# A Comparison of Neural Network Methods for Accurate Sentiment Analysis of Stock Market Tweets

Narges Tabari(✉), Armin Seyeditabari, Tanya Peddi, Mirsad Hadzikadic, and Wlodek Zadrozny

University of North Carolina at Charlotte, Charlotte, NC, USA
{nseyedit,sseyedi1,tpeddi,mirsad,wzadrozn}@uncc.edu

**Abstract.** Sentiment analysis of Twitter messages is a challenging task because they contain limited contextual information. Despite the popularity and significance of this task for financial institutions, models being used still lack high accuracy. Also, most of these models are not built specifically on stock market data. Therefore, there is still a need for a highly accurate model of sentiment classification that is specifically tuned and trained for stock market data.

Facing the lack of a publicly available Twitter dataset that is labeled with positive or negative sentiments, in this paper, we first introduce a dataset of 11,000 stock market tweets. This dataset was labeled manually using Amazon Mechanical Turk. Then, we report a thorough comparison of various neural network models against different baselines. We find that when using a balanced dataset of positive and negative tweets, and a unique pre-processing technique, a shallow CNN achieves the best error rate, while a shallow LSTM, with a higher number of cells, achieves the highest accuracy of 92.7% compared to baseline of 79.9% using SVM. Building on this substantial improvement in the sentiment analysis of stock market tweets, we expect to see a similar improvement in any research that investigates the relationship between social media and various aspects of finance, such as stock market prices, perceived trust in companies, and the assessment of brand value. The dataset and the software are publicly available. In our final analysis, we used the LSTM model to assign sentiment to three years of stock market tweets. Then, we applied Granger Causality in different intervals to sentiments and stock market returns to analyze the impact of social media on stock market and visa versa.

**Keywords:** Sentiment analysis · Neural networks · Social media · Stock market

## 1 Introduction

With the rise of social networks and micro-blogging, the amount of textual data on the Internet has grown rapidly, and the need to analyze it has increased along


C. Alzate et al. (Eds.): MIDAS 2018/PAP 2018, LNAI 11054, pp. 51–65, 2019.
https://doi.org/10.1007/978-3-030-13463-1_4

with it. Sentiment analysis has emerged as a useful and influential approach for using this data to investigate people's emotions and understand human behavior in multiple domains. For example, Bollen and Pepe [2] used social-media sentiment analysis to predict the size of markets, while Antenucci et al. [1] used it to predict unemployment rates over time.

Historically, sentiment analysis has been used to analyze longer form documents (e.g., reports, news stories, and blogs), but in the last few years, microblogging applications have seen a spike in their usage. These platforms – Twitter, Instagram, and Facebook – have rapidly become popular with professionals, celebrities, companies, and politicians, along with students, employees, and consumers of many services. The popularity of these platforms, and especially Twitter (which is text-oriented and fine-grained) provides a unique opportunity for companies and researchers to obtain a concise understanding of a single topic (e.g., the stock market) from different viewpoints.

Although social media and blogging are popular and widely used channels for discussing different topics, it is challenging to analyze their content. For example, Twitter messages generally have many misspelled words, grammatical errors, non-existent words, or unconventional writing styles. Additionally, the specific vocabulary used for analysis will depend on the topic under consideration, since the meaning and sentiment of a word can change in different contexts. For example, a word in a professional context might have positive or neutral sentiment (e.g., tax), while the same word generally has a negative sentiment in casual conversations. This prompted Loughran and Mcdonald [13] to suggest that using non-business word lists for sentiment analysis in a business context is inappropriate when using a Bag-of-Words approach.

Although many studies have concentrated on Twitter sentiment analysis in the context of the stock market, most of them either did not use a context-specific dataset, or they had low accuracy for their sentiment predictions. For example, Kolchyna et al. [10] combined lexicon-based approaches and support vector machines to classify tweets, resulting in a final accuracy of 71%. The topic of task 5 of the SemEval competition [3] was to perform fine-grained sentiment analysis on stock market tweets. Jiang et. al [7] won the first place in this competition by applying an ensemble method consisting of Random Forest, Support Vector Machine, various regression algorithms, and a combination of multiple features, such as word embeddings and lexicons. In our SemEval paper [24], we achieved an accuracy slightly lower the winning model, but with a simpler approach that used a Random Forest classifier and a revised financial lexicon from [13] as our feature set. In a recent paper, Sohangir et al. [22] evaluated regression models, data mining, and deep learning methods for sentiment analysis of financial tweets derived from StockTwits[1], and found that their CNN performed well, with an accuracy of 90.8%, while their LSTM did not perform as well, achieving an accuracy of only 69.9%.

In our work, after a precise labeling of our tweet dataset using Amazon Mechanical Turk (AMT), we applied vigorous and thorough preprocessing tech-

[1] www.stocktwits.com.

niques on the dataset. Then we created our baseline models, by building on our previous work [24], and then used SVM, and TF-IDF as our feature vector. Finally we thoroughly compared different Convolutional Neural Network (CNN) and Recurrent Neural Network (LSTM) with each other. We found that when using a balanced dataset of positive and negative tweets, and a specific pre-processing technique, a shallow CNN achieves the best error rate, while a shallow LSTM model, with a higher number of cells, achieves the highest accuracy of 92.7%. This is a significant improvement from our baseline or previous work in sentiment analysis in context of stock market.

Although sentiment analysis has been thoroughly studied before, we believe our work is novel in two different ways. First, there is not a publicly available annotated tweet dataset in context of stock market. Therefore, we believe that this dataset can help improve research in this area. Second, to the best of our knowledge most research on sentiment analysis in context of stock market has been studied widely using either basic machine learning classifiers or lexicon based models. Our work, on the other hand, is a one of the few thorough comparisons of neural network models that has been used in this context. And furthermore, none of previous models produced accuracy for the sentiments as high as our model.

We believe that this paper will open ways for research in a few areas measuring the impact of social media on various aspects of finance, such as stock market prices, perceived trust in companies, the assessment of brand value, and more. For instance, having a model that can predict highly accurate sentiment scores in this context, can help with the understanding of the causality analysis between social media and stock market better, or improve the prediction of stock prices using social media [2,12,13,17]. In addition, it can also be used to improve the quality of social media trust networks for stock market [18].

The outline of the paper is as follows. The dataset, the specification of how it was labeled using the Amazon Mechanical Turk, and information about labels are explained in Sect. 2. Section 3 covers the preprocessing techniques, and baseline methods. In Sect. 4, we explain all our deep learning models in details, and Sect. 5 thoroughly explains our deep learning results. In Sect. 6, first we describe our Granger Causality model and then we apply that model on the sentiments derived from Sect. 5 and stock market returns. Further, we analyze this causal analysis. And finally, we conclude our work in Sect. 7.

## 2 Data

Tweets were pulled from Twitter using Twitter API between 1/1/2017 and 3/31/2017. In our filters we only pulled tweets that are tweeted from a "Verified" account. A verified account on Twitter suggests that the account is a public interest and that it is authentic. An account gets verified by Twitter if the used is a distinguished person in different key interest areas, such as politics, journalism, government, music, business, and others. A tweet is considered stock related, if it contains at least one of the stock symbols of the first 100 most frequent stock

symbols that were included in SemEval dataset form [24]. We were able to pull roughly 20,000 tweets in that interval using mentioned filters.

### 2.1 Labeling Using Amazon Mechanical Turk

The data was submitted to Amazon Mechanical Turk was asked to be labeled by 4 different workers. Snow et al. [21] suggested that 4 workers is sufficient to have enough people submitted their opinion on each tweet, and to ensure the results would be reliable. We assigned only AMT masters as our workers, meaning they have the highest performance in performing wide range of HITs (Human Intelligence Tasks). We also asked the workers to assign sentiments based on the question: "Is the tweet beneficial to the stock mentioned in tweet or not?". It was important that tweet is not labeled based on perspective of how beneficial it would be for the investor, but rather how beneficial it would be to the company itself. Each worker assigned numbers from −2 (very negative) to +2 (very positive) to each tweet. The inter-rater percentage agreement between sentiments assigned to each tweets by the four different workers had the lowest value of 81.9 and highest of 84.5. We considered labels 'very positive' and 'positive' as positive when calculating the inter-agreement percentage.

At the end, the average of the four sentiment was assigned to each tweet as the final sentiment. Out of 20013 tweet records submitted to AMT, we assigned neutral sentiment to a tweet if it had average score between [−0.5, +0.5]. We picked the sentiment positive/negative if at least half of workers labeled them positive/negative. Table 1 is a summary of the number of tweets in each category of sentiment.

One downside of this dataset was that the number of positive and negative tweets are not balanced. In order to overcome this issue, we tried many things. At the end balancing the train set by oversampling our negative tweets led to the best result. We also have tried under-sampling positive train set, but it performed worse in accuracy.

**Table 1.** Summary of tweets labeled by Amazon Mechanical Turk.

| Range | Label assigned to tweets | Count |
|---|---|---|
| [−2, −0.5] | Negative | 2082 |
| [−0.5, 0.5] | Neutral | 9008 |
| [0.5, 2] | Positive | 8386 |

## 3 Method and Models

### 3.1 Preprocessing

Twitter messages due to its nature of being informal text, requires a thorough preprocessing step in order to improve classifier's prediction. Twitter messages

generally contain a lot of misspelled words, grammatical errors, words that does not exist, or has been written in a non-conventional way. Therefore, in our preprocessing step, we attempted to address all these issues in order to retrieve the most information possible from each tweet.

**Text Substitution.** We applied two different text substitutions. In our first attempt, we substitute every word that contains both number and a letter with <alphanum> tag, and all the numbers with the tag <num>. For instance, '12:30' would be replace with <num>:<num>, 'ftse100' will be replaced by <alphanum>, and '500' with <num>.

This way, all hours and measures are treated the same way. This reduces the number of non-frequent words in our vocabulary. For example, every time expression is replaced by <num>:<num>, and every price by $<num>.

**Spelling Correction.** In order to address the issue of misspelled words and try to retrieve as many words possible so that it can be recognizable by Word2Vec.[2] For example, we removed '-' or '.' in every word and checked whether after this operation they would be recognizable by Word2Vec. Additional preprocessing operations included:

- Removing 'ś'
- Changing word in 'Word1-word2' format to 'word1 word2'
- Deleting consecutive duplicate letters.
- Deleting '-' or '.' between every letter of word.

### 3.2 Word Embeddings

Word embeddings have been the most effective and popular feature in Natural Language Processing. The two most popular word embedding are GloVe [16] and Google's Word2Vec [14]. We used 300-dimensional pre-trained Word2Vec vectors whenever we could find a word available, and otherwise we assigned random initializations. From roughly 10,000 tokens in our vocabulary, around 600 of them was randomly initialized. It was essential for us to use pre-trained embeddings since we used to create a vocabulary in order to see if a particular word exists or not.

As future work, it would be interesting to train a new embedding model for stock market and see if that would increase the accuracy of our model.

### 3.3 Baseline Model

We used Amazon Mechanical Turk to manually label our stock market tweets. In order to create a baseline for our analysis, we applied on the current dataset the

[2] We applied Google's Word2Vec pre-trained model with 300 dimension to get word embeddings from each word.

preprocessing techniques explained before, and the same machine learning classification method and feature sets we designed for [24]. We modified Loughran's lexicon of positive and negative words [13] to be suited for stock market context, and used it to calculate the number of positive and negative words in each tweet as features. For example, 'sell' has a negative sentiment in stock market context, that has been added to Loughran's lexicon. We ultimately added around 120 new words to his list. Also, we replaced a couple of words that come together in a tweet, but has different sentiment in stock market context with one word, to be able to assign their actual sentiment. For example, 'Go down' and 'Pull back' both contain negative sentiment in stock's perceptive. Around 90 word-couples were defined specifically for this purpose. Table 2 shows the baseline for different machine learning classifiers.

**Table 2.** Baseline accuracy for 11,000 tweet dataset.

| Classifier | Feature set | Accuracy |
|---|---|---|
| Random forest | [TF-IDF] | 78.6% |
| Random forest | [TF-IDF, pos/neg count] | 78.9% |
| Random forest | [TF-IDF, pos/neg count, Wrod-couple] | 79.4% |
| SVM | [TF-IDF] | 77.9% |
| SVM | [TF-IDF, pos/neg count] | **79.9%** |
| SVM | [TF-IDF, pos/neg count, Wrod-couple] | 79.5% |

## 4 Neural Network Models

### 4.1 Convolutional Neural Networks

Convolutional Neural networks (CNNs) have been shown to be useful in a variety of applications, specially in image processing. Although they have been designed originally for image processing and classification, they found their way into natural language processing. Thus models created using CNNs led to state of the art result in text classification [8,15], and specifically in classifying tweets [19,20].

Our CNN model[3] contains an input layer, in which after pre-processing, we reshape each tweet to a matrix. Then we have a convolutional layer with specific filters, and finally a max-pooling layer. Specification of each layer is described as follows:

**Input Layer:** CNNs originally were introduced for image classification, and by design have a fixed size input layer. Therefore, the problem with using CNNs for tweet classification is the difference in size (i.e. number of words) in tweets. To overcome this problem, we made all tweets the same size by adding padding to

[3] Our model, was built and modified based on a Convolutional network available at https://github.com/bernhard2202/twitter-sentiment-analysis.

shorter tweets and cutting off the longer ones to make all our tweets the same length. We set the length of tweets to 35; and among all the tweets in our data, we had only 63 tweets that had to be shortened. This way, we could represent each tweet in our dataset by a $35 \times 300$ dimensional matrix; 35 being the number of terms in each tweets, and 300 is the dimension of the representative vector in our pre-trained embeddings.

**Convolutional Layer:** Having our input matrix and the convolutional layer, consisting of multiple sliding window functions, the whole matrix embedding vector (word), and these convolutions slide through the matrix to generate an output with each move. For example, a filter of length 5 would go through all 35 embedding vectors (words), 5 rows at a time for 30 steps, generating 31 outputs. In our experiment we used convolutions covering three, four and five words at a time, and the output is passed to a ReLU activation function.

**Max-pooling and Soft-max:** Then we create a 384 dimensional vector with max-pooling on the outputs of our convolutions for each tweet (in example above each convolution creates 31 outputs for each tweet, we select the maximum and disregard all others, so we get one output for each of 384 convolutions). This output vector then will be passed to a soft-max layer to generate a normalized probability score for classification.

**Training and Regularization:** Stochastic optimization with cross-entropy-loss was used to train the CNN using Adam optimizer [9]. The data was divided 90% to 10% as train and development sets. After every 1000 training step the performance of the CNN on development data was evaluated and the training was stopped after eight epochs (i.e. 70k training steps) with learning rate of 1e-4. We used this learning rate, because it is low enough to make the neural network more reliable. Although, this makes the optimization process slow, it was not our concern because of our relatively small dataset. A dropout layer for convolutions was used to avoid overfitting during training. This layer disables each neuron with the probability of 0.5, resulting in a network which uses on average half the neurons in the network in each training step.

### 4.2 Experimenting with Recurrent Neural Networks

Recurrent neural networks have been shown to be a powerful tool in many NLP tasks such as sentiment analysis [25], machine translation [23], and speech recognition [5]. In RNNs the input is fed to the network sequentially as opposed to CNNs, where you feed the whole input into the network at once. This makes RNNs a preferred candidate for sequential data with various size inputs, such as text. They are constructed with inter-unit connections which creates a directed graph, and their internal state can be considered to be a memory which keeps track of previous states.

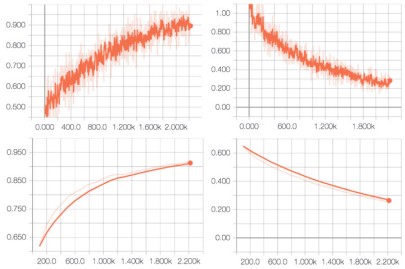


**Fig. 1.** Plots of accuracy and loss for each step in train and test set for best loss in CNN, from tensorboard. Top-left is the accuracy and top-right is the loss for train set. Bottom-left shows the accuracy and bottom-right shows the loss for each run in test set.

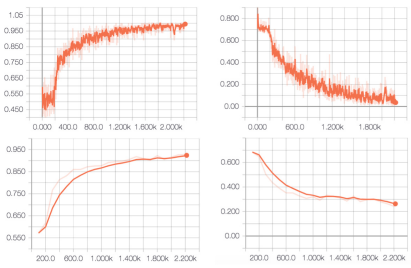


**Fig. 2.** Plots of accuracy and loss for each step in train and test set for best accuracy in LSTM, from tensorboard. Top-left is the accuracy and top-right is the loss for train set. Bottom-left shows the accuracy and bottom-right shows the loss for each run in test set.

An issue that arises from this design is that RNNs cannot handle long-term dependencies reliably during back propagation, resulting in vanishing or exploding gradients. This happens because the error propagates over a long distance in the network. Long Short-Term Memory (LSTM)network tries to overcome this issue by adding an explicit memory component to the network's architecture to prevent the gradients to decay very fast (and clipping large gradients prevents the exploding gradient problem). This is why we decided to try a LSTM network.

In this task, we used a network consisting an embedding layer, one layer of 128 LSTM units and a softmax layer to normalize the output. We also tried variations of this architecture: once with 256 LSTM cells, and once with two layer of 128 LSTM cells. You can see the performances for each of these architectures (along with other models) in Tables 3 and 4.

## 5 Results

As explained in the discussion of pre-processing, additional challenge of our dataset was the unbalanced nature of sentiments. In one experiment, we used an unbalanced test set as well as unbalanced train dataset. However, the result really jumped in accuracy when we used balanced train and test dataset. We re-sampled the negative tweets to create the same number of negative tweets as the positive ones. By doing that, our test set accuracy increased by 8% in CNN and 10% with LSTM.

Additional changes in preprocessing improved our accuracy drastically. We tried out two different preprocessing alterations. First attempt was examining the effect of removing or keeping '#' and '$' in the dataset. In all of our runs, we let these two characters remain in our dataset. The idea was that each hashtag would differentiate the word with or without these character and result in better capturing of the context. But ultimately, removing them increased the accuracy.

**Table 3.** Result of accuracy of different models

| NN | Specification | Train | Test |
|---|---|---|---|
| CNN | Unbalanced train/test | 91.5% | 80.6% |
| CNN | Balanced train/test | 89.7% | 88.7% |
| CNN | Remove '#' and '$' | 89.7% | 91.6% |
| CNN | Unique Tag | 95.9% | 90.4% |
| LSTM | Unbalanced Train/Test | 98.3% | 81.6% |
| LSTM | Balanced Train/Test | 97.9% | 91.6% |
| LSTM | Remove '#' and '$' | 91.8% | 91.8% |
| LSTM | Unique Tag | 98.4% | 91.1% |
| LSTM | 2 layer + 128 cell | 83.6% | 86.6% |
| LSTM | 1 layer + 256 cell | 98.4% | **92.7%** |

**Table 4.** Result of Loss of different models

| NN | Specification | Train | Test |
|---|---|---|---|
| CNN | Unbalanced train/test | 0.25 | 0.40 |
| CNN | Balanced train/test | 0.26 | 0.30 |
| CNN | Remove '#' and '$' | 0.31 | **0.253** |
| CNN | Unique tag | 0.20 | 0.27 |
| LSTM | Unbalanced train/test | 0.07 | 0.68 |
| LSTM | Balanced train/test | 012 | 0.31 |
| LSTM | Remove '#' and '$' | 0.28 | 0.27 |
| LSTM | Unique tag | 0.03 | 0.34 |
| LSTM | 2 layer + 128 cell | 0.39 | 0.31 |
| LSTM | 1 layer + 256 cell | 0.04% | 0.259 |

We believe this was due to the fact that our vocabulary was relatively small (10643 words), removing these characters helped with eliminating non-frequent words and reducing number of features. The effect of removing these characters can be seen in the lowest loss of 0.25 in our CNN model. Figure 1 shows the accuracy and loss for this model, for both train and test set in each step.

Second, we replaced all of our tags that have been explained in Sect. 3.1 with just one tag <num> with the same justification for removing characters. But, for both LSTM, and CNN we had slight decrease in accuracy and increase in loss.

LSTMs, in general, trained faster than CNNs, and the best accuracy was achieved when we used the higher number of LSTM cells (256) with only one layer. Our highest accuracy was 92.7% in this model, which was a significant jump from baseline. We removed both '#' and '$' from our dataset, for this model.

The 2-layer LSTM did not perform well in accuracy and loss. We believe such increase in the complexity of model would require more data for training. Figure 2 shows the accuracy, and loss for this model.

## 6 Comparing the Sentiments with Stock Market Returns

To begin, we downloaded the closing prices for the 100 stock ticker symbols mentioned in our labeled dataset of tweets.[4] Then, we calculated the relative daily return for each company, which is an asset's return relative to a benchmark

[4] Of the 100 companies mentioned, we replaced the stock symbols of companies that were owned by another with the symbol of the parent company. Specifically, we replaced $LNKD (LinkdIn) with $MSFT (Microsoft) and replaced $SCTY (Solar City) with $TSLA (Tesla). We also excluded the following companies from the list of 100 companies: VXX, GLD, SPY, GDX, SPX, WFM, EMC, APP, BRCM, and GMCR. These companies were either not currently trading, their trading data could not be found, or they were a specific index. (e.g., S&P 500).

per day. This is the preferred measure of performance for an active portfolio[5], because it is normalized, and because it a stationary time-series, a feature that is essential for most time-series analysis (and specifically, Granger causality). Stationary time-series means that they have a time-invariant mean and variance.

We used the following formula to calculate relative stock return:

$$\begin{aligned} &Stock\ return = \tfrac{(p_1 - p_0)}{p_0} \\ &p_0 = Initial\ stock\ price \\ &p_1 = Ending\ stock\ price \end{aligned} \tag{1}$$

### 6.1 Granger Causality Models

Granger causality (GC) is a probabilistic theory of causality [6] that determines if the information in one variable can explain another.

The advantage of this model is that it is both operational and easy to implement, but it is criticized for not actually being a model of causality (rather, it's a model of increased predictability). Critics have pointed out that even when A has been shown to Granger cause B, it does not necessarily follow that controlling A will directly influence B. Further, nor does it tell us the magnitude of the effect on B. Granger Causality is primarily used for causal notions of policy control, explanation and understanding of time-series, and in some cases, for prediction.

***Formal Definition of Granger Causality:*** A time-series Y can be written as an autoregressive process[6], which means that the past values of Y can, in part, explain the current value of Y. Formally, an autoregressive model is defined as follows:

$$Y_t = \alpha + \sum_{i=1}^{k} \beta_j Y_{t-i} + \epsilon_t. \tag{2}$$

To define his version of causality, Granger introduced another variable X to the autoregressive model, which also has past values like Y.

$$Y_t = \alpha + \sum_{i=1}^{k} \beta_j Y_{t-i} + \sum_{j}^{k} \lambda_j X_{t-j} + \epsilon_t. \tag{3}$$

If adding X improves the prediction of current values of Y, when compared to the predictions from the autoregressive model alone, then X is said to "Granger cause" Y. Technically, Granger causality is an F-test, where the null hypothesis

[5] https://www.investopedia.com.

[6] An autoregressive (AR) model is a representation of a type of random process; as such, it is used to describe certain time-varying processes in nature, economics, etc. The autoregressive model specifies that the output variable depends linearly on its own previous values and on a stochastic term (an imperfectly predictable term); thus the model is in the form of a stochastic difference equation. https://en.wikipedia.org/wiki/Autoregressive_model.

is that all of the $\lambda$ are equal to zero for all $j$. Note that you can also test the reverse case; that is, test whether Y "Granger causes" X. Both causal directions, or none, are possible. Tests for Granger causality should only be performed on stationary variables, which means that they have a time-invariant mean and variance. Specifically, this means that the variables must be I(0)[7] and that they can be adequately represented by a linear AR(p) process[8].

### 6.2 Our Granger Causality Model

$$\textbf{Model (1):} \quad RV \sim Lags(RV, LAG) + Lags(SSC, LAG) \tag{4}$$

$$\textbf{Model (2):} \quad SSC \sim Lags(SSC, LAG) + Lags(RV, LAG) \tag{5}$$

Model one determines if sentiment scores have a causal effect on stock return values, while model two determines if sentiment scores affect stock return values. In both models, the lag ($LAG$) is the number of days the cause precedes the effect, the return value ($RV$) is the calculated daily return for 83 different stocks, and the sentiment scores ($SSC$) are from Table 3.

### 6.3 Three Year Comparison of Social Media Sentiment Analysis and Stock Market Returns

In this section, we performed an in-depth causal analysis for the three stocks most commonly referred to in social media – Apple, Facebook, and Amazon – over a period of three years from 2015–2017. We used our LSTM model Table 3 to assign sentiments to an expanded Twitter dataset, which had 386,251 tweets and covered the same three year period as the stock return values. We then applied the two GC models described in 5 to find causal relationships between the sentiments and return values at five different intervals: fifteen and thirty minutes, one and three hours, and one day. For a particular interval, all of the sentiments in that interval were summed to get an aggregate score. We found causal relationship between tweet sentiments and return values for Amazon and Facebook (in both directions) at fifteen minutes, three hours, and one day. No causal relationships were found for Apple.

Looking more closely at the results of the causality analysis, we see in Tables 5 and 6 that before three hours, the value of the lag fluctuates, but at three hours

[7] In statistics, the order of integration, denoted I(d), of a time series is a summary statistic, which reports the minimum number of differences required to obtain a covariance-stationary series. https://en.wikipedia.org/wiki/Order_of_integration.

[8] The autocorrelation function of an AR(p) process is a sum of decaying exponentials. The simplest AR process is AR(0), which has no dependence between the terms. Only the error/innovation/noise term contributes to the output of the process, so in the figure, AR(0) corresponds to white noise. https://en.wikipedia.org/wiki/Autoregressive_model.

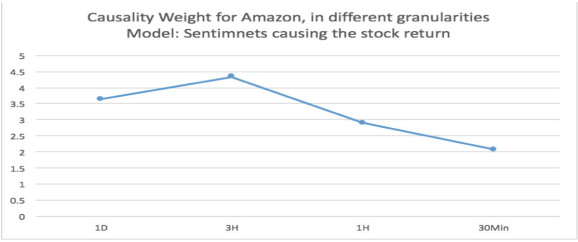
**Table 5.** F-test and P-value for three year data: sentiment causes the stock return

| Stock ticker | Interval | Fvalue | Pvalue | LagNo |
|---|---|---|---|---|
| AMZN | 1D | 3.64 | 0.026 | 2 |
| AMZN | 3 h | 4.33 | 0.013 | 2 |
| AMZN | 1 h | 2.89 | 0.033 | 3 |
| AMZN | 30 min | 2.06 | 0.043 | 7 |
| APPL | 30 min | 2.08 | 0.034 | 8 |
| FB | 15 min | 4 | 0.018 | 2 |
| FB | 3 h | 14.74 | 4.30E−07 | 2 |
| FB | 30 min | 2.31 | 0.04 | 5 |

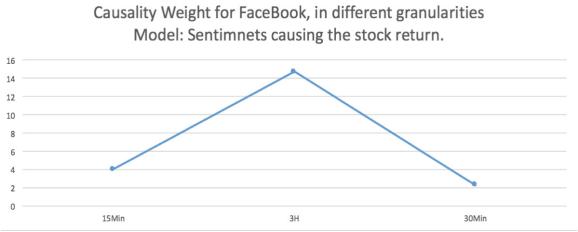
**Table 6.** F-test and P-value for three year data: stock return causes sentiment

| Stock ticker | Interval | Fvalue | Pvalue | LagNo |
|---|---|---|---|---|
| AMZN | 15 min | 4.314 | 0.013 | 2 |
| AMZN | 30 min | 2.069 | 0.043 | 7 |
| AMZN | 1 h | 4.59 | 0.01 | 2 |
| AMZN | 3 h | 11.857 | 7.31E−06 | 2 |
| APPL | 1 h | 2.395 | 0.014 | 8 |
| FB | 15 min | 6.24 | 0.001 | 2 |
| FB | 1 h | 2.633 | 0.032 | 4 |
| FB | 3 h | 6.264 | 0.001 | 2 |

and one day, it stabilizes at a lag of two. We also calculated the causality weight as suggested by Geweke [4], who proved that the linear dependence of a causal model (i.e., the causality weight) can be captured by the F-measure. For both Amazon and Facebook, we found the greatest causality weight at three hours (Figs. 3 and 4). This result, along with the stabilization of the lag at three hours, suggests that we should select an interval of three hours for further analysis. The F-value and the P-value of the analysis is shown in Tables 6 and 5.

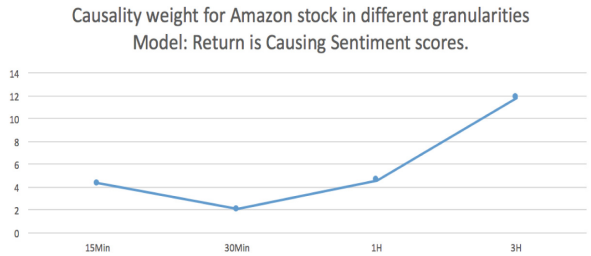



(a) Amazon shows significant causal weight on 30MIN, 1HOUR, 3HOUR and 1DAY intervals.

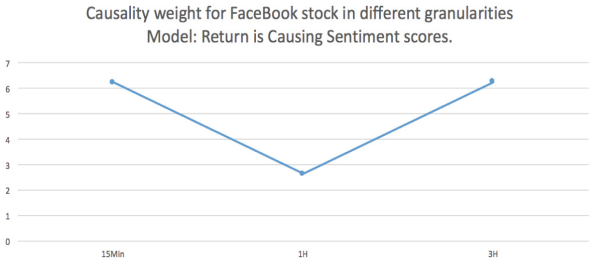



(b) Facebook shows significant causal weight on 15MIN, 3HOUR and 1DAY intervals.

**Fig. 3.** Statistically significant weights for model 1: sentiment causes the stock return. For both stocks, the causality weight was strongest at the 3 h time. The lowest causal weight occurred at 30 min interval.




(c) Amazon shows significant causal weight on 15MIN, 30MIN, 1HOUR, 3HOUR intervals.




(d) Facebook shows significant causal weight on 15MIN, 1HOUR and 3HOUR intervals.

**Fig. 4.** Statistically significant weights for model 2: stock return causes the sentiments. For both stocks, the causality weight was strongest at the 3 h time. The lowest causal weight occurred at 30 min interval for Amazon and 1 h for Facebook.

## 7 Conclusion

In this paper, we first introduced a stock market related tweet dataset that has been labeled by positive or negative sentiments using Amazon Mechanical Turk. In the second part of our paper, we thoroughly compared various deep learning models, and finally introduced our LSTM model with 256 cells, which outperformed all the other models, with accuracy of 92.7%.

While this model has the best accuracy achieved in sentiment analysis of stock market tweets, there are still places for improvement. We suggest some other steps to be added to the pre-processing analysis. For example, it would be interesting to analyze the hashtag-ed words, and figure out if they are a real indicator of a subject or not (e.g. using the frequency of hashtag being mentioned in dataset). If not, they can be separated and considered to be regular words. Also, having a larger tweet dataset would help us to try out other types of deep learning models, e.g. deeper networks. Another attempt in this area could be to create domain focused word embeddings for finance.

In the final part, we analyzed the causal link between our tweet dataset, and the stock market return in different intervals. This is one of the few analyses of causality between tweets and stock prices, the other being [2,11], which has interesting result. In our analysis, we used an expanded dataset of stock return

values that spanned a period of three years, from 2015 to 2017. Because we had a fine granularity of the return values and the sentiments (per minute), we partitioned both our return values and sentiment scores into five intervals: fifteen and thirty minutes, one and three hours, and one day. For each interval, we then used Granger to identify causal relationships between return values and sentiments for three companies: Apple, Facebook, and Amazon. Using Granger causality analysis at the different intervals for Amazon, Facebook, and Apple, we identified significant causal links, at a lag of three hours and one day, for Amazon and Facebook. The strongest causal weight for these two stocks occurred at a three hour lag. Importantly, the causal link existed in both directions: tweets influenced future stock market returns, and stock market returns influenced future tweets. This research can open research areas in social media impact on finance through creation of better datasets and careful analysis of other models of causality.

## References

1. Antenucci, D., Cafarella, M., Levenstein, M.C., Ré, C., Shapiro, M.D.: Using social media to measure labor market flows. NBER (2014)
2. Bollen, J., Pepe, A.: modeling public mood and emotion: twitter sentiment and socio-economic phenomena, pp. 450–453 (2011)
3. Du, S., Xi, Z.: SemEval17.pdf (39), 120–125 (2016)
4. Geweke, J., Geweke, J.: Measurement of linear dependence and feedback between multiple time series measurement of linear dependence and feedback between multiple time Series **77**(378), 304–313 (2018)
5. Graves, A., Mohamed, A., Hinton, G.E.: Speech recognition with deep recurrent neural networks. CoRR abs/1303.5778 (2013). http://arxiv.org/abs/1303.5778
6. Hitchcock, C.: Probabilistic causation. In: Zalta, E.N. (ed.) The Stanford Encyclopedia of Philosophy. Metaphysics Research Lab, Stanford University, Winter 2016 (edn.) (2016)
7. Jiang, M., Lan, M., Wu, Y.: ECNU at SemEval-2017 Task 5: an ensemble of regression algorithms with effective features for fine-grained sentiment analysis in financial domain. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pp. 885–890 (2017). https://doi.org/10.18653/v1/S17-2152, http://www.aclweb.org/anthology/S17-2152
8. Johnson, R., Zhang, T.: Deep pyramid convolutional neural networks for text categorization. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 562–570 (2017). https://doi.org/10.18653/v1/P17-1052, http://aclweb.org/anthology/P17-1052
9. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR abs/1412.6980 (2014). http://arxiv.org/abs/1412.6980
10. Kolchyna, O., Souza, T.T.P., Treleaven, P., Aste, T.: Twitter sentiment analysis: lexicon method, machine learning method and their combination, p. 32 (2015). http://arxiv.org/abs/1507.00955
11. Kouloumpis, E., Wilson, T., Moore, J.: Twitter sentiment analysis: the good the bad and the OMG! In: Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM 2011), pp. 538–541 (2011). http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/download/2857/3251?iframe=true&width=90%25&height=90%25

12. Lillo, F., Miccichè, S., Tumminello, M., Piilo, J.: How news affect the trading behavior of different categories of investors in a financial market. Papers.Ssrn.Com (April), 30 (2012). https://doi.org/10.1080/14697688.2014.931593, http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2109337
13. Loughran, T.I.M., Mcdonald, B.: When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. J. Financ. (2010, forthcoming)
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. CoRR abs/1301.3781 (2013). http://arxiv.org/abs/1301.3781
15. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space, pp. 1–12 (2013). https://doi.org/10.1162/153244303322533223, http://arxiv.org/abs/1301.3781
16. Pennington, J., Socher, R., Manning, C.D.: GloVe: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, pp. 1532–1543 (2014). https://doi.org/10.3115/v1/D14-1162
17. Ranco, G., Aleksovski, D., Caldarelli, G., Grčar, M., Mozetič, I.: The effects of twitter sentiment on stock price returns. PLoS One **10**(9), 1–21 (2015). https://doi.org/10.1371/journal.pone.0138441
18. Ruan, Y., Durresi, A., Alfantoukh, L.: Using twitter trust network for stock market analysis. Knowl.-Based Syst. **145**, 207–218 (2018). https://doi.org/10.1016/j.knosys.2018.01.016, http://www.sciencedirect.com/science/article/pii/S0950705118300248
19. dos Santos, C.N., Gatti, M.: Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. Coling-2014 pp. 69–78 (2014)
20. Severyn, A., Moschitti, A.: Twitter sentiment analysis with deep convolutional neural networks. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR 2015, pp. 959–962 (2015). https://doi.org/10.1145/2766462.2767830, http://dl.acm.org/citation.cfm?doid=2766462.2767830
21. Snow, R., Connor, B.O., Jurafsky, D., Ng, A.Y., Labs, D., St, C.: Cheap and fast - but is it good? Evaluating non-expert annotations for natural language tasks, pp. 254–263, October 2008
22. Sohangir, S., Wang, D., Pomeranets, A., Khoshgoftaar, T.M.: Big data: deep learning for financial sentiment analysis. J. Big Data **5**(1) (2018). https://doi.org/10.1186/s40537-017-0111-6
23. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. CoRR abs/1409.3215 (2014). http://arxiv.org/abs/1409.3215
24. Tabari, N., Seyeditabari, A., Zadrozny, W., Tabari, N.: SentiHeros at SemEval-2017 task 5: an application of sentiment analysis on financial tweets, pp. 857–860 (2017)
25. Zhao, X., Wang, C., Yang, Z., Zhang, Y., Yuan, X.: Online news emotion prediction with bidirectional LSTM, pp. 238–250 (2016)