Karlsruhe Institute of Technology

Institute of Finance, Banking and Insurance

Chair of Financial Engineering and Derivates
Prof. Dr. Marliese Uhrig-Homburg

Bachelor thesis

Investor Sentiment on Twitter and Its Link to the Stock Market

Author: Jonas Rothfuss
Ludwig-Wilhelm-Str. 3
Karlsruhe, Germany
E-Mail: jonas.rothfuss@gmx.de

Karlsruhe, November 8th 2016

Contents

List of Figures

List of Tables

List of Abbreviations

AAPL	Stock Ticker Apple Inc.
CEFD	Closed-End Fund Discounts
DJIA	Dow Jones Industrial Average
EMH	Efficient Market Hypothesis
GS	Stock ticker Goldman Sachs Group, Inc.
GPU	Graphic Processing Unit
LIWC	Linguistic Inquirer and Word Count
LSA	Latent Semantic Analysis
LSTM	Long Short-Term Memory
MSFT	Stock ticker Microsoft Corporation
n-gram	sequence of n items/words in a text
NB	Naive Bayes (Classifier)
NLP	Natural Language Processing
OLS	Ordinary Least Squares
PMI	Point Wise Mutual Information
POS	Part of Speech Tagging
RNN	Recursive Neural Network
SGD	Stochastic Gradient Descent
SST	Stanford Sentiment Treebank
SVM	Support Vector Machine
SWN	SentiWordNet
tf-idf	term frequencyinverse document frequency
VADER	Valcence Aware Dictionary for Sentiment Reasoning

1 Introduction

The question whether stock markets can be predicted to certain degree has always been a heavily discussed issue in academia and business. Early research in this area was was strongly influenced by the Random Walk Theory (Fama, 1965a; Cootner, 1964) and the Efficient Market Hypothesis (EMH) (Fama, 1965b; Fama et al., 1969). Core idea of the EMH is that asset prices embody all available information and price changes are driven by new information. The Random Walk Theory states that stock market prices follow a random walk and thus cannot be predicted. Both theories are compatible and imply that it is not possible to consistently generate risk-adjusted profits that are higher than the average market return. However, various empirical studies such as Butler and Malaikah (1992), Gallagher and Taylor (2002), and Qian and Rasheed (2007) show that stock market prices are predictable to a certain degree, thereby questioning the Efficient Market Hypothesis. Also the the EMH can barely explain stock market anomalies that seem to be unjustified given available information.

What Keynes referred to as "animal spirits" in 1936, is subject of today's behavioral finance since the "potential role of investor sentiment in financial markets has received considerable attention from economists" (Karabulut, 2013). Studies in behavioral economics and psychology showed that not only facts and information, but also emotions and mood influence financial decision-making (Nofsinger, 2005; Rick and Loewenstein, 2008). Therefore, it is reasonable to assume that sentiment as expressed in media like news or online blogs might be related to stock prices.

I pose the question how investor sentiment, expressed in Tweets that are specifically related to stocks, interacts with stock markets. Using two million Tweets, I analyze whether relevant sentiment can be extracted and examine its link to stock returns. Specifically, it is investigated if the sentiment of equity Tweets reflects past stock returns or predicts stock price movements in the future.

In a first approach, sentiment is extracted with a lexicon-based sentiment analysis method (i.e. VADER), resulting in a sentiment score for each Tweet. OLS regression is then used to investigate how these sentiment scores interact with past and future

stock returns in a time range of one hour. I find stock returns preceding a Tweet to be positively correlated with its sentiment, whereas negative correlation is found for stock-returns succeeding a Tweet by approximately 30 minutes. Results suggest that the sentiment of equity Tweets follows investor sentiment with a time delay, reflecting stock returns in the past and predicting a reversion of the previous stock price movement in the near future. In a further step, a short-term trading strategy based on the sentiment scores is implemented and evaluated. I find that betting on a stock price reversion in response to strong sentiment polarity in equity Tweets yields significantly higher returns than a naive randomized trading strategy.

With regard to stock market movements on a daily basis, I combine the sentiment scores of all Tweets throughout a day to a daily sentiment value. A Granger causality analysis is then used to investigate the hypothesis that investor sentiment, as measured by VADER, is predictive of changes in stock market closing values. Regarding this, no evidence for an interaction of Tweet sentiment and daily stock returns is found.

Since machine learning methods were shown to outperform simple lexicon approaches in many applications of sentiment analysis, I examine the ability of machine learning models to predict stock markets based on the collected Tweets. First, a base-line sentiment analysis model with word occurrence counts as features and common machine learning classifiers is applied. Finally, I train and validate a state-of-the-art deep learning model for sentiment classification, aiming to predict stock markets based on the Tweets. Against expectations, all machine learning approaches that were implemented and tested perform significantly worse than VADER in terms of stock return prediction based on equity Tweets. Results indicate that machine learning methods can hardly learn meaningful sentiment features and patterns due to the fuzziness of the underlying stock market data.

The theoretical foundation for the impact of investor sentiment on stock markets was introduced by De Long et al. (1990). In an effort to explain wild stock market movements that seem to be unjustified by fundamentals[1], De Long et al. propose a theoretical model

[1] Tetlock, 2007.

attempting to explain why stock prices often diverge significantly from fundamental values. Core assumption of their theory is the existence of two types of traders - irrational noise traders who have random beliefs about the value of a certain stock and rational arbitrageurs. Furthermore, both investor types are assumed to be risk averse. They show the existence of an equilibrium in which stock prices are influenced by the noise traders' random beliefs. For instance, if investor sentiment is negative, meaning that the noise traders's averaged beliefs are more pessimistic than the arbitrageurs' rational belief, noise traders sell stocks to arbitrageurs and thereby temporarily depress stock returns. As the stock price significantly drops under its fundamental value, rational arbitrageurs increasingly start to buy the respective stocks, inducing a reversion back to the fundamental value. However, the volatility of noise traders' beliefs creates additional risk that deters arbitrageurs from aggressively betting against them, thus impeding an efficient price reversion.

There is a growing number of empirical studies that find a linkage between investor sentiment and financial markets. Beside indirect, market based proxies for investor sentiment such as closed-end fund discounts, dividend premiums and number of IPOs (Lee et al., 1991; Baker and Wurgler, 2006), there is an increasing number of studies in empirical finance that measure investor sentiment in texts (e.g. news, blogs), using techniques known from natural language processing.

Antweiler and Frank (2004) attempt to measure investor sentiment based on internet stock message boards by extracting whether the messages tend to contain a "buy", "sell" or "hold" trading recommendation. Although they find evidence that the stock messages help predict market volatility, they do not find statistically significant results that bullish or pessimistic messages can forecast stock returns. Tetlock (2007) and Garcia (2013) conducted studies with regard to investor sentiment based on columns of the Wall Street Journal and New York Times, respectively. Both find that pessimism, indicated by predominantly negative words in the columns, predicts downward pressure on the Dow Jones Industrial Average (DJIA). Important to note is the fact that statistically significant predictability of stock returns using news paper content could just be shown for downward

trends, not for positive stock market trends.

Recent work focuses on creating a measure of public mood based on content in social media and relating it to stock market indices. Bollen et al. (2011) analyze the text of daily Twitter feeds by measuring the overall sentiment polarity as well as 6 dimensions of mood. First and foremost, they find that especially the mood dimension "Calm" is Granger-causative to DJIA values. While the Twitter mood dimension "Happy" does not significantly contribute to stock market predictions, Karabulut (2013) shows that Facebook's Gross National Happiness indicator is able to forecast daily US stock market returns. Both studies use public mood extracted from the entire corpus of social media (i.e. Twitter and Facebook) as proxy for investor sentiment on a daily basis. Results are in accordance with Hirshleifer and Shumway (2003) who find that sunny weather, positively affecting public mood, is correlated with daily stock returns.

However, how social media content that specifically refers to financial markets is related to stock price changes has not yet been addressed in the literature. In general, the majority of empirical work in this field analyzes the relationship of investor sentiment and stock returns on a daily basis or considers even longer timespans. Shorter time frames on a sub-day level have only received little attention in financial literature so far. To my knowledge, this is the first study that considers timespans of less than one hour for interaction of investor sentiment in media and stock markets. The results obtained for the analysis of short-term relationships between Tweet sentiment and stock prices tell a consistent story and are in accordance with the noise trader theory proposed by De Long et al. (1990).

Since I use modern techniques of natural language processing and machine leaning some readers with background in finance might not be acquainted with, Section 2 provides an outline of common approaches and recent developments in sentiment analysis. Section 3 reports how the Tweets were collected, preprocessed and labelled. In Section 4, I describe the lexicon-based approach that is used to extract sentiment from the Tweets and explain the analytical steps, performed to investigate the relationship to short-term stock movement and daily stock returns. Section 5 gives an introduction to the machine

learning models that are applied, followed by a comprehensive assessment of the models' ability to predict stock market returns. Finally, I conclude in Section 6 by discussing the results and suggesting future research on the influence of investor sentiment in stock markets.

2 Sentiment Analysis

Sentiment Analysis is considered as one of the major task of Natural Language Processing (NLP). Whilst the problem formulation in other fields of NLP such as part-of-speech tagging is relatively clear, sentiment analysis is a broader category of tasks consisting of multiple problem dimension.

One basic problem dimension with respect to sentiment is the analysis of polarity: "Given an opinionated piece of text, wherein it is assumed that the overall opinion in it is about one single issue or item, classify the opinion as falling under one of two opposing sentiment polarities, or locate its position on the continuum between these two polarities" (Pang and Lee, 2008). Most common is the so called sentiment polarity classification which is a binary classification that distinguishes between "positive" and "negative" sentiment. Early research conducted in this area mainly focuses on pieces of text that clearly express the author's subjective opinion on a specific entity. Typical problem sets with regard to sentiment polarity are reviews. One of the most discussed tasks in research is the classification of movie review sentiment as initially addressed in Pang et al. (2002) and Turney (2002). Aside from binary polarity classification, there has been research capturing sentiment on a multi-way scale (Snyder and Barzilay, 2007) and fine-grained sentiment classification with three or more classes (Pang and Lee, 2005; Socher et al., 2013).

Another problem dimension of sentiment analysis is to distinguish between subjective and objective text. Whether the author only states facts or expresses his/her subjective opinion within a piece of text interferes with sentiment polarity detection. Subjective texts tend to contain more attributive verbs and adjectives (e.g. to adore, to hate,

terrible, fantastic, good, bad) which indicate the underlying sentiment. Thus, deciding on sentiment polarity given a subjective statement is usually easier in comparison to objective text. Pang and Lee (2004) even showed that removing objective passages and sentences from a text when determining its sentiment polarity can improve the performance. However, text data for sentiment classification does not always have to be strongly opinionated. News can be considered as good or bad without being subjective (Pang and Lee, 2008). By just mirroring facts like "IPhone sales increased by 25% in the first quarter" a piece of news can be positive or negative for an underlying entity such as the company or the respective stock. Hence, classifying equity news according to its impact on stock prices has been considered sentiment classification in the literature as well (Koppel and Shtrimberg, 2006).

Sentiment analysis varies in its scope. Tasks within the domain range from document level, such as deciding on the positivity/negativity of web reviews consisting of multiple paragraphs or sentences to phrase level sentiment, e.g. short comments in social media. Due to the variety in scope, different sentiment related tasks might require very different approaches, especially with regard to the features used for the analysis. According to Asghar et al. (2014) there are three types of morphological features: semantic, syntactic and lexico structural. Semantic features are based on contextual information or semantic orientation. Typical unsupervised methods which are used to determine the semantic orientation of words and phrases are Latent Semantic Analysis (LSA) and Point Wise Mutual Information (PMI) (Turney, 2002; Turney and Littman, 2003). Syntax type features use NLP tools such as Part of Speech Tagging (POS), n-grams and word dependencies. Lexico structural features provide statistical information about a text such as word distribution and special symbol frequencies.

Traditional sentiment analysis approaches use so called Bag-of-Word models where text is represented as multiset of its words. As these models are indifferent to word order, syntactic features and semantic compositionality are neglected entirely. Often Bag-of-Word models are used in combination with sentiment lexicons that list relevant words with their respective sentiment polarity.

While sentiment analysis, solely based on semantic orientation of words and their frequency within a text, works well for longer documents, short texts usually require more sophisticated approaches that combine multiple types of features. An explanation is that attributive verbs and adjectives, as well a nouns that contain information on sentiment polarity occur in a certain frequency within a text. Thus, longer pieces of text usually contain more such words and phrases. When accumulating the information on semantic orientation throughout a longer document, the sentiment polarity can be predicted relatively accurate (e.g. 74% accuracy on reviews in Turney (2002)). However, short pieces of text such as a single sentences do not contain enough words that are rich of information on sentiment polarity when viewed isolated. Instead, it is necessary to capture the interaction of semantic as well as syntactical features; in particular, how words and phrases are arranged within an expression and semantically interact.

A challenge in NLP is to craft features that represent the characteristics of natural language (i.e. syntax and semantic) and at the same time are convenient to process and analyse mathematically. An approach that gained huge popularity in NLP are vector space models, initially introduced by Salton et al. (1975). Vector space models represent words as vectors in a continuous space (i.e. $\mathbb{R}^n$). Semantically related words are located in the same region of the vector space. Although a multitude of methods to generate semantic vector spaces have been introduced and utilized, they all "depend in some way or another on the Distributional Hypothesis, which states that words that appear in the same contexts share semantic meaning"[2]. A big advantage of semantic vector spaces when compared with other features is the mathematical convenience. Text, represented as numerical vector can be directly used as input for many machine learning methods such as deep learning.

@marketstocknews: RT TheStreet: Apple has a new $780 million annual revenue stream – and it's growing https://t.co/ELUZ9lX5OC $AAPL
Heterkuria95: RT LNPServices: #Milestone: #Bing Now Profitable As #Windows10 Success Boosts Usage - https://t.co/apdq6NBjeB - $MSFT #Tech
Best stocks of September 2015: $cmcsa $etfc $btu $gme $fitb $ge https://t.co/zhJQdBUbqO $$ #stocks
Will Disney Earnings Crash $DIS Stock Again? https://t.co/LwL3wiecdb
I don't even like/use #windows, but I like the stock $msft https://t.co/0PMn5rfZwG

Table 2
Five examples selected from the stock related Tweets that were collected between October 2015 and September 2016

3 Data

3.1 Retrieval

Over a time period of 10 month (Oct 21, 2015 to Aug 23, 2016) 2,011,944 Tweets were collected via the TwitterAPI. All of the Tweets are written in English language and contain at least one ticker symbol of the 30 firms in the Dow Jones Industrial Average (e.g. $AAPL). Table 2 shows five examples from the more than 2 million Tweets that were gathered. 45% of these Tweets are so called Re-Tweets thus not providing any new text or content. More than 85% of tweets only contain one stock ticker. However, the amount of tweets referring a specific stock varies significantly. While more than half of the tweets refer to Apple of Microsoft, firms like United Technology Corporation or United Health Group get only little attention on Twitter. Figure 1 illustrates how the stock mentions throughout the collected tweets are distributed among the most popular firms.

In order to validate models and trading strategies, the retrieved data is split into two parts. The partion of Tweets that were posted between Oct 21, 2015 and Jul 10, 2016 is used for analysis and selection of models. To avoid overfitting and correctly assess the performance of trading strategies that were developed based on the first part of data, all Tweets from Jul 11 to Aug 23, 2016 are put aside and solely used for validation.

[2] https://www.tensorflow.org/versions/r0.11/tutorials/word2vec/index.html; Aug 2016

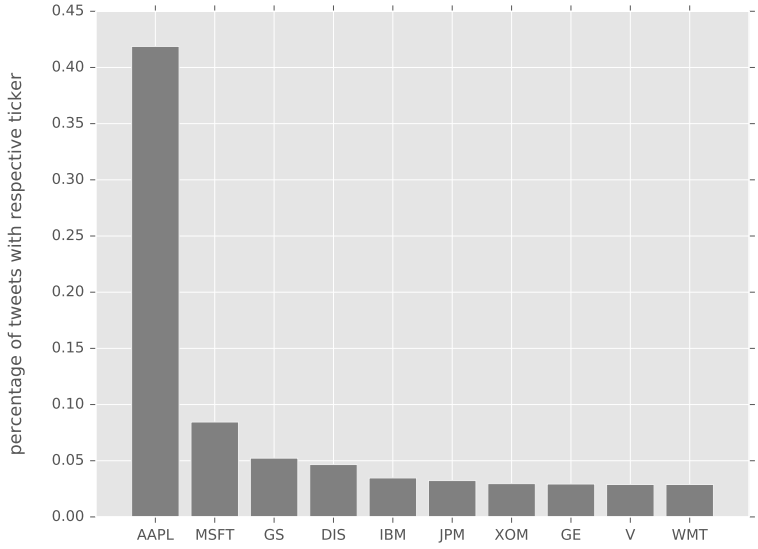


Figure 1
Percentage of collected tweets that contain a respective stock ticker. Displayed are only 8 of 30 tickers with the highest occurance within the tweets.

3.2 Preprocessing and cleaning

Before sentiment analysis or other NLP tasks are performed it is important to preprocess and clean the text data. First, this includes removing data that is duplicate, corrupted or not relevant to the task. Thus, I remove all direct re-tweets as they a are duplicates of tweets that have already been posted. Second, transforming and cleaning the text to unify the format and reduce unnecessary variance within the text can significantly improve the performance of NLP procedures. Main goal is to keep the required vocabulary as small as possible without compromising too much information. With regard to this, I perform following steps:

- transform the text to lower case
- replace hyperlinks with the generic term 'url'
- replace user mentions such as '@celine22' with 'user'
- limit question and exclamation marks to two in a row
- delete the '#' sign in front of hash tags

- replace stock tickers with the corresponding company name
  e.g. '$MSFT' $\longrightarrow$ 'microsoft'
- remove multiple-dot punctuation; e.g. '....' $\longrightarrow$ '.'

3.3 Labelling

For tweets that were posted within the New York Stock Exchange trading hours (14:30 UTC to 21:00 UTC) I calculate different short-term stock price returns $r_{\Delta t}$

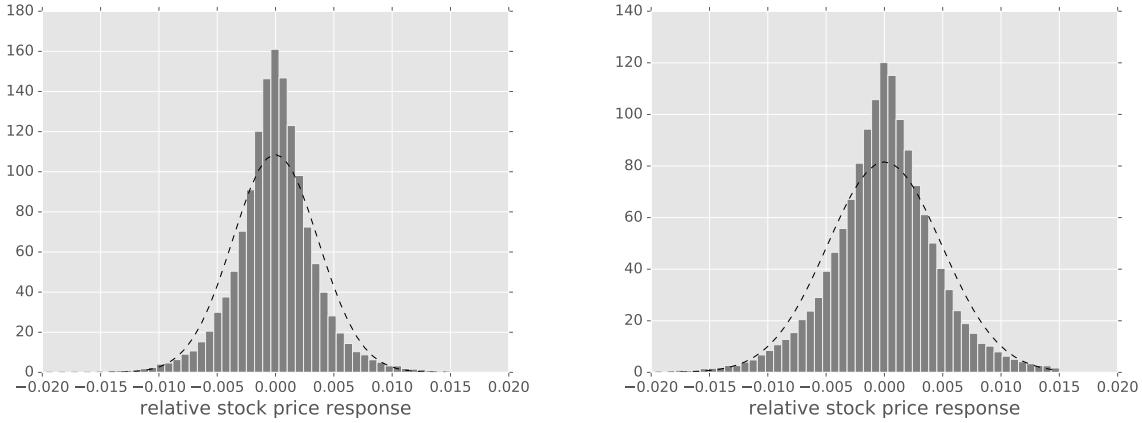
$$r_{\Delta t} = \frac{p(t_{Tweet} + \Delta t)}{p(t_{Tweet})} - 1 \tag{1}$$

where t_{Tweet} denotes the time at which the tweet was posted, Δt the corresponding time lag and $p(t)$ the corresponding stock price at time t. Specifically I consider 1, 5, 10, 20, 30, 40 and 60 minute time lags. Due to computational burdens of machine learning I will put the main focus on 30 minute and 1 hour lags. The stock price responses are approximately normal distributed (Figure 2) with $\mu \approx 0$. As one would expect the 1 hour stock price responses have a higher variance than the 30 min price movements.

In section 4.1 stock returns, preceding a Tweet, are considered as well. In case the time difference Δt denotes a backward looking timespan (e.g. - 30 min), returns are defined as follows:

$$r_{-\Delta t} = \frac{p(t_{Tweet})}{p(t_{Tweet} - \Delta t)} - 1 \tag{2}$$

In order to perform sentiment classification with machine learning methods it is necessary create discrete labels for the tweets. Correspondent to the stock price responses $r_{30\text{min}}$ respectively $r_{1\text{h}}$, I assign one of the labels "negative", "neutral" and "positive" each. Tweets with a stock price response ranging among the 25% most negative price responses are assigned the label "negative", tweets with r among the 25% most positive price responses "positive" and the remaining tweets "neutral". The label assignment approach can be formally expressed through a mathematical map that utilizes the 25th and



(a) 30 min lag **(b)** 1 hour lag

Figure 2
Distribution of relative stock price response corresponding to a tweet a) 30 minutes b) 1 hour after the tweet has been posted and normal distribution (dashed line) with sample-estimated μ and σ

75th percentile of the price responses to distinguish among the three sentiment labels:

$$l(r) = \begin{cases} \text{"positive"} & \text{if } r \geq p_{75} \\ \text{"neutral"} & \text{if } p_{25} < r < p_{75} \\ \text{"negative"} & \text{if } r \leq p_{25} \end{cases} \tag{3}$$

Since these labels were created based on changes of the associated stock price within the next 30 min / 1 hour, the labels do not necessarily reflect text sentiment as known from natural language processing.

4 A Lexicon Based Approach - Valence Aware Dictionary for Sentiment Reasoning (VADER)

Many state-of-practice sentiment analysis benchmarks rely on sentiment lexica to supply features. In its core, these lexica contain lists of words associated with a sentiment label or score . Given a text, words listed in the lexicon are extracted from the text and annotated with their sentiment value, using the dictionary scores. Finally, the sentiment scores are aggregated into a single score or label for the entire text document (Taboada

et al., 2011).

Typical state-of-practice benchmarks in the domain of lexicon based sentiment analysis are the General Inquirer (Stone et al., 1966), Linguistic Inquiry and Word Count (LIWC) (Pennebaker et al., 2001; Pennebaker et al., 2007), as well as SentiWordNet (SWN) (Baccianella et al., 2010).

As traditional lexicon based sentiment models like the General Inquirer or LIWC were developed with focus on longer text documents, the nature of microblog content "poses serious challenges to practical applications of sentiment analysis. Some of these challenges stem from the [...] contextual sparseness resulting from shortness of the text and a tendency to use abbreviated language conventions to express sentiments" (Hutto and Gilbert, 2014). Addressing this issue, Hutto and Gilbert presented a rule-based sentiment model with underlying dictionary called VADER. They created a lexicon attuned for miroblog texts by aggregating lexical features from well-established, existing sentiment lexica and supplementing additional lexical features that are frequently used in social media (e.g. emoticons, acronyms etc.). In addition, they formulated five simple rules that "embody grammatical and syntactic conventions that humans use when expressing or emphasizing sentiment intensity" (ibid.).

When validating the 3-class classification performance of VADER in four distinct domain contexts (Social Media, Product Reviews, Movie Reviews and NY Times Editorials), it dominates all other established sentiment lexicon baselines by F1 score[3]. Especially when classifying Tweets, with a F1 score of 0.96 VADER significantly outperforms other lexicon based models (F1 score $\leq$ 0.77).

[3] Measure of a classifier's accuracy that is insensitive to imbalanced classes; for detailed explanation see section 5.1

4.1 VADER Sentiment and Its Linkage to Short-Term Stock Returns

As VADER appears to be the most suited lexicon based sentiment analysis tool for this purpose, I use VADER to generate a sentiment scores for each of the collected Tweets, attempting to predict the corresponding stock price responses. For generating VADER sentiment scores, the following steps are performed: First, the Tweets are split up into words and tokens. Then each word/token is looked up in the sentiment lexicon and if listed there, annotated according to its sentiment polarity. Negative, positive and neutral sentiment annotations are each aggregated into an intensity score from 0 to 1. Thereby, linguistic rules accounting for negation and sentiment intensity are incorporated. Then the three intensity scores are combined to a compound sentiment score ranging from -1 to 1. A negative score indicates negative sentiment polarity, whereas positive sentiment is suggested by positives scores.

To examine how VADER sentiment scores s_i are related to short-term stock price returns r_i within 1 hour time range after a Tweet i is published, I perform a linear regression with the following model:

$$r_i = \alpha + \beta * s_i + \epsilon_i \tag{4}$$

Classical linear regression assumes that the error terms ϵ_i are independent and identical distributed. However, I find that the stock returns r_i are (auto)correlated, thus strongly violating this assumption. To correctly assess the predictability of stock market returns through Twitter sentiment, a time series approach would be preferable. Unfortunately the time steps between the posting of Tweets are unevenly spaced, thus making analysis with time series approaches like ARIMA, or performing Granger-Causality very difficult. In order to account for dependence of stock returns in the setting of linear regression, the Tweets and corresponding stock responses are clustered by day. Within the daily clusters, correlation of the error terms ϵ_i is allowed, whereas independence among the daily clusters is assumed.

regressand	parameter	coef	std error	t	P> |t|
r_{1min}	Intercept	0.0355	0.036	0.997	0.320
	VADER score	-0.0396	0.070	-0.564	0.573
r_{5min}	Intercept	0.0138	0.119	0.116	0.908
	VADER score	-0.1138	0.149	-0.766	0.445
r_{10min}	Intercept	-0.0076	0.211	-0.036	0.971
	VADER score	-0.1312	0.213	-0.616	0.539
r_{20min}	Intercept	0.0821	0.383	0.214	0.831
	VADER score	-0.6013	0.311	-1.931	0.055
r_{30min}	Intercept	-0.0230	0.566	-0.041	0.968
	VADER score	-1.0613	0.434	-2.444	**0.016**
r_{40min}	Intercept	-0.0568	0.763	-0.074	0.941
	VADER score	-0.8941	0.476	-1.877	0.062
r_{1h}	Intercept	-0.0967	1.069	-0.091	0.928
	VADER score	-0.3257	0.570	-0.572	0.568

Table 3

OLS regression summary - Predicting short-term stock price responses with VADER sentiment scores: This table shows the OLS estimates of the coefficients a and β in equation 4 fitted on ca. 220,00 Tweets and corresponding Dow Jones stock data from Oct 2015 to Aug 2016 with daily error clusters. Regressand: stock return r_i (in basepoints) after a Tweet i is posted. Regressor: VADER compound sentiment score s_i of Tweet i

Surprisingly, OLS estimates of parameter β are negative, indicating a negative relationship between sentiment scores and short-term stock returns. I conduct a t-test (H_0: $\beta = 0$) to determine whether there is a significant linear relationship between Tweet sentiment and stock market. Table 3 holds the OLS estimates and t-test results.

Given a confidence level of 95%, the hypothesis that there is no linear relationship between VADER sentiment scores and stock market returns can be rejected for 30 min lags (p-value = 0.016). This observation is reinforced by the fact that 20 min and 40 min stock returns also show the same tendency (p-val: 0.055 and 0.062). In contrast, a p-value of 0.568 for 1 hour stock returns suggests that the effect observed for 30 minute lags becomes increasingly distorted by new information and general volatility of stock prices as more time passes by.

When considering the hypothesis that the measured sentiment reflects "noise traders beliefs relative to Bayesian beliefs" (Tetlock, 2007) of rational arbitrageurs as introduced in the theoretical work of De Long et al. (1990), it is crucial to take the temporal flow of causality into account. Either the measured VADER scores forecast investor sentiment

regressand	parameter	coef	std error	t	P> $\|t\|$
r_{-1min}	Intercept	0.0355	0.036	0.997	0.320
	VADER score	-0.0396	0.070	-0.564	0.573
r_{-5min}	Intercept	0.0567	0.159	0.356	0.722
	VADER score	0.5792	0.181	3.196	**0.002**
r_{-10min}	Intercept	0.0259	0.310	0.084	0.933
	VADER score	1.4632	0.276	5.304	**0.000**
r_{-20min}	Intercept	-0.2944	0.599	-0.492	0.624
	VADER score	2.9952	0.518	5.784	**0.000**
r_{-30min}	Intercept	-0.5945	0.850	-0.699	0.485
	VADER score	3.7423	0.616	6.074	**0.000**
r_{-1h}	Intercept	-1.2516	1.514	-0.826	0.410
	VADER Score	5.3584	1.009	5.311	**0.000**

Table 4

OLS regression summary - Linear estimation of past stock-returns with VADER sentiment scores: This table shows the OLS estimates of the coefficients a and β in equation 4 fitted on ca. 220,000 Tweets and corresponding Dow Jones stock data from Oct 2015 to Aug 2016 with daily error clusters. Regressand: stock return r_i (in basepoints) before a Tweet i is posted. Regressor: VADER compound sentiment score s_i of Tweet i

or they reflect past investor sentiment. If the former would be the case, one would expect to measure a positive correlation for stock-returns in the future. However, the findings in Table 3, i.e. negative correlation, indicate the opposite. In case the measured Tweet sentiment follows investor sentiment with a time-delay, one would expect a positive correlation of sentiment scores and stock returns in the near past. To further analyse the latter hypothesis, I perform an OLS regression with daily error clusters for stock returns within a 1 hour range before a respective Tweet is published. Figure 4 lists the OLS regression results for past stock returns. Indeed, stock returns that precede the publishing time of a Tweet by five minutes or more show a significantly positive correlation with its Vader score.

To sum up, future stock returns are shown to be negatively correlated with VADER sentiment scores, whereas positive correlation is found for past stock returns. This suggests that sentiment of equity Tweets follows past investor sentiment in such a way that Tweets reflect the cause of the current stock price deviation through their embedded sentiment and at the same time predict a price reversal in the near future (i.e. 30 min). A brief scenario shall elucidate this: According to De Long et al. (1990), noise traders

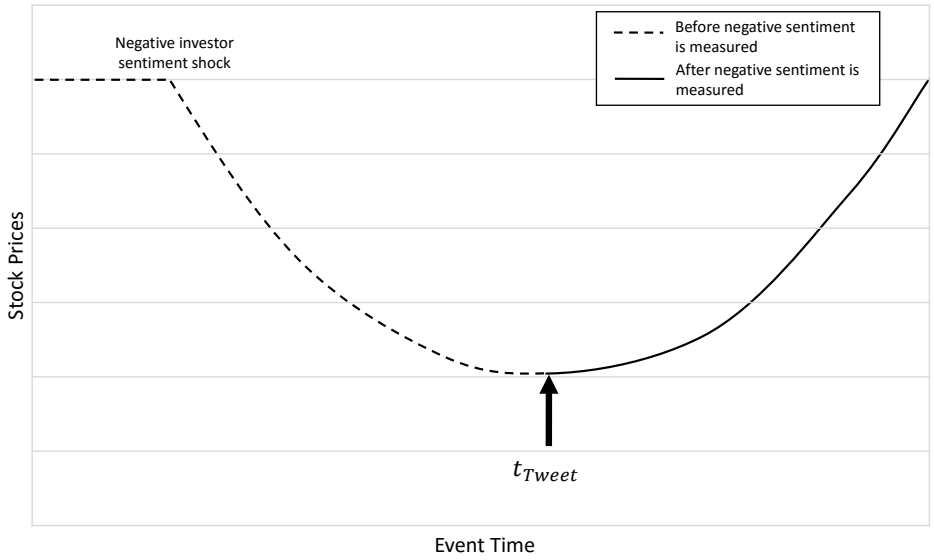


Figure 3

The illustration depicts the theoretical impact of a one-time increase in negative investor sentiment on stock prices. A negative investor sentiment shock is followed by a stock price decease as noise traders sell stocks. Rational arbitrageurs start to buy "under-priced" stocks thus triggering a reversal to the stock price backed by fundamental. Results indicate that sentiment extracted from Tweets follow investor sentiment with time-delay. (In style of Tetlock, 2007)

hold random beliefs about the fundamental value of a stock. If, by chance, the number of noise traders that have pessimistic beliefs about a stock significantly outrages the number noise traders that estimate the stock value higher than it actually is, the majority of noise traders tries to sell their stocks putting downward pressure on the stock price. I refer to this as a negative investor sentiment shock. Tweets reflecting the negative investor sentiment start to occur on Twitter with some time delay to the actual negative sentiment shock. Thus, the stock price has already been driven down by noise traders' pessimistic beliefs and rational arbitrageurs start to buy "under-priced" stocks, inducing a reversion back to the price backed by fundamentals. In this setting, stock returns preceding an equity Tweet, which reflects the pessimistic beliefs, are negative and therefore positively correlated with the negative sentiment score. Stock returns succeeding a Tweet are characterized by the stock price recovery. Hence, a negative correlation between returns in the near future and sentiment scores is observed. Figure 3 illustrates the scenario described. For positive investor sentiment shocks, the line of reasoning would be equivalent.

While the relationship of VADER scores and short-term stock returns in the past

is highly significant, a significant correlation of sentiment scores and short-term returns succeeding a Tweet was only found in the range of 20 to 40 minutes. There are two hypothesis explaining this phenomenon, though a combination of both is most likely. First, the unpredictability of noise traders' beliefs creates risk that deters rational arbitrageurs from aggressively betting against noise traders (De Long et al., 1990) and thereby impeding efficient reversal of prices to fundamentals. Second, new information affecting prices and changes in noise traders' beliefs over time increasingly distort the predicted price changes as more time passes.

Results suggest that equity Tweets with positive/negative sentiment predict a reversion of the stock price movement, observed before a respective Tweet was posted. To further validate these findings, a simple trading strategy based on a hand-crafted sentiment classifier is introduced and tested on excess returns. The classifier assigns each Tweet a sentiment label 'positive' when its compound sentiment score exceeds the positive threshold t and 'negative' when the score is negative and smaller than $-t$. Tweets with a score within the threshold boundaries $[-t, t]$ are classified as 'neutral'. The classifier can formulated mathematically as follows:

$$c(s) = \begin{cases} \text{"positive"} & \text{if } s \geq +t \\ \text{"neutral"} & \text{if } -t < s < +t \\ \text{"negative"} & \text{if } s \leq -t \end{cases} \tag{5}$$

Based on the the 3-class sentiment classification a simple trading strategy shall be assessed: If a Tweet is classified as 'negative' buy the corresponding stock and hold it for 30 min. If a Tweet is assigned the sentiment label 'positive' make a short sell and hold the position for 30 min. On the first glimpse, this trading strategy might seem counter intuitive. However, as a significant negative correlation was determined for 30 min lags, an inverted strategy that expects negative stock market movement in response to positive sentiment is justified. The estimated profit $\tilde{p}$ per trade when implementing the strategy for different classification thresholds t is shown in Figure 4.

To investigate the performance of this trading strategy I test against the hypothesis

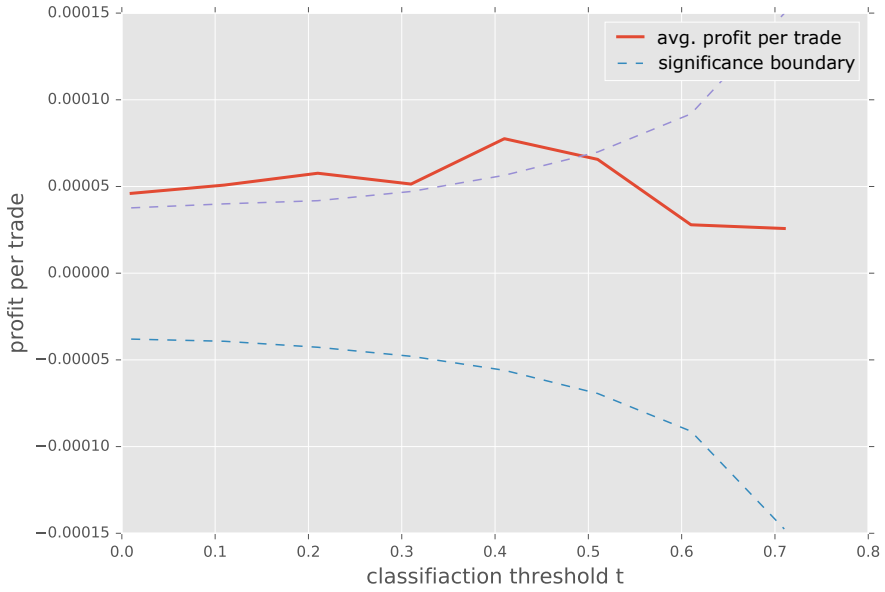


Figure 4
Estimated profit per trade when going short if VADER compound score $> t$ and long if sentiment score $< -t$; long/short position is hold for 30 min; significance boundary: 2.5% and 97.5% percentile of sampled mean profit distribution (50000 samples) corresponding to randomized trades

that the trading strategy cannot outperform a naive strategy that makes random decisions. The random trading strategy is implemented in such a way that it makes as many trades as the sentiment based strategy, but randomly decides when the trades are made and which trade action (buy/sell) is chosen each time. Building a statistical model to assess the trading strategy requires many assumption that may bias results. Hence, I simulate the strategy 50000 times to sample a distribution of mean profits. The 2.5% and 97.5% percentile of the sampled distributions with deceasing number of performed trades along the x-axis, in correspondence to the classification threshold t, is illustrated in Figure 4 as well. For all trading classification thresholds t between 0.01 and 0.5, the sentiment based trading strategy yields significant excess returns when compared to the naive randomized strategy. In fact, equity Tweets that are assigned sentiment scores greater than 0.5 occur very infrequent which would make a trading strategy with $t > 0.5$ impractical in the first place. Since t is the only hyper-parameter of this trading strategy, the fact that any arbitrary choice of $t \in (0, 0.5]$ in advance would have lead to a significant excess returns, when compared to the dummy strategy, suggests that the strategy could

have been implemented ex ante.

However, the hypothetical sentiment-based trading strategy neglects any costs. To implement the strategy, frequent portfolio turnovers are required, resulting in high costs for commission, bid-ask spreads, limited market liquidity and taxes. Therefore, it is questionable whether the sentiment-based strategy would be still profitable in practice when incorporating these costs.

4.2 Influence of Investor Sentiment on Daily Stock Price and Index Movements

After analyzing the linkage of Tweet sentiment and short-term stock price movements, I expand my analysis to longer time periods. Instead of assessing the sentiment of each Tweet individually, sentiment of Tweets posted within a day is aggregated. On each workday the mean of all VADER sentiment scores corresponding to Tweets that are published between 4 pm ET (NYSE closes) and 4 pm the following day is calculated. Mondays all Tweets over the weekend since Friday 4pm are incorporated in the mean score. Based on these aggregations a time series S_t of daily sentiment scores is established.

I am concerned with the questions if daily changes of investor sentiment on Twitter forecasts movements in the stock market. To address this question I apply the statistical method of Granger Causality Analysis to the sentiment time series, denoted S_t, versus the corresponding stock price and Dow Jones Industrial Average (DJIA). Granger analysis is a econometric technique used to determine if one time series can be used to forecast another. Since the concept of causality in this context is questionable, I use it in a fashion as Gilbert and Karahalios (2010) and Bollen et al. (2011) did in a comparable setting; not to actually test for causality but to examine if one time series contains predictive information about the other.

At first, I make use of all tweets when calculating S_t and conduct a Granger-Causality analysis with regard to the stock index DJIA. The DJIA time series D_t reflects daily close-to-close index changes. It is defined as $D_t = DJIA_t - DJIA_{t-1}$ where one time step corresponds to one day (except on Mondays). To test whether the sentiment time series

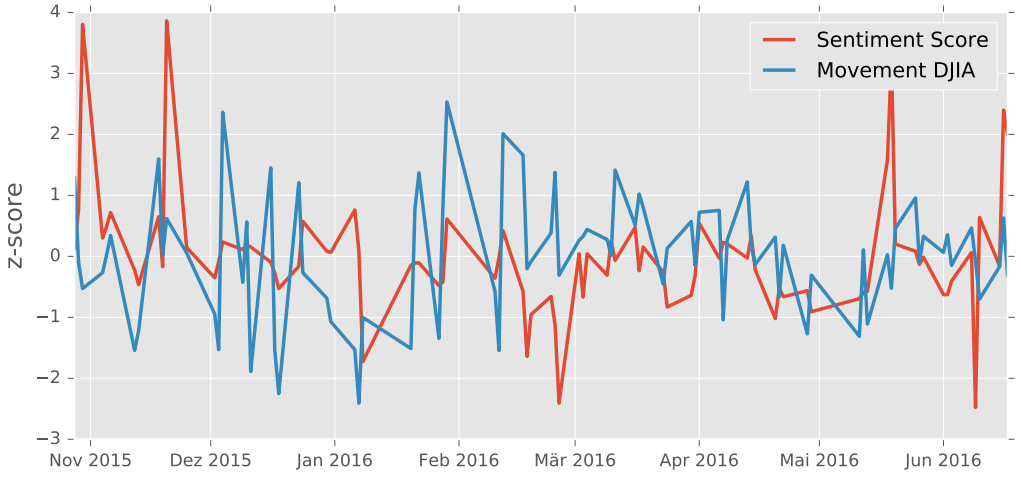


Figure 5
Overlap of day-to-day difference D_t of DJIA and daily mean sentiment score S_t. Due to differences in scale the z-scores of both time series are illustrated.

S_t is able predict how the DJIA changes from day to day, I fit two linear models, M_1 and M_2, with OLS and assess their variance explained. M_1 makes only use of lagged of D_t comparable to an univariate autoregression. The number of lags included in the model is determined by n. In addition to the first model, M_2 includes n lagged values of the sentiment time series S_t.

$$M_1 : D_t = \beta_0 + \sum_{i=1}^{n} \beta_i D_{t-i} + \epsilon_t \tag{6}$$

$$M_2 : D_t = \beta_0 + \sum_{i=1}^{n} \beta_i D_{t-i} + \sum_{i=1}^{n} \gamma_i S_{t-i} + \epsilon_t \tag{7}$$

Finally, the summed squares of residuals of both models are compared to determine if using lagged values of S_t significantly increases the variance explained. Using the F-statistic, I test against the null hypothesis $H_0 : \gamma_{\{1,..,n\}} = 0$. Based on the p-values listed in Table 5, the hypothesis that S_t is not able to forecast changes in the DJIA cannot be rejected.

As many firms in the DJIA receive only little attention on Twitter while other firms such as Apple, Microsoft and Goldman Sachs account for most of the equity Tweets, results might be biased due to the imbalance. Hence, I conduct the Granger-Causality

lag	DJIA	AAPL	MSFT	GS
1 day	0.5779	0.1755	0.2019	0.2146
2 days	0.8029	0.2005	0.4069	0.4429
3 days	0.7339	0.4324	0.2400	0.2400
4 days	0.7345	0.6273	0.7916	0.2920
5 days	0.6283	0.4492	0.8136	0.4330
6 days	0.6552	0.3658	0.8414	0.3077

Table 5
Statistical significance (p-values) of Granger-Causality correlation between daily sentiment score S_t and daily stock market changes D_t

analysis in similar fashion for the stock price movements of Apple, Microsoft and Goldman Sachs individually. In each case only Tweets containing the respective stock ticker are included in the sentiment scores S_t. The resulting p-values are illustrated in Table 5 as well. Since no statistical significance could be found, it is reasonable to assume that investor sentiment in equity Tweets extracted through VADER can barely forecast the stock market on a day-to-day basis.

Bollen et al. (2011) and Karabulut (2013) conducted studies with comparable methodologies. They also use the method of Granger-Causality analysis to investigate how social media content is related to the daily changes of the Dow Jones Industrial Average. Interestingly, both find a significant relationship between measures of public mood extracted from social media and the DJIA. The only remarkable difference in their approaches from the one, described in this section, is the scope of the underlying social media corpus. While I only use stock-related Tweets, they extract sentiment from the entire corpus of English Tweets and US Facebook posts respectively. This may explain the different results. While public mood, measured in many different ways, was shown to be significantly correlated with future stock returns, my study focuses on stock market related social media content. Hence, the extracted sentiment reflects beliefs towards the value of specific stocks within the finance affine clientele rather than expressing the general mood state of the entire English speaking social media community.

I conclude that noise traders beliefs about the true value of stocks may be influenced by general public mood states as suggested by Bollen et al. and Karabulut. However, there is little evidence for a relationship between noise traders beliefs and stock related

content on Twitter aggregated on a daily basis. Instead, meaningful interactions between stock returns and sentiment of equity Tweets are found in an hour range (section 4.1). Therefore, it is reasonable to assume that sentiment extracted from equity Tweets follows short-term fluctuations in investor sentiment rather than forecasting changes in investor sentiment on a daily to weekly basis.

5 Machine Learning Methods

Despite the fact that sentiment classification methods based on feature lexica (Section 4) have a long tradition in NLP, machine learning approaches gained popularity in the domain of sentiment analysis throughout the last decade. Building and maintaining comprehensive sentiment lexica and semantic rules (e.g. for negation) demands strong expertise and extensive effort from linguists.

In contrast, modern machine learning methods do not require sophisticated hand crafted features as input. In many cases good results can be achieved by using raw text or low level features that do not require much effort to build. For instance, commonly used classifiers such as Naive Bayes or Support Vector Machines were shown to work well with simple word or bi-gram occurrence counts (Pang et al., 2002).

A more recent development in sentiment analysis is feature learning. "It allows to learn expressive features for the documents directly from the raw data" (Albertini et al., 2014). In fact, machine learning algorithms such as in Socher et al. (2011) and Maas et al. (2011) are used to generate continuous vector-space representation for words and phrases. Based on learned word vector features, powerful deep learning methods such as recursive and recurrent neural networks have been used in latest research to perform sentiment analysis.

After performing sentiment analysis on the collected equity tweets with a lexicon based approach, I aim to asses the potential of machine learning for making smart trading decisions based on investor sentiment expressed in Tweets. The equity Tweets that were collected differ in many ways from an usual sentiment analysis setting. First, words,

phrases and tokens (i.e. emoticons) that clearly express sentiment orientation (e.g. excited, fantastic, bad, success ...) only occur sparsely within Tweets related to stocks. Second, the shortness of the Tweets combined with the use of abbreviated language and special expressions such as hashtags makes it, even for humans, hard to interpret the intended meaning and asses the semantic orientation of such Tweets. In some cases, knowledge of finance specific jargon is necessary to do so.

These characteristics make it very difficult to reliably extract investor sentiment with a lexicon based approaches such as VADER. The underlying general-purpose lexicon neither comprises features specific to the task, nor can relevant nuances.

Theoretically, machine learning models have the potential to learn domain specific features and create sophisticated internal representation that allow them to capture such nuances, relevant for successfully assessing investor sentiment. In section 4, a significant relationship between VADER scores and short-term stock market returns was found. The question I pose and try to answer in the following is whether modern machine learning methods are able to forecast stock price changes based on the collected equity Tweets even better than the lexicon-based approach VADER. First, I establish a benchmark by creating several base line models that use standard machine learning classifiers. In a further step, I implement a state of the art deep learning model for sentiment classification and analyse if it capable of making smart trading decisions given the Tweets as input. Due to the computational burden that comes with machine learning, the following analysis will be restricted to 30 minute and 1 hour stock price returns.

5.1 Classification based on word occurrence counts

In this section a simple machine learning baseline model for sentiment classification is introduced. Occurrence counts of distinct words and tokens in the Tweets are used as features. To produce these features, the Tweets are split up into tokens and words, followed by creating a word-count matrix. Given n Tweets and V as set of all words/tokens occurring throughout the Tweets, the token-count matrix is defined as $C \in \mathbb{N}^{n \times |V|}$, where $c_{i,j}$ denotes the number of times token/word j occurs in tweet i. Due to the short length

of a tweet and the large set of possible tokens/words, most counts $c_{i,j}$ are 0 resulting in a sparse feature space.

Using the word-count matrix C as input, three different machine learning classifiers are applied, attempting to predict the the stock price movement label of a tweet as assigned in 3.3. First a simple probabilistic classifier that assumes independence of the features, known as Naive Bayes Classifier, is applied. As word-occurrence counts are sparse features, dependencies among the words only play a minor role. Thus, Naive Bayes Classifiers work surprisingly well in the domain of NLP. Another classifier I use due its good performance in sparsity are random forests, an ensemble learning method based on decision trees. As a third classification approach, Support Vector Machines (SVM) are chosen. SVMs attempt to divide classes in the feature space by a clear gap that is as wide as possible. SVMs are often used with kernels that implicitly project the input data to higher-dimensional feature space to allow for non-linear separation. However, I do not apply any kernel method since the word-count feature space is already high-dimensional as well as sparse and thus can be separated linearly.

To assess the performance of the three classifiers I use 5-fold cross validation. Therefore the data is split into five parts of equal size. Then the classifiers are trained on four of the five splits and validation metrics are calculated on the remaining split. I repeat this process five times and shuffle the splits each time so that all five splits were used for validation. In the end, the metrics calculated in all five validation runs are averaged.

Specifically, I calculate two common classification metrics - accuracy and F1 score. Accuracy simply states the percentage of training samples that were classified correctly. However, when class labels are not uniformly distributed among the classes, the use of accuracy can be misleading. In our case 50% of the Tweets were labelled neutral while the classes "positive" and "negative" equally share the remaining half. To illustrate the problem, two naive classifiers that assign all Tweets the label "neutral" and "positive" respectively shall be considered. Although both classifiers share the same level of naivety, classifying all Tweets "neutral" results in an accuracy of 50% whereas always assigning "positive" only classifies 25% of the data correctly. Due to the sensitivity of the accuracy

metric towards imbalanced classes, the F1 score is often consulted as validation metric in classification settings since it accounts for possible imbalances among the classes. Hence, I primarily use the F1 score for assessing and comparing performance of classifiers.

Moreover, the estimated profit per trade of a trading strategy based on the classifiers predictions is calculated. The trading strategy can be formulated the following way:

- If a positive stock price reaction is predicted go long and hold the position for 30 min, respectively 1 hour
- If the class "neutral" is predicted do not trade the stock corresponding to the Tweet
- If a negative stock price reaction is predicted go short and hold the position for 30 min, respectively 1 hour

The evaluation of the three classification methods Naive Bayes, Support Vector Machines and Random Forests is illustrated in Table 6. Considering the F1 score both both time lags, the Random Forest approach outperforms the other two classifiers by more than 3 percent points. Nonetheless, an F1 score of less than 0.4 indicates that all three sentiment classifiers do not perform much better than picking one of the classes by chance. The estimated profits are close to zero and all of the corresponding p-values are greater than 20%, concluding that the proposed trading strategy based on standard machine learning sentiment classification does not lead to significant excess returns.

Since the Tweets were labelled "negative", "positive" and "negative" in correspondence to the approximately normally distributed 30 min / 1 hour stock returns, the training data might be very noisy. Hence, it is very hard for baseline machine learning models to learn from the provided data and make meaningful predictions.

5.2 Deep Learning for Sentiment Classification

As in many other areas of research, the resurgence of artificial neural networks (i.e. deep learning), shaped the domain of natural language processing during the last years. Accounting for the hierarchical structure of human language, Socher et al. (2010) presented tree shaped artificial neural network model for NLP, referred to as recursive neural

Classifier	metric	30 min lag	1 hour lag
Naive Bayes Classifier	accuracy	0.4906	0.4893
	F1 score	0.3534	0.3529
	estimated profit	$-9.12 * 10^{-7}$	$-5.95 * 10^{-6}$
	p-value	0.4520	0.4550
SVM (linear kernel)	accuracy	0.4852	0.4795
	F1 score	0.3610	0.3618
	estimated profit	$7.041 * 10^{-6}$	$-2.91 * 10^{-6}$
	p-value	0.2595	0.5225
Random Forests	accuracy	0.4509	0.4505
	F1 score	0.3922	0.3947
	estimated profit	$4.42 * 10^{-6}$	$-3.74 * 10^{-6}$
	p-value	0.3154	0.5036

Table 6
5-fold cross validation metrics for various supervised 3-class sentiment classification methods based on word occurance features. a) classifiaction accuracy b) F1 Score c) estimated profit $\hat{p}$ per trade when going long at positive assigned label and short when negative sentiment is predicted. One trade: holding short/long position for 30 minutes and 1 hour respectively d) p-value of estim. profit under i.i.d. assumption; $H_0 : \hat{p} = 0$

network (RNN). Originally, the idea of a recursively built neural networks was introduced by Goller and Kuchler (1996) to learn distributed representations of structured objects such as logical terms. Meanwhile the concept of RNNs has been applied to various problems of NLP, such generating parse trees, Part-of-Speech-Tagging (POS), Named-Entity-Recognition (NER), as well as sentiment analysis.

Usually, neural network language models use semantic vector spaces as features. This means that every word is represented by a n-dimensional vector in a continuous vector space (i.e. $\mathbb{R}^n$). Word vectors, also called word embeddings, can be generated through random initialization or by retrieving them from a general semantic vector space model like GloVe (Pennington et al., 2014) or Word2Vec (Mikolov et al., 2013). Word embeddings that were pre-trained as in Word2Vec or generated from a co-occurrence counts (GloVe) already embody semantic and syntactical information. Therefore, using these "smart" word embedding instead of random initialization, significantly improves the performance of machine learning models on top.

In contrast to bag-of-words models like VADER that mainly rely on sentiment of individual words, RNNs can also capture the compositionality of sentences and phrases.

Hence, they are able to interpret more complex negations and modular statements in the right way. To make the model more sensitive to changes in sentiment that result from interacting words and phrases (e.g. fairly good, not bad, pretty awesome) Socher et al. (2013) introduced a recursive-neural tensor network, an extension of RNNs, which basicly adds a tensor product to each recursive unit in order to improve interaction among the words and phrases.

With increasing sentence length, the depth of the tree-shaped RNNs usually grows as well. This comes with difficulties during training, know as the problem of vanishing and exploding gradients (Bengio et al., 1994). Exploding gradients can be avoided through a simple but effective technique called gradient clipping (Pascanu et al., 2013). A way to deal with vanishing gradients proposed by Hochreiter (1998) is to use long short-term memory (LSTM) units instead of regular neural network layers. LSTM units contain gates that allow them to decide whether an input is important and should be remembered or a value is less relevant and can be forgotten. These gates to enable the network units to emphasize relevant inputs and persist them over a longer sequence of propagation steps.

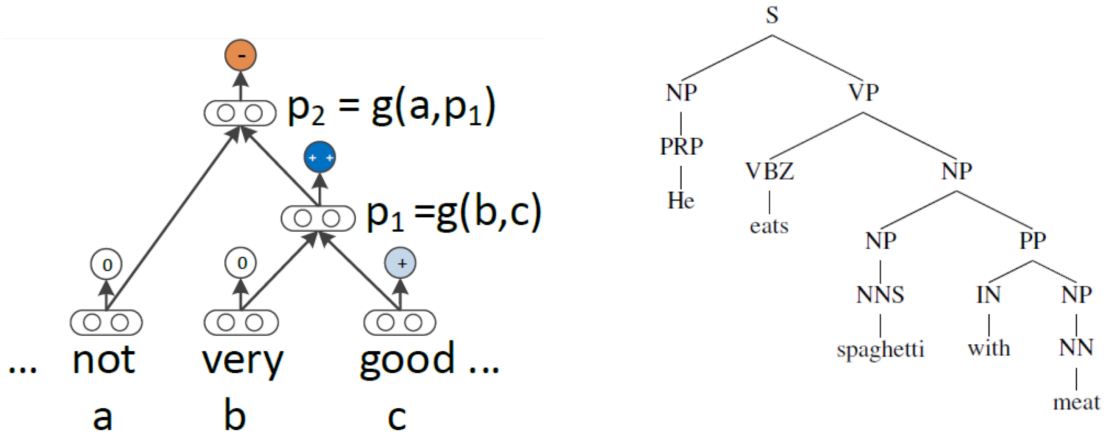
Tai et al., 2015 generalize the idea of LSTMs, originally designed for recurrent neural networks, to arbitrary tree structure. They also showed that the use of tree-structured LSTMs for sentiment analysis could further increase the accuracy for fine-grained movie review sentiment classification.

5.3 A Deep Learning Approach - Tree-Structured Long Short-Term Memory Networks

5.3.1 Model

Amongst other techniques such as convolutional neural networks as presented in Kim (2014), tree-structured LSTMs generate state-of-the-art results in terms of sentiment classification on well-established corpora. For this reason, I implement a tree LSTM model and apply it to the equity Tweets that were collected.

Although deep learning models can be very powerful, they might be difficult to train



(a) RNN for Sentiment Classification[5] **(b)** Constituency Parse Tree[6]

Figure 6
Scematical illustration of a) recursive neural network for sentiment classifiaction and b) constituency parse tree in chomsky normal form

and require huge amounts of data as well as a lot of computation power. Usually the success of deep learning heavily depends on the amount and quality of training data. Also fine tuning of hyperparameters and proper use of regularization are important to ensure good performance without overfitting the model. Since a technical and mathematical details of the applied model and training process are beyond the financial scope of this thesis, they are described in the appendix. In the following, I provide a rough description of the model as well as an outline of steps that were necessary to prepare the data inputs and train the model.

RNNs and tree-structure LSTMs rely on a tree-like representation of the text that shall be passed through the network. As Tweets can vary in its length and structure it is necessary to determine such a hierarchical representation for every distinct Tweet. I use the Stanford CoreNLP[4] tool with a tf-idf language model to generate a binary constituency parse tree (Figure 6b) for each Tweet. In a following step, all non-terminal unary nodes of the tree (e.g. PRP in Figure 6b) are removed so that the tree is strictly binary. Based on the binarized parse tree, a recursive neural network structure, as illustrated in Figure 6a, can be build individually for each Tweet.

As neural networks require numerical vectors as input, words and tokens represented

[4] http://stanfordnlp.github.io/CoreNLP/
[1] Socher et al. (2013)
[6] Source: http://cs224d.stanford.edu/lectures/CS224d-Lecture10.pdf

by terminal nodes in the parse tree need to be converted into word embeddings. I retrieve word vectors from the semantic vector space model GloVe (Pennington et al., 2014). Each word/token is represented by a 300-dimensional vector of real numbers that already embodies semantic and syntactic information about the respective word/token.

On the first stage of forward propagation, two word embeddings (b and c in Figure 6a) are fed into a LSTM unit which aggregates the two word vectors, resulting in a abstract internal representation (p_1), a 100-dimensional vector. Then, level by level up the tree, internal representations and word embeddings are fed through LSTM units creating semantic representations of increasing abstraction. This process culminates in a single vector at the root of the tree which is an aggregation of sentiment relevant information gathered from the entire Tweet. Finally, a softmax classifier that uses the root vector representation as input calculates probabilities for the three sentiment classes 'negative', 'neutral, 'positive. The sentiment label with the highest probability is then assigned to the Tweet.

Throughout the tree-structured LSTM network there are 220,703 parameters (i.e. weights and biases) that need to be trained. Furthermore, word embeddings are included in the training updates. As suggested in Tai et al. (2015) I use a cross entropy loss with L2 regularization as cost function that shall be minimized during training. Optimization steps are performed through stochastic gradient decent (SGD) in combination with AdaDelta, an adaptive learning rate method introduced by Zeiler (2012).

5.3.2 Training Approaches

To correctly assess the performance of artificial neural network models, generating training, development and test splits of the underlying data was shown to be vital. While the training split is solely used for training the model with SGD, the development split is is held back during training to measure the performance in order to do hyper parameter tuning, avoid overfitting and/or compare different model variants. Although the development split is never directly used for training, it indirectly affects the final model configuration throughout the process of model selection and parameter tuning. There-

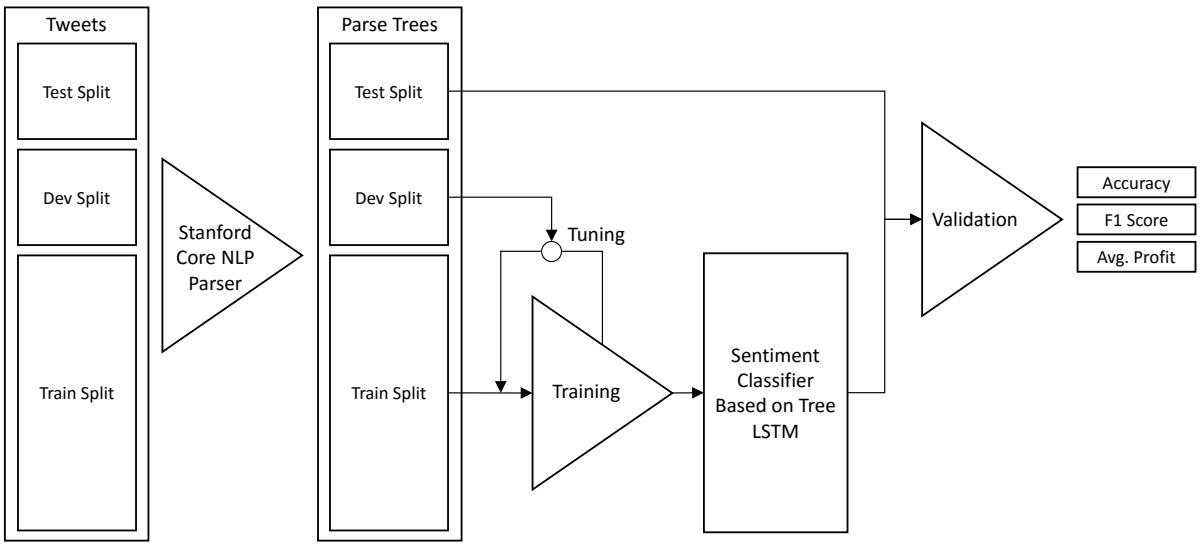


Figure 7
Process of Data Preparation, Training and Valitation

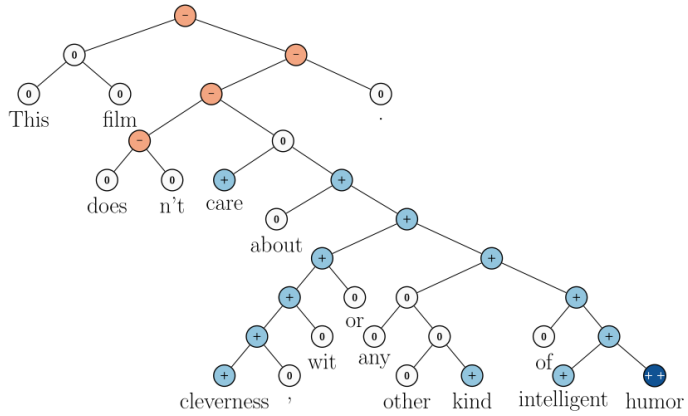


Figure 8
Stanford Sentiment Treebank - Example of a movie review parse tree with sentiment labels (Socher et al., 2013)

fore, a third split, the test split, is held back and never even touched until the final model configuration has been determined. For the final assessment of the chosen model, performance metrics are calculated based on the test set. Figure 7 depicts the whole process of data preparation, training, tuning and validation.

The best way to measure the final performance of a model is to test it on data that was independently retrieved from the training and development split. Thus, I randomly sample training and development split (85%/15%) from the Tweets that were posted between October 2015 and June 2016, whereas the test data is comprised of Tweets that were published at a later point of time i.e. from July to September 2016.

Figure 9
Classification accuracy on the development split throughout the process of training on the Tweets labelled according to a) 30 min and b) 1 hour stock price responses

To train the model, several approaches are used. First, I train the tree-based LSTM with well establish sentiment corpora, namely the Stanford Sentiment Treebank and the Sanders Analytics Twitter Sentiment Corpus[7]. The Stanford Sentiment Treebank (SST) was established by Socher et al. (2013) addressing the need for a sentiment corpus that is customized for tree-based models. It comprises of 11,855 sentences from movie reviews. In contrast to a regular sentiment datasets, the sentences are provided as parse trees in which every node was assigned a sentiment label (Figure 10). This structure allows a sentiment model to optimally capture the basic concepts of sentiment polarity and negation during training. Although movie reviews are notably different from equity Tweets, the SST constitutes an excellent basis for training a sentiment classifier.

Sander's Twitter Sentiment Corpus consists of 5513 hand-classified Tweets that refer to large Dow Jones companies such as Apple and Microsoft. Due to its nature, the corpus provides excellent training data for the context of equity micro blogs and investor sentiment. When trained and validated on the two corpora, the model improves fast and monotonous, converging after five training epochs.

[7] Source: http://www.sananalytics.com/lab/twitter-sentiment/

Training Data	metric	30 min lag	1 hour lag
SST & Sanders Corpus	accuracy	0.4915	0.4915
	F1 score	0.3527	0.3527
	estimated profit	$5.95 * 10^{-7}$	$5.96 * 10^{-7}$
	p-val profit	0.2918	0.5095
Collected Tweets (30 min lag label)	accuracy	0.3744	
	F1 score	0.3777	
	estimated profit	$3.2 * 10^{-7}$	
	p-val profit	0.2995	
Collected Tweets (1h lag label)	accuracy		0.3387
	F1 score		0.3430
	estimated profit		$-1.37 * 10^{-5}$
	p-val profit		0.1907

Table 7
Validation results based on test split (49,000 Tweets) : a) classification accuracy b) F1 score c) estimated profit $\hat{p}$ per trade when deploying a trading strategy based on the sentiment predictions d) p-value of estim. profit under i.i.d. assumption; $H_0 : \hat{p} = 0$

In a second approach the model is trained with the collected equity Tweets directly. The Tweets are labelled as described in section 3.3. Since 30 min and 1 hour stock responses were examined in the context of VADER, I continue with this approach and separately train the tree LSTM network with labels corresponding to the 30 min lag and 1 hour lag. In contrast to he training on SST and Sander's Corpus, neither training nor development error seem to decrease during the process of training. As illustrated in Figure 9, the classification accuracy measured on the development split randomly changes during training, not indicating any improvement.

5.3.3 Validation and Model Assessment

The model that was trained on the Stanford Sentiment Treebank and Sanders Twitter Sentiment Corpus is first validated on a test split that is comprised of data from both datasets. The trained model predicts the correct sentiment label with 64.79% accuracy (F1 score = 0.6471), which is a remarkable result given the fact that only root labels of the SST were used for training. The chance of mislabelling of a positive sentence/Tweet as negative and vice versa within the test set is lower than 15 %.

The important question is whether the model can generalize well and predict short-term stock price changes based on Tweet sentiment. Thus, I validate the model per-

lag	DJIA	AAPL	MSFT	GS
1 day	0.9720	0.6287	0.2740	0.3364
2 days	0.6529	0.5804	0.2079	0.6243
3 days	0.8053	0.3935	0.3031	0.7244
4 days	0.9111	0.5146	0.3910	0.9638
5 days	0.8377	0.6658	0.3820	0.8144
6 days	0.9176	0.7187	0.0918	0.8741

Table 8

Statistical significance (p-values) of Granger-Causality correlation between daily sentiment score S_t and daily stock market changes D_t; S_t is defined as weighted sum of sentiment labels troughout a day, predicted by the tree-structured LSTM model trained on SST and Sanders Corpus; Null Hypothesis: encorporating lagged sentiment values in the regression model does not increase the variance explained (see section 4.2)

formance by predicting sentiment labels for the Tweets in the test split and calculating measures of performance. Because the texts included in Stanford Sentiment Treebank data are more opinionated than equity Tweets, the majority (>90%) of Tweets in the test set is predicted to be neutral. Both accuracy and F1 score (see Table 7) indicate that predictions do not notably outperform random classification by chance. Furthermore, a trading strategy as in section 4.1 that bets on a reversion of the current stock price movement in response to Tweets with negative/positive sentiment polarity is analyzed. The estimated profit per 30 min trade of such a trading strategy exceeds average market return of -1.68 basepoints per 30 min lag. However, even under the strong assumption of independent and identically distributed 30 min stock price responses, it cannot be considered a significant excess return (p-val = 0.2918). Equivalent results were found for a training strategy wherein positions are hold for 1 hour.

Consistent with the problems that were observed during training, validation results as shown in Table 7 indicate that the model, trained with the labelled equity Tweets, is not able to make predictions that are significantly better than randomly selecting one of the three classes.

In a further step, I asses the model's ability to predict stock market changes on a daily basis. First, predicted sentiment labels corresponding to Tweets throughout a day are aggregated to a daily sentiment score. This score is defined as weighted sum of the labels, where -1 denotes negative, 1 positive and 0 neutral sentiment. Similar

to section 4.2, a Granger-Causality analysis is used to investigate the relationship of daily sentiment scores and stock markets. Table 8 holds the p-values corresponding to the Granger-Causality analysis performed. Neither for the DJIA nor for specific stocks are found to be significantly Granger-correlated with the respective time series of daily sentiment scores.

All in all, results indicate that machine learning methods are unsuited for the task of predicting stock market returns by extracting relevant sentiment from equity Tweets. When trained standard sentiment corpora the models do not generalize well enough in order to make useful predictions in the domain of financial social media content. If directly trained with the Tweets, labelled with the corresponding short-term stock market return, the models fail to learn meaningful patterns due to the fuzziness of the data. Since deep learning model are rely more on good training data quality than simpler machine learning classifiers, they perform worse on the task.

6 Conclusion

This study systematically analyses the relationship between sentiment in stock related Tweets and stock returns. Addressing the core question, how the content of stock related Tweets is linked stock price movements, I assess multiple methods for sentiment extraction and consider different timespans for interaction.

Beside traditional lexicon-based sentiment analysis approaches, I analyze the capability of machine learning models to capture sentiment from Tweets, relevant for predicting stock market responses. Due to the fuzziness of stock market data, basic machine learning classifiers as well as deep learning models are found to be not capable of learning to make valuable stock market predictions based on equity Tweets.

Studies that investigate the interaction of investor sentiment in social media and daily stock markets returns (Bollen et al., 2011; Karabulut, 2013) conclude that noise traders beliefs about the true value of stocks may be influenced by general public mood states. In contrast, I find no statistical evidence for a relationship between sentiment,

extracted from equity Tweets, and daily stock market returns. A likely explanation for the different findings might be that Bollen et al. and Karabulut use general public mood in social media as poxy for investor sentiment, whereas this study specifically concentrates on stock market related content on Twitter.

When examining shot-term interactions, I find a statistically significant negative correlation of sentiment in equity Tweets, extracted with VADER, and stock returns in a 20 to 40 minute range after a respective Tweet is posted. Moreover, stock returns preceding a Tweet are shown to be positively correlated with its sentiment. Based on these results, I conclude that sentiment of equity Tweets follows past investor sentiment and thus reflects the cause of the current stock price deviation from fundamentals. The negative correlation of sentiment and 30 minute stock returns indicates predictability of a partial stock price reversion to fundamentals in the near future. Stock returns, succeeding the publication of a Tweet by more than 40 min, were found to be increasingly uncorrelated with the measured sentiment. This is compatible with theory of De Long et al. (1990) which states that increased risk impedes efficient reversal of prices to the fundamentals, combined with a cumulating distortion of the expected stock price reaction due to new information and noise trader's changing beliefs over time.

A simple zero-cost trading strategy that uses the VADER sentiment scores to make trading decisions, betting on a reversion of the current stock price movement in response to Tweets with strong sentiment polarity, is shown to have significantly higher returns than a naive randomized dummy strategy. As costs for commission, bid-ask spreads, limited market liquidity and taxes are neglected, it is questionable whether the sentiment-based strategy would be still profitable in practice. Finally, such limitations to high-frequency arbitrage may impede markets from responding efficiently to the sentiment embedded in financial Twitter content (cf. Tetlock, 2007).

This analysis only considers Tweets that refer to stock tickers (e.g. $AAPL) thus limiting the authors and readers of the Tweets to a finance affine clientele. Incorporating Tweets that refer to the company (e.g. #Apple) on a more general level in a future analysis could reveal interesting insights on how the overall public opinion on specific

companies affects their stock price. Moreover, the analysis is limited to a 10 moth time period characterized by economic growth. To gain better understanding of the interaction between sentiment of equity Tweets and asset prices, analyzing the relationship throughout a whole economic cycle, especially during recession, might be a promising approach for future research.

Declaration

I declare that I have developed and written the enclosed bachelor thesis

Investor Sentiment on Twitter and Its
Link to the Stock Market

completely by myself, and have not used sources or means without declaration in the text.

Karlsruhe, November 8th 2016

.....

(Forename Surname)

References

Albertini, Simone, Alessandro Zamberletti, and Ignazio Gallo (2014). "Unsupervised feature learning for sentiment classification of short documents". In: *Journal for Language Technology and Computational Linguistics* 29.1, pp. 1–15.

Antweiler, Werner and Murry Frank (2004). "Is All That Talk Just Noise? The Information Content of Internet Stock Message Boards". In: *The Journal of Finance* 59.3, pp. 1259–1294.

Asghar, Muhammad Zubair et al. (2014). "A Review of Feature Extraction in Sentiment Analysis". In: *J. Basic. Appl. Sci. Res* 4.3, pp. 181–186.

Baccianella, Stefano, Andrea Esuli, and Fabrizio Sebastiani (2010). "SentiWordNet 3 . 0 : An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining". In: *Proceedings of the Seventh Conference on International Language Resources and Evaluation*, pp. 1–12.

Baker, Malcolm and Jeffrey Wurgler (2006). "Investor Sentiment and the Cross-Section of Stock Returns". In: *The Journal of Finance* 61.4, pp. 1645–1680.

Bengio, Yoshua, Patrice Simard, and Paolo Frasconi (1994). "Learning long-term dependencies with gradient descent is difficult". In: *IEEE Transactions on Neural Networks* 5.2, pp. 157 –166.

Bollen, Johan, Huina Mao, and Xiaojun Zeng (2011). "Twitter mood predicts the stock market". In: *Journal of Computational Science* 2.1, pp. 1–8.

Butler, Kirt C. and S. J. Malaikah (1992). "Efficiency and inefficiency in thinly traded stock markets: Kuwait and Saudi Arabia". In: *Journal of Banking and Finance* 16.1, pp. 197–210.

Cootner, Paul H. (1964). *The random character of stock market prices*. Cambridge, Mass: M.I.T. Press.

De Long, J. Bradford, Andrei Shleifer, and Lawrence H. Summers (1990). "Noise Trader Risk in Financial Markets". In: *The Journal of Political Economy* 98.4, pp. 703–738.

Fama, Euge et al. (1969). "The Adjustment of Stock Prices to New Information". In: *International Economic Review* 10.1, p. 1.

Fama, Eugene F. (1965a). "Random Walks in Stock Market Prices". In: *Financial Analysts Journal* 21.5, pp. 55–59.

— (1965b). "The Behavior of Stock-Market Prices". In: *The Journal of Business* 38.1, p. 34.

Gallagher, Liam A. and Mark P. Taylor (2002). "Permanent and Temporary Components of Stock Prices: Evidence from Assessing Macroeconomic Shocks". In: *Southern Economic Journal* 69.2, p. 345.

Garcia, Diego (2013). "Sentiment during Recessions". In: *The Journal of Finance* 68.3, pp. 1267–1300.

Gilbert, Eric and Karrie Karahalios (2010). "Widespread Worry and the Stock Market". In: *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*, pp. 58–65.

Goller, C. and A. Kuchler (1996). "Learning task-dependent distributed representations by backpropagation through structure". In: *Proceedings of International Conference on Neural Networks*. Vol. 1. IEEE, pp. 347–352.

Hirshleifer, David and Tyler Shumway (2003). "Good Day Sunshine: Stock Returns and the Weather". In: *The Journal of Finance* 58.3, pp. 1009–1032.

Hochreiter, Sepp (1998). "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 06.02, pp. 107–116.

Hutto, CJ J. and Eric Gilbert (2014). "Vader: A parsimonious rule-based model for sentiment analysis of social media text". In: *Eighth International AAAI Conference on Weblogs and ...* Pp. 216–225.

Karabulut, Yigitcan (2013). "Can Facebook Predict Stock Market Activity?" Frankfurt.

Keynes, JM (1936). *General theory of employment, interest and money*. Palgrave Macmillan, p. 472.

Kim, Yoon (2014). "Convolutional Neural Networks for Sentence Classification". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1746–1751.

Koppel, Moshe and Itai Shtrimberg (2006). "Good News or Bad News? Let the Market Decide". In: *Computing Attitude and Affect in Text: Theory and Applications*, pp. 297–301.

Lee, Charles M. C., Andrei Shleifer, and Richard H. Thaler (1991). "Investor Sentiment and the Closed-End Fund Puzzle". In: *The Journal of Finance* 46.1, p. 75.

Maas, Andrew L et al. (2011). "Learning Word Vectors for Sentiment Analysis". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150.

Mikolov, Tomas et al. (2013). "Efficient Estimation of Word Representations in Vector Space". In: *Proceedings of the International Conference on Learning Representations*, pp. 1–12.

Nofsinger, John R. (2005). "Social Mood and Financial Economics". In: *Journal of Behavioral Finance* 6.3, pp. 144–160.

Pang, B. and L. Lee (2004). "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts". In: *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, p. 271.

Pang, Bo and Lillian Lee (2005). "Seeing stars". In: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics - ACL '05*. 1, pp. 115–124.

— (2008). "Opinion mining and sentiment analysis". In: *Foundations and Trends in Information Retrieval* 2.12, pp. 1–135.

Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan (2002). "Thumbs up?: sentiment classification using machine learning techniques". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 79–86.

Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio (2013). "On the difficulty of training Recurrent Neural Networks". In: *Proceedings of the 30th International Conference on Machine Learning*. 2. Atlanta.

Pennebaker, James W., Martha E. Francis, and Roger Roger (2001). *Linguistic Inquiry and Word Count*. Tech. rep.

Pennebaker, James W et al. (2007). "The Development and Psychometric Properties of LIWC2007".

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543.

Qian, Bo and Khaled Rasheed (2007). "Stock market prediction with multiple classifiers". In: *Applied Intelligence* 26.1, pp. 25–33.

Rick, Scott and George Loewenstein (2008). "The role of emotion in economic behavior". In: *Handbook of Emotion*, pp. 138–156.

Salton, G., A. Wong, and C. S. Yang (1975). "A vector space model for automatic indexing". In: *Communications of the ACM* 18.11, pp. 613–620.

Snyder, B. and R. Barzilay (2007). "Multiple aspect ranking using the good grief algorithm". In: *Proceedings of NAACL HLT*, pp. 300–307.

Socher, Richard, Christopher D Manning, and Andrew Y Ng (2010). "Learning Continuous Phrase Representations and Syntactic Parsing with Recursive Neural Networks". In: *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Fea- ture Learning Workshop*.

Socher, Richard, Jeffrey Pennington, and Eh Huang (2011). "Semi-supervised recursive autoencoders for predicting sentiment distributions". In: *Conference on Empirical Methods in Natural Language Processing, EMNLP* i, pp. 151–161.

Socher, Richard, Alex Perelygin, and Jy Wu (2013). "Recursive deep models for semantic compositionality over a sentiment treebank". In: *Proceedings of the ...* Pp. 1631–1642.

Stone, Philip J., Dexter C. Dunphy, and Marshall S. Smith (1966). *The General Inquirer: A Computer Approach to Content Analysis*. Oxford, England: M.I.T. Press.

Taboada, Maite et al. (2011). "Lexicon-Based Methods for Sentiment Analysis". In: *Computational Linguistics* 37.2, pp. 267–307.

Tai, Kai Sheng, Richard Socher, and Christopher D. Manning (2015). "Improved semantic representations from tree-structured long short-term memory networks". In: *Proceedings of ACL*, pp. 1556–1566.

Tetlock, Paul C. (2007). "Giving Content to Investor Sentiment: The Role of Media in the Stock Market". In: *The Journal of Finance* 62.3, pp. 1139–1168.

Turney, Peter D (2002). "Thumbs up or thumbs down? Semantic Orientation applied to Unsupervised Classification of Reviews". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)* July, pp. 417–424.

Turney, Peter D and Michael L Littman (2003). "Measuring Praise and Criticism: Inference of Semantic Orientation from Association". In: *ACM Transactions on Information Systems* 21.4, pp. 315–346.

Zeiler, Matthew D. (2012). "ADADELTA: An Adaptive Learning Rate Method".

Appendix

I Further Regression Results

stock	parameter	coef	std error	t	P> |t|
AAPL	Intercept	-0.7799	0.807	-0.967	0.335
	VADER score	-0.8132	0.582	-1.398	0.164
MFST	Intercept	0.9374	0.767	1.223	0.223
	VADER score	-0.5673	1.413	-0.402	0.689
GS	Intercept	0.7216	1.028	0.702	0.484
	VADER score	-3.6726	1.821	-2.017	**0.045**
DIS	Intercept	0.2806	1.962	0.143	0.886
	VADER score	-2.6954	3.900	-0.691	0.490
IBM	Intercept	1.8096	2.228	0.812	0.418
	VADER score	-4.7604	2.325	-2.047	**0.042**

Table 9
OLS regression summary - Predicting short-term stock price responses with VADER sentiment scores: This table shows the OLS estimates of the coefficients a and β in equation 4 fitted on 234,818 Tweets and stock data, corresponding to a) Apple Inc b) Microsoft Corporation c) Goldmn Sachs Group Inc. and d) IBM Corporation, from Oct 2015 to Aug 2016 with daily error clusters. Regressand: stock return r_i (in basepoints) within 30 minutes after a Tweet i is posted. Regressor: VADER compound sentiment score s_i of Tweet i

II Tree-Structured Long Short-Term Memory Networks

The basic idea of tree-structured LSTM networks is described in section 5.3. Based on constituency parse trees, a tree-shaped network of LSTM units is build. Word embeddings that represent piece of text, i.e. a Tweet, are propagated trough the tree-structured network in a bottom-up fashion.

To account for the tree structure, slight modifications in the architecture of LSTM units, originally designed for recurrent neural networks by Hochreiter (1998), are required. With regard to that, Tai et al. (2015) propose two extensions of the original architecture:

Child-Sum Tree-LSTMs and N-ary Tree-LSTMs. Both variants allow a LSTM unit to incorporate inputs from an arbitrary number of child nodes.

A Child-Sum Tree-LSTMs

Child-Sum Tree-LSTMs sum up the hidden states h_k of child units and then treats them as one input. Thus, these units are not able to draw any information from the arrangement of the child nodes which makes it more difficult to incorporate the word order. Given an underlying parsing tree, let $C(j)$ denote the set of children of node j. The translation functions of a Child-Sum Tree-LSTM unit are the following:

$$\tilde{h}_j = \sum_{k \in C(j)} h_k$$

$$i_j = \text{sigm}\left(W^{(i)} x_j + U^{(i)} \tilde{h}_j + b^{(i)}\right)$$

$$f_{jk} = \text{sigm}\left(W^{(f)} x_j + U^{(f)} h_k + b^{(f)}\right)$$

$$o_j = \text{sigm}\left(W^{(o)} x_j + U^{(o)} \tilde{h}_j + b^{(o)}\right)$$

$$u_j = \tanh\left(W^{(u)} x_j + U^{(u)} \tilde{h}_j + b^{(u)}\right)$$

$$c_j = i_j \odot u_j + \sum_{k \in C(j)} f_{jk} \odot c_k$$

$$h_j = o_j \odot \tanh(c_j)$$

where $k \in C(j)$

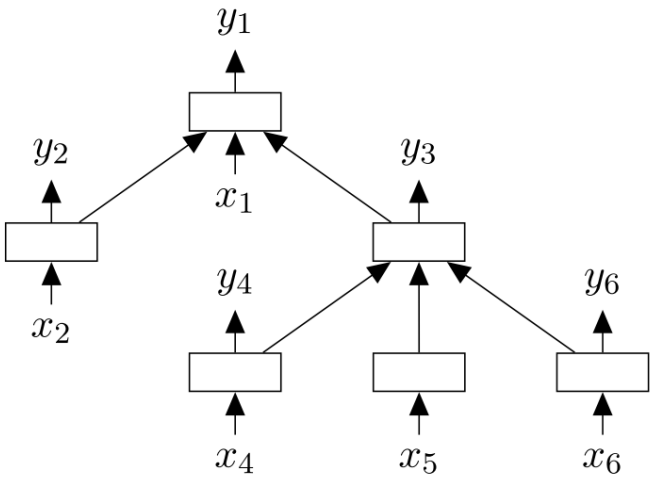


Figure 10
Tree-structured LSTM network with arbitrary branching factor; Source: Tai et al. (2015)

B N-ary Tree-LSTMs

In contrast to Child-Sum Tree-LSTMs, N-ary LSTMs can process information from child nodes differently, depending on their order. However, this requires to specify the maximal number of child nodes N beforehand. For any of the N possible child node positions there exist separate weight matrices. Since the parse trees that were created for the Tweets are binary, the branching factor N is 2 which means there are distinct weights for words and phrases that are fed from the left and from the right, respectively. This adds further intelligence to the model without letting the number of parameters explode. Hence, I use the N-ary Tree-LSTM network for the purpose of sentiment classification.

The translation equations are defined as follows:

$$i_j = \text{sigm}\left(W^{(i)}x_j + \sum_{l=1}^{N} U_l^{(i)} h_{jl} + b^{(i)}\right)$$

$$f_{jk} = \text{sigm}\left(W^{(f)}x_j + \sum_{l=1}^{N} U_{kl}^{(f)} h_{jl} + b^{(f)}\right)$$

$$o_j = \text{sigm}\left(W^{(o)}x_j + \sum_{l=1}^{N} U_l^{(o)} h_{jl} + b^{(o)}\right)$$

$$u_j = \tanh\left(W^{(u)}x_j + \sum_{l=1}^{N} U_l^{(u)} h_{jl} + b^{(u)}\right)$$

$$c_j = i_j \odot u_j + \sum_{l=1}^{N} f_{jl} \odot c_{jl}$$

$$h_j = o_j \odot \tanh(c_j)$$

III Classifier and Cost Function

Suppose j the root node of the LSTM tree that received a Tweet as input. I use a softmax classifier to predict the sentiment label $\hat{y}_j$ given the the information $\{x_j\}$ contained in the Tweet. The softmax classifier takes the hidden state h_j of the root LSTM unit as input:

$$\hat{p}_\theta\left(y|\{x_j\}\right) = \text{softmax}\left(W^{(s)}h_j + b^{(s)}\right)$$

$$\hat{y}_j = \arg\max_y \hat{p}_\theta\left(y|\{x_j\}\right)$$

The cost function is defined as sum cross entropy losses throughout the training data. Let θ denote the parameters of the model, then the cross entropy loss corresponding to a Tweet $\{x\}^{(k)}$ is defined as negative log-likelihood of the true label $y^{(k)}$. Furthermore a L2 regularization term with hyperparameter λ is added:

$$J(\theta) = -\frac{1}{n}\sum_{k=1}^{n} \log\, \hat{p}_\theta\left(y^{(k)}|\{x\}^{(k)}\right) + \frac{\lambda}{2}||\theta||_2^2$$

where n is the number of Tweets in the training set and (k) indicates the kth Tweet in the data.

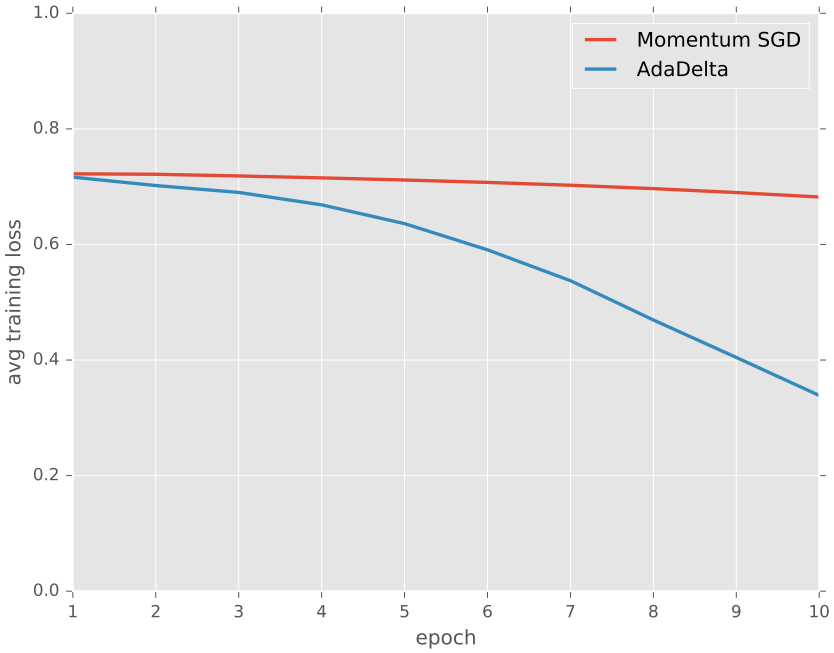


Figure 11
Training performance comparison of optimizers: Average training cross-entropy loss troughout the first 10 training epochs when training the tree-structure LSTM model on the SST utilizing a) SGD with momentum method and gradient clipping b) AdaDelta with gradient clipping

IV Optimizer

To minimize the cross entropy loss function and thereby enhance the model's capability to correctly classify Tweets based on their sentiment, I use Stochastic Gradient Descent (SGD) in combination with AdaDelta, an adaptive learning rate method.

Basic concept of of SGD is to perform gradient decent steps for each data item in the training set individually rather than calculating derivatives of the entire cost function $J(\theta)$. For the kth item in the training data an update step is defined as

$$\theta_t = \theta_t - 1 - \eta \, \nabla J(\theta; y^{(k)}, \{x\}^{(k)})$$

with $J(\theta; y^{(k)}, \{x\}^{(k)}) = \log \hat{p}_\theta \left(y^{(k)} | \{x\}^{(k)}\right) + \frac{\lambda}{2} ||\theta||_2^2$ and η as learning rate.

A downside of classical SGD is the fixed learning rate. On the one hand it is hard to find the optimal learning rate, on the other hand adjusting the learning rate during training by e.g. annealing can significantly improve the performance of the optimizer.

With regard to the learning rate problem various adaptive learning rate methods such as Adagrad, Adam or RMSprop have been introduced. A technique that was shown to be particularly efficient is Adadelta (Zeiler, 2012). It is an extension of Adagrad and uses a recursively defined decaying average of squared gradient and instead the average over all previous squared gradients g_t^2. The running average $E[g^2]_t$ is defined as follows:

$$E[g^2]_t = \gamma E[g^2]_{t-1} + (1-\gamma) g_t^2.$$

Usually γ is set to 0.9 or 0.95. Based on $E[g^2]_t$ and the running average of squared previous updates $E[\Delta\theta^2]_{t-1}$ a new update is calculated:

$$\Delta\theta_t = -\frac{\sqrt{\mathrm{E}[\Delta\theta^2]_{t-1} + \epsilon}}{\sqrt{\mathrm{E}[g^2]_t + \epsilon}}\, g_t$$

The term ϵ is added for numerical stability i.e. to avoid division by zero. Usually it is set close to zero, e.g. $\epsilon = 10^{-8}$. Based on the parameter update $\Delta\theta_t$ that was just calculated, the running average of squared updates $E[\Delta\theta^2]_t$ is renewed.

$$E[\Delta\theta^2]_t = \gamma E[\Delta\theta^2]_{t-1} + (1-\gamma)\Delta\theta_t^2$$

Finally, the update is applied to the model parameters:

$$\theta_t = \theta_{t-1} + \Delta\theta_t$$

I compare the performance of AdaDelta to SGD with momentum method during training of the tree-structure LSTM network on the SST data. As illustrated in Figure 11, the use of AdaDelta drastically enhances the process of training.

V Implementation and Infrastructure

The tree-structured LSTM is mainly implemented with the Python library Theano. This library helps to efficiently evaluate mathematical expressions with high-dimensional tensors trough the use of a Graphical Processing Units (GPU). To make the computation

power the GPU accessible I use Nvidia's GPU toolkit CUDA.

As training can take several days up to weeks on a personal computer, I run the training process on an Amazon Web Services EC2 GPU instance (g2.2xlarge). Although computational intensive processes such as back propagation could be accelerated significantly on a high-end GPU, one training session still took about three days.