurrent neural networks showed considerable improvements [44]. As such, [19] proposed Hybrid Attention Networks (HAN) which combine recurrence with attention mechanisms [40] for the task of stock predictions. As deep learning models are generally able to process heterogeneous inputs, further efforts started using not only the news as input but also fed in past price information to improve the accuracy of their predictions. Various incremental improvements were achieved through Multi-Filters Neural Networks (MFNN) [24], recurrent, continuous latent variables [43] and incorporative attention mechanisms [42]. All of these models are very specialised in predicting stock prices using both news information and prices. Another model that performs well on the StockNet data set [43] is Adv-ALSTM [14], even though it only considers price information and disregards the news input completely. It uses adversarial training to improve generalisation.

This surprising result brought about the idea of doing the opposite in this project: We completely disregard past price information and explore which accuracy can be reached when only tweets are used as model inputs. This makes the task harder than the one tackled by [43, 42, 24], and also positions our approach farther from practical applications, as a real-world algorithmic trading agent would usually have access to real market data. However, this thesis aims not to present itself as a candidate for a state-of-the-art algorithmic trading agent but rather to investigate the extent to which tweet sentiment by itself is feasible for stock market predictions.

Thereby, our method remains directly comparable to [19]. We expect to gain improvements in prediction performance by applying state-of-the-art sentiment analysis models based on Transformers [40] to the stock prediction task. In particular, our models build on BERT [13] and its derivations RoBERTa [23] and ALBERT [22].

3 Methodology

This chapter forms the central part of the thesis and outlines its methodology. It describes the specific problem that is being tackled, the data used, and the model proposed. Finally, the experimental results are shown.

3.1 Problem formulation

Formally, the research problem this thesis is aiming to solve is the following: Let $\mathcal{T} = \{x_i | 1 \leq i \leq N\}$ be a set of tweets about an asset a for which each tweet x_i is defined as the tuple of its textual content c_i, the number of followers of its author f_i and the time it was posted t_i, i.e., $x_i = (c_i, f_i, t_i)$. Predict the movement direction

$$d_a^t = \begin{cases} 1, & \text{if } r_a^t \geq u \\ 0, & \text{if } r_a^t \leq l \\ -1, & \text{otherwise} \end{cases} \tag{3.1}$$

where r_a^t are the returns of the asset on the prediction day t, u is an upper and l is a lower limit for classifying the movement as up (1) or down (0). The returns are measured as rates of return, i.e. the relative changes in adjusted closing prices. We denote d_a^t as the real, observed direction and $\hat{d}_a^t$ as the prediction.

As samples for which the price is considered to stay the same (-1), i.e. $l < r_a^t < u$ are filtered out, the task is reduced to a binary classification problem. u and l are hyper-parameters that we set reasonably so that the training data set is balanced. For concrete values, see Table 3.1.

The tweets in $\mathcal{T}$ are collected from before day t. We experimented with two configurations for selecting the relevant set of input tweets:

- **Max lag**: We consider all tweets until the max lag value Δt, i.e., $\mathcal{T} = \{x_i | t - \Delta t < t_i < t\}$. For instance, if $\Delta t = 3$, we consider all tweets posted between three days ago and the event day.
- **Fixed lag**: We only consider tweets from a fixed lag day, i.e., $\mathcal{T} = \{x_i | t_i = t - \Delta t\}$. In this mode, if $\Delta t = 3$, we only consider tweets from exactly three days before the event.

By this formulation, the model makes the intentional simplification of not considering intraday changes. Also, it only gets as input the set of tweets and no historical market information whatsoever.

3.2 Data

When using data-driven approaches, the choice of the data set is essential for the success of the analysis. For this project, Twitter data and stock market price information is required.

However, in order for a language model to learn to predict the sentiment of a tweet, it needs to be trained on a large corpus of text. Fortunately, sentiment analysis is well-studied, and various good data sets exist. The *SemEval-2017 task 4* [34], which forms part of the more general *TweetEval* [3], is rather coarse-grained with only three labels (*positive*, *neutral* and *negative*), while others, such as *emotion* [36] and *go_emotions* [12] are more fine-grained with 6 and 27 emotions respectively. In this project, we use transfer learning, i.e., we employ models that have been pre-trained on other data sets so they can directly be fine-tuned.

The other remaining ingredient for the task at hand is the asset price information. It can be costly to obtain, especially in high frequencies. Therefore, we use the *StockNet* data set[1] from [43], which contains both tweets and price information. It provides data of 88 stocks from 01/01/2014 to 31/12/2015. Additionally to being more easily accessible, using an existing data set instead of collecting a new one increases the comparability of our results.

The pre-processing is reduced to a minimum as the machine learning model is trained in an end-to-end fashion. The tasks performed are the following:

- From the raw tweets and prices, data samples are constructed as follows: For each day t, for each stock a, a sample is created that consists of all

[1] `https://github.com/yumoxu/stocknet-dataset`

relevant tweets $\mathcal{T}$ (according to the time lag mode) and the movement in percent on prediction day t. We call one such sample a *movement*.
- The movements are split up temporally the same way as it is done in [42, 43] for comparability reasons. That is, data from 01/01/2014 – 31/07/2015 makes up the training set, 01/08/2015 – 30/09/2015 the validation set and 01/10/2015 – 31/12/2015 the test set. The training set is used to train the model, the validation set is used to find the ideal hyper-parameters, and the test set is used to compute performance metrics that are ultimately reported and used to compare to other methods.
- We clean the tweet texts with standard approaches to facilitate comprehension of the language model: remove URLs, newline characters, duplicate spaces and cashtags. Finally, we also discard duplicate tweets within movements, as they do not contribute new information.
- We experimented with only considering movements for which at least a certain number of tweets is available, but found that it did not make a large practical difference as for most movements, sufficient information was present.

3.3 Model

This section describes the model architecture we use. The hyper-parameters are explained in detail in section 3.4.

The machine learning model we use to solve the task is a deep neural network. We call it *Sentiment for Price Prediction*, `SePP` for short, and propose two variants. The first, `SePP.Pipe`, is a pipeline approach that first detects the sentiment of every tweet and then performs a weighting and prediction step. The second, `SePP.Int`, is an integrated model where all tweets are concatenated. The model then predicts their overall sentiment, and, lastly, a prediction for the stock price movement based on the joint sentiment representation is derived.

Our model $\mathcal{M}$ predicts the movement direction of an asset a at day t as

$$\hat{d}_a^t = \mathcal{M}(\mathbf{c}, \mathbf{f})$$

where $\mathbf{c}$ is the vector of tweet contents c_i and $\mathbf{f}$ the vector of followers f_i.

Due to the nature of the task, which can, in essence, be seen as a binary classification problem of the direction of returns, we use Binary Cross Entropy (BCE) loss, defined as follows:

$$\mathcal{L}(\mathbf{d}, \hat{\mathbf{d}}) = \frac{1}{N} \sum_{i=1}^{N} \left(d_i \log \left(\hat{d_i} \right) + (1 - d_i) \log \left(1 - \hat{d_i} \right) \right)$$

where $\hat{\mathbf{d}}$ is a batch of predicted directions and $\mathbf{d}$ are the real, observed directions. We chose the BCE loss function, as it speeds up training and contributes to better generalisation in comparison with Mean Squared Error (MSE) [5].

The function returns zero if all predictions exactly match the labels, but punishes incorrect predictions, especially those taken with high confidence. Consequently, the training procedure, which is the optimisation problem of minimising the loss, can be written as

$$\theta^* = \arg\min_{\theta} \mathcal{L}(\mathbf{d}, \mathcal{M}_\theta(\mathbf{c}, \mathbf{f}))$$

where θ are the parameters of the model that are learned during training.

In the following subsections, both our model variants are explained in detail.

3.3.1 Pipeline (`SePP.Pipe`)

A schematic version of `SePP.Pipe` is depicted in Figure 3.1. In mathematical notation, we denote it as $\mathcal{M}_P$. All tweet contents c_i are fed into a sentiment detection model Φ that yields for every tweet a sentiment s_i. These sentiments are then, together with the number of followers of the tweet author f_i, put into a weighting module Ω, that yields a joint sentiment $\bar{s}_a^t$. This representation that can be interpreted as an estimation of Twitter users' sentiment about asset a on day t then is fed into a prediction module Θ that computes the expected stock price movement direction of the asset $\hat{d}_a^t$. Taken together, the predictions are made with

$$\begin{aligned} \hat{d}_a^t &= \mathcal{M}(\mathbf{c}, \mathbf{f}) \\ &= \Theta(\Omega(\Phi(\mathbf{c}), \mathbf{f})). \end{aligned}$$

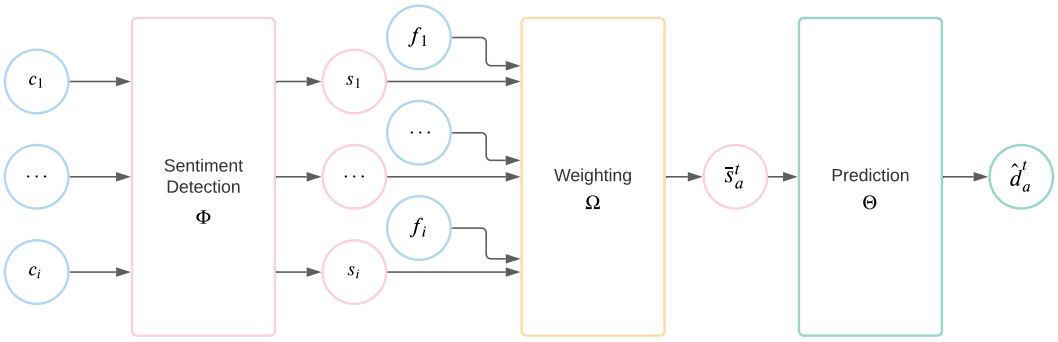

Figure 3.1: Schematic depiction of the model `SePP.Pipe`.

In the following, we explain the different sub-modules of the model in detail.

Sentiment detection

The first part of the model is the sentiment detection module Φ. It takes as input a batch of tweet contents $\mathbf{c}$ and computes for every tweet c_i a corresponding sentiment s_i.

This is a typical NLP task, known as *sentiment analysis*. Much research has been done in this field, and large language models have made significant advances in recent years. These models are typically trained on huge corpora to produce good accuracy. Fortunately, they are built to support transfer learning, i.e. the possibility of taking pre-trained versions of them and fine-tuning them to the task at hand. This is also what our model is built upon.

The inputs to the model are always the same, starting with the tweet contents c_i. However, before the text can be fed into the model, it needs to be prepared by a tokeniser that splits it up into multiple tokens, removes unknown ones and brings them into the encoding that the model expects. This might vary depending on the model used.

The output $s_i \in \mathbb{R}^K$ varies in shape, i.e. K is dependant on the pre-trained model used. For example, a pre-trained RoBERTa [23] model[2] has $K = 3$ where $s_{i,1}$ is the probability of the text having a negative sentiment, $s_{i,2}$ a neutral one and $s_{i,3}$ a positive one. The sum of all probabilities is one, i.e. $|s_i| = 1$. The output of this sub-module is the sentiment of each tweet, which is then further processed by the weighting sub-module.

[2] `https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment`

We experiment with the following pre-trained models from *Hugging Face*[3]:

- **BERT**[4]: This model builds on the BERT architecture [13] and, therefore, has 12 layers, a hidden size of 768, 12 attention heads and a total of 168 million parameters. The version we are using is fine-tuned for sentiment analysis on product reviews in English, Dutch, German, French, Spanish and Italian. It predicts sentiments in the number of stars from 1 to 5, hence, $K = 5$.
- **RoBERTa**[5]: Building on the RoBERTa architecture [23], this model has the same number of layers, attention heads and hidden size as the BERT model, but it only has 125 million parameters. It has been pre-trained on the TweetEval benchmark [3] for sentiment analysis and predicts probabilities for *positive*, *neutral* or *negative*, i.e., $K = 3$.
- **ALBERT**[6]: Finally, the ALBERT model [22] is significantly smaller than the other two and only has 11 million parameters. It is also not specifically pre-trained for sentiment analysis but only for general language tasks. We included it into our experiments hoping that it might be trainable more easily on our smaller data set. As it is not yet fine-tuned, we train the sentiment classification head from scratch and therefore set $K = 2$, for our use case of *up* and *down* movements.

As detailed in Section 3.4, we found the BERT model to perform best in our experiments.

Weighting

At this stage, the model has the sentiments s_i for each tweet and now computes a joint representation for all tweet sentiments $\bar{s}_a^t$. This can be interpreted as the collective opinion of Twitter users of asset a for day t. To derive this quantity, the sub-module has to learn the importance of individual tweets relative to each other.

Therefore, the outputs of the sentiment detection together with the number of followers $\mathbf{f}$ are fed into the weighting sub-module Ω:

$$\bar{s}_a^t = \Omega(\Phi(\mathbf{c}), \mathbf{f}).$$

[3] https://huggingface.co

[4] https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment

[5] https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment

[6] https://huggingface.co/albert-base-v2

As the name of the sub-module suggests, Ω essentially performs a weighting operation, wherein $\bar{s}_a^t$ is a combination of the sentiments s_i. However, the weighting coefficients are not simply a normalised version of the follower counts but are a function therein, i.e. the parameters determining the importance of individual tweets are a linear function in the tweet author's follower count. Thereby, the sub-module can learn dynamically how important the follower count is and whether a particular sentiment should have a constant bias irrespective of the followers.

Prediction

The final sub-module of the model Θ performs the actual prediction of the return direction, i.e.

$$\hat{d}_a^t = \Theta(\bar{s}_a^t).$$

The prediction sub-module is a Multi-layer Perceptron (MLP) that takes in the sentiment representation $\bar{s}_a^t$ and outputs the predicted return direction d_a^t. The MLP consists of a linear layer that projects the output into Θ_D-dimensional latent space, a LeakyRELU activation function and a final linear layer that compresses the representation down to one dimension. Θ_D is a hyper-parameter. Running a sigmoid activation function on the last layer's outputs yields the final probability for an upwards movement, meaning that if it is greater than 0.5 it is classified as *up* and otherwise as *down*.

3.3.2 Integrated (`SePP.Int`)

The integrated variant of our model, which we call $\mathcal{M}_I$ in mathematical notation, uses similar components as the pipeline approach. A schematic version of it is depicted in Figure 3.2. In contrast to the pipeline approach, the sentiments of the tweets are not derived individually but jointly. This means that they are simply concatenated before they are fed into the sentiment detection module Φ. Formally, the predictions are generated as

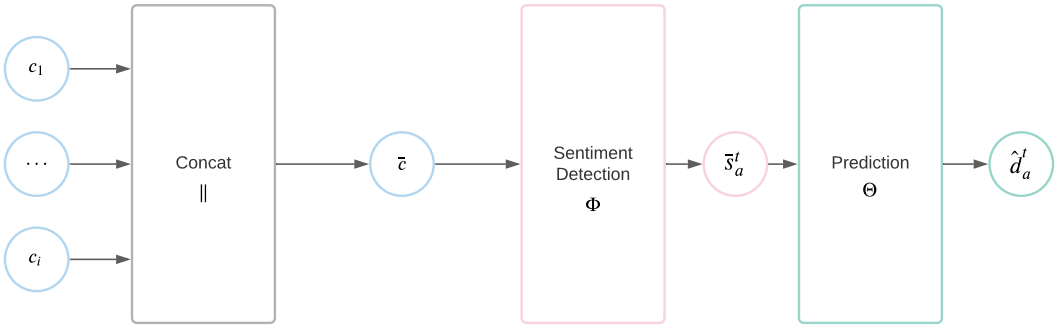

Figure 3.2: Schematic depiction of the model `SePP.Int`.

$$\begin{aligned}\hat{d}_a^t &= \mathcal{M}_I(\mathbf{c}) \\ &= \Theta\left(\Phi\left(\bar{c}\right)\right) \\ &= \Theta\left(\Phi\left(\mathop{\Big\|}_{i=1}^{N} c_i\right)\right)\end{aligned}$$

where $\bar{c} = \|_{i=1}^{N} c_i = c_1 \,\|\, \ldots \,\|\, c_N$ denotes the concatenation of the tweet contents.

Note that this model variant does not make use of the number of followers of the tweet authors $\mathbf{f}$ at all, as it would be non-trivial to incorporate them into the Transformer inputs. Future work might explore this possibility by adding additional input embeddings for them and possibly the time lag.

3.3.3 Variant comparison

This subsection compares the pipeline with the integrated architecture and points out the advantages and disadvantages of the two. The following are advantages of `SePP.Int` over `SePP.Pipe`:

- Many individual tweets are noise. When considering them in isolation, it is not possible to derive a clear sentiment and draw a conclusion about the future stock price. Therefore, the model has to learn to ignore them for the final prediction or take multiple tweets together and come up with a joint representation. This is much harder for `SePP.Pipe`, as it can only do so in the weighting module Ω, which knows much less about the data than the Transformer in the sentiment detection module Φ. In `SePP.Int`,

all tweets are concatenated and processed jointly by Φ, teaching it to focus on the relevant tweets. Hence, it should be more resistant to noise.
- The pipeline architecture requires the tweets to be processed by the sentiment detector individually, and only then they can be fed into the weighting module. This is computationally less efficient and makes the variant more resource-intensive to train, as the gradients for all tweets must be kept in memory for the backward pass.
- The integrated architecture is simpler and less task-specific. This can be advantageous if new tasks should be tackled with it.

However, there are also some disadvantages of `SePP.Int` when comparing it to `SePP.Pipe`:

- The integrated variant does not consider the number of followers of a tweet's author at all. As this can provide valuable information[7] to which the model does not have access, the pipeline variant has an advantage in this respect.
- The pipeline approach is more interpretable in that it allows to analyse which tweets have contributed to which degree to the final prediction.
- The integrated model is architecturally limited in the maximum number of tweets it can process. For instance, for the RoBERTa [23] models we consider, all tweets taken together can consist of a maximum of 512 tokens. At the same time, `SePP.Pipe` can theoretically be scaled up to an infinite number of tweets, given the availability of GPU memory. However, that does not necessarily mean that the model will be semantically able to make sense of that many tweets.

In the end, we found that both variants yield good results if they are trained correctly, with `SePP.Pipe` performing slightly better in our experiments. However, as the results are relatively close together, further tuning might change this.

3.4 Experiments

We ran an extensive number of experiments to find the best configuration of hyper-parameters for the task. Fine-tuning the largely pre-trained models until convergence took around 15 minutes for `SePP.Int` and 30 minutes for `SePP.Pipe` on an NVIDIA Tesla K80 GPU with 12GB of memory. Computing a

[7] `https://www.cnbc.com/2021/01/29/elon-musks-tweets-are-moving-markets.html`

Parameter	`SePP.Pipe`	`SePP.Int`
Optimiser	AdamW [25]	
LR	$\{10^{-3}, 10^{-4}, \mathbf{10^{-5}}\}$	
Prediction module hidden dim. Θ_D	$\{16, 32, \mathbf{64}\}$	
Time lag mode	$\{\mathbf{fixed}, \max\}$	
Time lag Δt	$\{0, 1, \mathbf{2}, 5, 10, 20\}$	
Threshold for classification as "down" l	-0.5%	
Threshold for classification as "up" u	$+0.6\%$	
Pre-trained sentiment model	{**BERT**, RoBERTa, ALBERT}	
Max. tweet length	512	60
Max. num tweets	50	-
Batch size	1	8

Table 3.1: Hyper-parameters we experimented with. Those that achieved top results are marked as bold.

prediction for a single movement, i.e., performing inference, takes 0.08 seconds for `SePP.Int` and 0.13 seconds for `SePP.Pipe`.

The model was trained on the training set and, after every epoch, evaluated on the validation set. We selected the best hyper-parameters based on the results of the validation test. After choosing, we computed the metrics on the test set and reported these in Table 3.2. The hyper-parameters with which we experimented are depicted in Table 3.1.

We trained our networks using early stopping, i.e., we stopped training when the validation loss started increasing. That already occurred after two to three epochs for the best performing models, despite using a standard dropout probability in the Transformer of 0.1. We also found a fixed time lag to work better than the maximum approach. One reason for that could be that samples that are too close to the date t are not yet relevant for the price, i.e., the sentiment has not materialised into a stock price change yet. We achieved top results when setting $\Delta t = 2$, considering tweets from two days before.

The implementation of our model is written in *Python*[8]. It uses the frameworks *PyTorch*[9] and *PyTorch Lightning*[10] for deep learning, *Hugging Face Transform-*

[8] `https://python.org`
[9] `https://pytorch.org`
[10] `https://pytorchlightning.ai`

ers[11] for NLP and *TorchMetrics*[12] and *pandas*[13] for various pre-processing, post-processing and evaluation tasks. For experiment tracking and visualisations, we use *Weights & Biases*[14] [4].

For comparability with other approaches, we evaluated our model on both accuracy and Matthews Correlation Coefficient (MCC) defined as

$$\begin{aligned} \text{Accuracy} &= \frac{1}{N}\sum_{i=1}^{N} 1\left(d_i = \hat{d}_i\right) \\ &= \frac{TP + TN}{TP + TN + FP + FN} \end{aligned}$$

and

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}}$$

where TP are true positives, TN are true negatives, FP are false positives and FN are false negatives. MCC is a good measure for imbalanced data sets that is well, yet not perfectly, suited to represent the information of a confusion matrix within a single measure [7].

The results on the StockNet data set [43] are depicted in Table 3.2. Our model `SePP.Pipe` achieves a new state of the art in accuracy, even though it does not consider pricing information as input in contrast to [24, 43, 42]. As pointed out in Section 2.2, we are solving a more difficult task. `SePP.Int` also achieves significantly better results than the best-performing approach with the same inputs, HAN [19]. However, as the values for MCC show, our models yield less balanced results.

[11] `https://huggingface.co/transformers/`
[12] `https://torchmetrics.readthedocs.io/en/latest/`
[13] `https://pandas.pydata.org`
[14] `https://wandb.ai/`

Model	**Accuracy**	**MCC**
Random (*theoretical result*)	50.00	0.000000
ARIMA [9]	51.39	-0.020588
RandForest [33]	53.08	0.012929
HAN [19]	57.64	0.051800
MFNN [24]	57.86	0.064324
Stocknet [43]	58.23	0.080796
SMPN.CONCAT [42]	58.55	<u>0.158640</u>
Adv-ALSTM [14]	59.12	0.113421
SePP.Int (ours)	59.71	0.077120
SMPN.PLUS [42]	59.74	0.129798
SePP.Pipe (ours)	<u>61.24</u>	0.064930

Table 3.2: Comparison of results on the StockNet data set [43]. All values except for the random baseline and our approach are taken from [42]. Underlined values denote best results.

4 Discussion

This chapter discusses the results we obtained with respect to both real-world application and theoretical consequences.

4.1 Real-world application

An obvious question that poses itself is how our model can be used in the real world. The approach we presented makes a prediction about whether the price of an asset will go up or down on a certain day. Let us call the probability by which a model correctly predicts the movement p. This is equivalent to the notion of *accuracy* as defined in Section 3.4. As we have reported, our models would thereby obtain $p \approx 0.6$.

In a world with no transaction costs and taxes, an algorithmic trading agent could consistently generate profits using a model with $p > 0.5$, but only if it does not exhibit systematic errors. Consider the following demonstration. Let us presume an agent uses a simple algorithm where K^a is the number of units of asset a we have in our depot, and K_{max} is the maximum number of units we are willing to buy of a. At each time step $t-1$,

- If $\hat{d}_t^a = 1$ (i.e., the model tells us that the price is going up tomorrow), ensure that $K^a = K_{max}$, meaning that if we currently do not have a in our depot, *buy* K_{max} units, else, do nothing (keep the existing ones).
- If $\hat{d}_t^a = 0$ (i.e., the model tells us that the price is going down tomorrow), ensure that $K^a = 0$, meaning that if we currently have a in our depot, *sell* it all, else, do nothing.

Table 4.1 illustrates how a hypothetical agent with the very high direction prediction probability of $p = 0.75$ could still end up with overall losses if the returns are not distributed symmetrically.

Presuming returns were symmetrically distributed, another issue occurring in the real world are transaction costs. Adding them to the equation renders

t	**Observed** r_t^a	**Predicted** $\hat{d}_t^a$	**Prediction**	**Realised**
1	-1.0%	0 (down)	Correct	0.0%
2	+0.5%	1 (up)	Correct	+0.5%
3	+0.5%	1 (up)	Correct	+0.5%
4	-4.0%	1 (up)	Incorrect	-4.0%
Total	-	-	75% correct	-3.0%

Table 4.1: This example illustrates how a model that only predicts the direction of the price can result in negative realised profits, even if it has high prediction accuracy. While this hypothetical model correctly predicts the direction of the price movements on the first three days, it fails to do so on the last day. As the movement is larger on that day, it outweighs profits realised before, resulting in an overall loss.

making profits with this agent even more challenging. It would be necessary to know the magnitude of the price change to decide whether or not it is worth the transaction costs selling or buying incurs.

Lastly, this approach could also prove to be susceptible to manipulation when used in the real world. [11] found that, already today, there exists a significant amount of spam generated by bots to affect automated agents. Suppose our models were directly applied in the real world without additional protection. In that case, spammers could generate fake messages to trick it into picking up the sentiment and deriving a decision from it. If such agents were used on a large scale, this could become a potentially dangerous tool to manipulate stock prices.

In conclusion, it becomes apparent that this model is not suitable to be directly used for automated trading. However, that was not its intended purpose. The following section discusses the theoretical conclusions we can draw from the results.

4.2 Consequences

In a semistrong-form efficient market according to the EMH, all relevant, publicly available information should be reflected in the price. As our and comparable models show, this does not seem to be entirely the case for the sample that we have investigated. Using publicly available tweet information, we were able to predict the future development of stock prices to a statistically significant, albeit seemingly minor degree of accuracy compared to a baseline of random

guessing. Still, even accuracies only slightly higher than 50% can have significant value in real markets. We also showed that past price information is not necessary to reach an accuracy of around 60% on the StockNet data set [43] and that the tweet sentiments are informative enough.

Still, this result is in line with Burton Malkiel's conclusion from 2003, stating that predictable patterns in stock returns can and will appear over time [27]. The fact that we have found one does not disprove the EMH, as the time frame is too short and the sample size is much too limited. In fact, we found it remarkable how hard it was to train a model on this data set in terms of stability and parameter sensitivity, indicating that the task we tackled was very difficult to learn, even for powerful models.

In practice, we do not expect that stock price prediction using machine learning models will make a big difference for private investors. Large language models are very resource-intensive; hence, they are not easily accessible, and if they prove capable of generating alpha, they will quickly be used by professional investors. However, as we have seen, mispricings do exist after all, and ML models might help in the future to detect them more quickly.

4.3 Future work

This section briefly details potential future research directions. First, model performance could likely be improved by reconsidering the selection of $\mathcal{T}$, i.e., which tweets are used for the prediction. Filters and a more advanced selection of terms related to stock prices could solidify the basis on which the sentiment is detected. Further, it might be beneficial to also feed in additional metadata, such as dates or the number of followers, likes or retweets. Recent methods incorporate additional modalities using higher-dimensional input embeddings [17]. Also, novel Transformer approaches that can deal with larger inputs might help analysing more potentially relevant information [20, 41, 45].

If the goal were to optimise our model's performance as an algorithmic trading agent, it would be beneficial to extend it such that it outputs its confidence for its predictions and then program the agent only to sell or buy if a certain level of confidence is given. Such an approach could save transaction costs and be useful to accommodate for prices staying the same.

Finally, another line of work might focus on more systematically analysing the

performance of such models on real-world data from Stocktwits[1], Reddit[2] or financial news outlets. The real-world applicability of such approaches could be investigated more thoroughly by computing the actual returns an automatic trading agent using these models could achieve. Doing so might allow for more general conclusions about the degree to which mispricings can be detected using sentiment analysis.

[1] https://stocktwits.com
[2] https://www.reddit.com/r/stocks/

5 Conclusion

In this thesis project, we presented a novel approach for stock market prediction using machine learning. Our neural network, SePP, receives as input tweets and predicts whether the stock price will go up or down in the next time step, based on the sentiments it detects in the tweets. To do so, it uses a pre-trained sentiment detection module based on the Transformer architecture [40] and custom-built components that we trained on the StockNet data set [43].

We achieved a new state-of-the-art result in accuracy on the data set and showed that it is not necessary to incorporate price information for that, as tweet sentiment is already informative. Hence, we found a pattern that is predictable for at least the time frame of the data set. While this study does not allow for general conclusions about the validity of market efficiency, it demonstrated how modern language models can be used to detect and exploit small patterns of publicly available information that are not yet priced in.

Appendix

Appendix A: Acronyms

AI Artificial Intelligence

ARMA Autoregressive Moving Average

ARIMA Autoregressive Integrated Moving Average

BCE Binary Cross Entropy

DL Deep Learning

EMH Efficient Market Hypothesis

GPU Graphics Processing Unit

HAN Hybrid Attention Networks

LR Learning Rate

MCC Matthews Correlation Coefficient

MFNN Multi-Filters Neural Networks

ML Machine Learning

MLP Multi-layer Perceptron

MSE Mean Squared Error

NLP Natural Language Processing

ReLU Rectified Linear Unit

Bibliography

[1] Shane (https://stats.stackexchange.com/users/5/shane).
The Two Cultures: statistics vs. machine learning? Cross Validated.
URL:https://stats.stackexchange.com/q/6 (version: 2017-04-08).
eprint: https://stats.stackexchange.com/q/6.
URL: https://stats.stackexchange.com/q/6 (cit. on p. 4).

[2] Malcolm Baker and Jeffrey Wurgler.
"Investor sentiment in the stock market".
In: *Journal of economic perspectives* 21.2 (2007), pp. 129–152 (cit. on pp. 1, 6).

[3] Francesco Barbieri et al. "Tweeteval: Unified benchmark and comparative evaluation for tweet classification".
In: *arXiv preprint arXiv:2010.12421* (2020) (cit. on pp. 10, 14).

[4] Lukas Biewald. *Experiment Tracking with Weights and Biases*.
Software available from wandb.com. 2020. URL: https://www.wandb.com/
(cit. on p. 19).

[5] Christopher M. Bishop. "Pattern Recognition and Machine Learning (Information Science and Statistics)". In: 2006 (cit. on p. 12).

[6] Johan Bollen, Huina Mao, and Xiaojun Zeng.
"Twitter mood predicts the stock market".
In: *Journal of computational science* 2.1 (2011), pp. 1–8 (cit. on p. 1).

[7] Sabri Boughorbel, Fethi Jarray, and Mohammed El-Anbari.
"Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric". In: *PLoS ONE* 12 (2017) (cit. on p. 19).

[8] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32 (cit. on p. 7).

[9] Robert Goodell Brown.
Smoothing, forecasting and prediction of discrete time series.
Courier Corporation, 2004 (cit. on pp. 6, 7, 20).

[10] James Chen. *Alpha*. 2021.
URL: https://www.investopedia.com/terms/a/alpha.asp (accessed on 10/24/2021) (cit. on p. 5).

[11] Stefano Cresci et al. "Cashtag piggybacking: Uncovering spam and bot activity in stock microblogs on Twitter".
In: *ACM Transactions on the Web (TWEB)* 13.2 (2019), pp. 1–27 (cit. on p. 22).

[12] Dorottya Demszky et al.
"GoEmotions: A Dataset of Fine-Grained Emotions". In: *58th Annual Meeting of the Association for Computational Linguistics (ACL)*. 2020 (cit. on p. 10).

[13] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *NAACL*. 2019 (cit. on pp. 8, 14).

[14] Fuli Feng et al.
"Enhancing Stock Movement Prediction with Adversarial Training".
In: *IJCAI*. 2019 (cit. on pp. 7, 20).

[15] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville.
"Deep Learning". In: *Nature* 521 (2015), pp. 436–444 (cit. on p. 4).

[16] Sanford J Grossman and Joseph E Stiglitz.
"On the impossibility of informationally efficient markets".
In: *The American economic review* 70.3 (1980), pp. 393–408 (cit. on p. 6).

[17] Jonathan Herzig et al.
"TaPas: Weakly Supervised Table Parsing via Pre-training".
In: *ArXiv* abs/2004.02349 (2020) (cit. on p. 23).

[18] Robert Mark Hierons. "Machine learning. Tom M. Mitchell. Published by McGraw-Hill, Maidenhead, U.K., International Student Edition, 1997. ISBN: 0-07-115467-1, 414 pages. Price: U.K. £22.99, soft cover."
In: *Software Testing, Verification & Reliability* 9 (1999), pp. 191–193 (cit. on p. 3).

[19] Ziniu Hu et al. "Listening to Chaotic Whispers: A Deep Learning Framework for News-oriented Stock Trend Prediction".
In: *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (2018) (cit. on pp. 7, 8, 19, 20).

[20] Andrew Jaegle et al.
"Perceiver IO: A General Architecture for Structured Inputs & Outputs".
In: *arXiv preprint arXiv:2107.14795* (2021) (cit. on p. 23).

[21] Diederik P. Kingma and Jimmy Ba.
"Adam: A Method for Stochastic Optimization".
In: *CoRR* abs/1412.6980 (2015) (cit. on p. 5).

[22] Zhenzhong Lan et al. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations". In: *ArXiv* abs/1909.11942 (2020) (cit. on pp. 8, 14).

[23] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *ArXiv* abs/1907.11692 (2019) (cit. on pp. 8, 13, 14, 17).

[24] Wen Long, Zhichen Lu, and Lingxiao Cui. "Deep learning-based feature engineering for stock price movement prediction". In: *Knowledge-Based Systems* 164 (2019), pp. 163–173 (cit. on pp. 7, 19, 20).

[25] Ilya Loshchilov and Frank Hutter. "Decoupled Weight Decay Regularization". In: *ICLR*. 2019 (cit. on pp. 5, 18).

[26] C Majaski. *Fundamental vs. Technical Analysis: What's the Difference?* 2021. URL: https://www.investopedia.com/ask/answers/difference-between-fundamental-and-technical-analysis/ (accessed on 10/20/2021) (cit. on p. 5).

[27] Burton G Malkiel. "The efficient market hypothesis and its critics". In: *Journal of economic perspectives* 17.1 (2003), pp. 59–82 (cit. on pp. 1, 6, 23).

[28] Christopher Manning and Hinrich Schutze. *Foundations of statistical natural language processing*. MIT press, 1999 (cit. on p. 7).

[29] J.B. Maverick. *The Weak, Strong, and Semi-Strong Efficient Market Hypotheses*. 2021. URL: https://www.investopedia.com/ask/answers/032615/what-are-differences-between-weak-strong-and-semistrong-versions-efficient-market-hypothesis.asp (accessed on 10/24/2021) (cit. on p. 6).

[30] Tomas Mikolov et al. "Efficient Estimation of Word Representations in Vector Space". In: *ICLR*. 2013 (cit. on p. 7).

[31] Anshul Mittal and Arpit Goel. "Stock prediction using twitter sentiment analysis". In: *Standford University, CS229 (2011 http://cs229. stanford. edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis. pdf)* 15 (2012) (cit. on p. 1).

[32] Usman Naseem et al. "Transformer based deep intelligent contextual embedding for twitter sentiment analysis". In: *Future Generation Computer Systems* 113 (2020), pp. 58–69 (cit. on p. 1).

[33] Venkata Sasank Pagolu et al. "Sentiment analysis of Twitter data for predicting stock market movements".
In: *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)* (2016), pp. 1345–1350 (cit. on pp. 7, 20).

[34] Sara Rosenthal, Noura Farra, and Preslav Nakov.
"SemEval-2017 Task 4: Sentiment Analysis in Twitter". In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.
Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 502–518. DOI: 10.18653/v1/S17-2088.
URL: https://aclanthology.org/S17-2088 (cit. on p. 10).

[35] Stuart J. Russell and Peter Norvig.
"Artificial Intelligence: A Modern Approach". In: 1995 (cit. on p. 3).

[36] Elvis Saravia et al. "CARER: Contextualized Affect Representations for Emotion Recognition". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.
Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 3687–3697. DOI: 10.18653/v1/D18-1404.
URL: https://www.aclweb.org/anthology/D18-1404 (cit. on p. 10).

[37] Dipanjan Sarkar, Raghav Bali, and Tushar Sharma.
"Practical Machine Learning with Python". In: *Apress*. 2018 (cit. on p. 4).

[38] Martin Sewell. "History of the efficient market hypothesis".
In: *Rn* 11.04 (2011), p. 04 (cit. on pp. 1, 6).

[39] Tim Smith. *Market Sentiment*. 2021.
URL: https://www.investopedia.com/terms/m/marketsentiment.asp (accessed on 10/24/2021) (cit. on p. 5).

[40] Ashish Vaswani et al. *Attention Is All You Need*. 2017.
arXiv: 1706.03762 [cs.CL] (cit. on pp. 1, 4, 7, 8, 25).

[41] Chuhan Wu et al. "Fastformer: Additive Attention Can Be All You Need".
In: *arXiv preprint arXiv:2108.09084* (2021) (cit. on p. 23).

[42] Hongfeng Xu et al. "Stock movement predictive network via incorporative attention mechanisms based on tweet and historical prices".
In: *Neurocomputing* 418 (2020), pp. 326–339 (cit. on pp. 7, 11, 19, 20).

[43] Yumo Xu and Shay B. Cohen.
"Stock Movement Prediction from Tweets and Historical Prices".
In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 1970–1979.

DOI: 10.18653/v1/P18-1183.
URL: https://www.aclweb.org/anthology/P18-1183
(cit. on pp. 2, 6, 7, 10, 11, 19, 20, 23, 25).

[44] Kaisheng Yao et al.
"Recurrent Neural Networks for Language Understanding". In:
Aug. 2013. DOI: 10.13140/2.1.2755.3285 (cit. on p. 7).

[45] Manzil Zaheer et al. "Big Bird: Transformers for Longer Sequences."
In: *NeurIPS*. 2020 (cit. on p. 23).