Stock Price Prediction using News

Ajit Sarkaar
Department of Computer Engineering
Virginia Tech
Blacksburg, VA 24060
ajitsarkaar@vt.edu

Ashish Kotian
Department of Computer Engineering
Virginia Tech
Blacksburg, VA 24060
ashishshk@vt.edu

Soumya Kumar
Department of Computer Engineering
Virginia Tech
Blacksburg, VA 24060
soumyakumar@vt.edu

Prerit Jain
Department of Computer Engineering
Virginia Tech
Blacksburg, VA 24060
preritjain8932@vt.edu

Abstract

In a financially volatile market such as the stock market, it is necessary to have a precise prediction of future price trends. Investing in stock market carries risk, but if approached in a disciplined manner secure prediction of the values of the stocks can be made. Making such predictions requires the implementation of advanced algorithms of machine learning. This project report contains studies with different machine learning algorithms such as LSTM (Long Short-Term Memory) and GRU (Gated Recurrant Unit). The experimental results show an accuracy of 54.4% when implemented using GRUs and an accuracy of 51.43% when implemented using LSTMs.

1 Introduction

In recent years, it has become very important to analyze all sorts of data to get the profits of investing in the stock market. However, prediction of stock market prices involve a lot of factors ranging from physical factors, psychological factors, news, current trends, trade policies etc. The ubiquity of data today enables investors, at any scale, to make better investment decisions. The main challenge in using machine learning to predict a stock price is ingesting and interpreting the data to determine which data is useful.

In this project, non-quantifiable data such as financial news about companies is taken into consideration and the future stock price trends are predicted in order to make calculated decisions on which stocks to invest into. Assuming that the news articles have an impact on the stock market, this is an attempt to study the relationship between news and stock trends. The findings will add value in determining whether the management's tone and characterization of a firm's trajectory is accurate. From a practical standpoint, this project aims to develop machine learning models for quantitative finance firms seeking to forecast the financial future of investments.

This project report is organized as follows: In section 2, we present the related work for this topic. In section 3, we discuss about the methods used to implement this project. In section 4, we present our experimental results and the reason for achieving those results. In section 5, we mention some changes that we plan on introducing in order to improve our results. Finally, in section 6 we conclude our work.

2 Related Work

The previous research on sentiment analysis of short texts includes emotional knowledge-based methods and feature-based classification methods. The former mainly focuses on the extraction and the sentiment classification based on opinion-bearing words and opinion sentences while the latter focuses on the sentiment classification based on features.[1][4]
With the development of deep learning algorithms, typical deep learning models such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have achieved remarkable results in computer vision and speech recognition. Word embeddings, CNNs and RNNs have been applied to text sentiment analysis and achieved remarkable results. furthermore, some studies have utilized recursive neural networks to construct the sentence level representation vector in sentiment analysis.

3 Method

The following flowchart gives an overview of the implementation methodology:

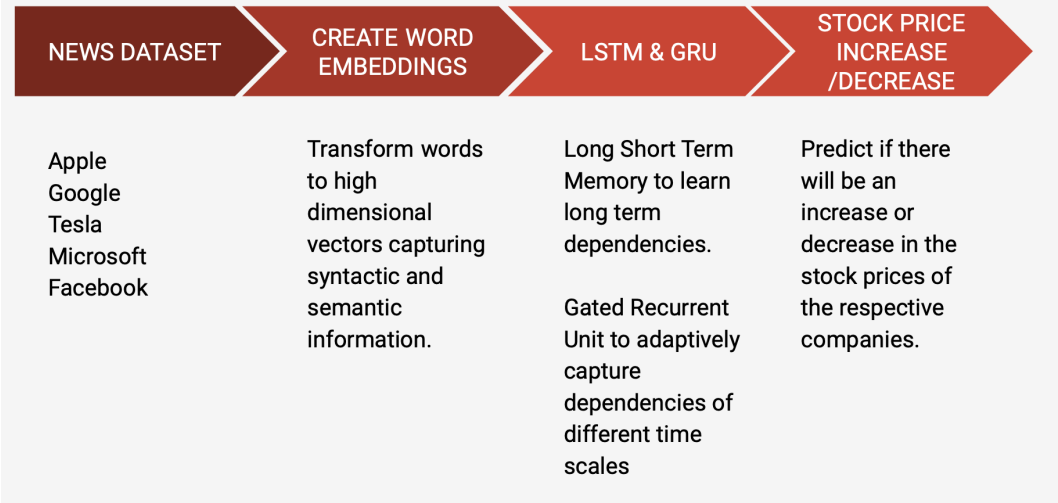


Figure 1: System Flow

The financial news data set is taken for five companies - Apple, Google, Tesla, Microsoft and Facebook. The word embeddings are created using GloVe for the news articles, which is fed to the neural networks and a qualitative as well as quantitative prediction is made regarding the stock price. Two model approaches have been used in this project, one is based on implementing the Gated Recurrent Unit (GRU) and the other is based on implementing Long Short-Term Memory (LSTM)[3]. Each of the following blocks have been explained in detail in the following sections.

3.1 Dataset

The data for this project comes from a dataset on Kaggle, and covers nearly eight years (2008–08–08 to 2016–07–01). This data basically comes from the market data provided by Intrinio and the news data provided by Thomson Reuters. This has been included in the following files: the Combined-NewsDJIA.csv, DJIAtable.csv and RedditNews.csv.

To obtain the change in prices for a particular day D, the closing price on day d was subtracted from closing price on day D+1 to account for fluctuations that were a result of the news released on any given day. The news mentioning only the six companies in their headlines or summary were considered and stock prices were mapped to each news sample using a left join merge operation.

3.2 Word Embeddings : GloVe

GloVe is used for generating word embeddings. It is essentially a log-bilinear model with a weighted least-squares objective. The model rests on the idea that ratios of word-word co-occurrence prob-

abilities have the potential for encoding some form of meaning which can be encoded as vector differences[9]. Therefore, the training objective would be to learn word vectors such that their dot product equals the logarithm of the words' probability of co-occurrence. In Keras, the embedding matrix is represented as a "layer" and maps positive integers in this case, indices corresponding to words, into dense vectors of fixed size (the embedding vectors).
It can be trained or initialized with a pre-trained embedding. The training set is quite small, hence the word embeddings were not updated but their values were left fixed.

The Embedding() layer takes an integer matrix of size (batch size, max input length) as input. This corresponds to the sentences that are converted into a lists of indices (integers), as shown in the figure below.

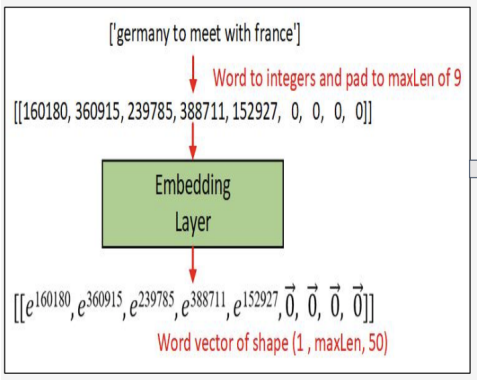


Figure 2: Glove

3.3 Model Approach 1:

The task of this model is to take the news string sequence and make a binary classification whether the Dow Jones' close value will rise or fall as compared to the previous day's close value. It outputs "1" if the value rises or stays the same and outputs "0" if the value decreases. The input text is pre-processed before being analyzed. All 25 news articles are concatenated to one long string for each day. These sentences are converted to lower-case, and characters, such as numbers and punctuation, that cannot be represented by the GloVe embeddings are removed. The next step is to convert all the training sentences into a lists of indices and zero-padding those lists, so that their length is same.

Layer (type)	Output Shape	Param #
embedding_7 (Embedding)	(None, None, 50)	20000050
gru_14 (GRU)	(None, None, 128)	68736
gru_15 (GRU)	(None, None, 128)	98688
gru_16 (GRU)	(None, 128)	98688
dense_2 (Dense)	(None, 1)	129
activation_2 (Activation)	(None, 1)	0

Total params: 20,266,291
Trainable params: 20,266,291
Non-trainable params: 0

Figure 3: Model Used for Approach 1

This model is built such that it can process word sequences in a simple implementation. It includes the pre-trained embedding layer, followed by the three stacked GRU layers, and finally a dense layer that generates the final output with sigmoid activation. The optimizer used in this case was ADAM. The model was experimented with the maximum sequence length of 500, 600 and 700. Finally, the maximum sequence length was set to 500 in order to cover the majority of text across all input samples, while maintaining a relatively short training time. Since this model uses GRUs instead of LSTMs, it is less computationally expensive to train.

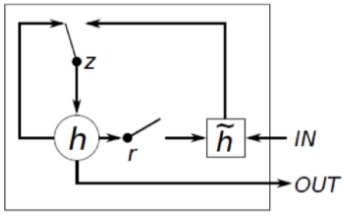


Figure 4: A typical GRU unit (Wang et al.)

3.4 Model Approach 2:

The second approach is inspired by Wang et al [1]. This approach uses a combination of convolutional layers (local and deep features) and LSTMs (distant features). The two input branches was designed to use different sized kernels (filters) for convolution operations in order to obtain improved results. Mean Squared Error (MSE) was used for training with ADAM, and Mean Absolute Error (MAE) was used for evaluation.

Layer (type)	Output Shape	Param #	Connected to
embedding_41_input (InputLayer)	(None, 250)	0	
embedding_42_input (InputLayer)	(None, 250)	0	
embedding_41 (Embedding)	(None, 250, 300)	9119400	embedding_41_input[0][0]
embedding_42 (Embedding)	(None, 250, 300)	9119400	embedding_42_input[0][0]
dropout_101 (Dropout)	(None, 250, 300)	0	embedding_41[0][0]
dropout_103 (Dropout)	(None, 250, 300)	0	embedding_42[0][0]
conv1d_41 (Conv1D)	(None, 250, 32)	28832	dropout_101[0][0]
conv1d_42 (Conv1D)	(None, 250, 32)	48032	dropout_103[0][0]
dropout_102 (Dropout)	(None, 250, 32)	0	conv1d_41[0][0]
dropout_104 (Dropout)	(None, 250, 32)	0	conv1d_42[0][0]
lstm_41 (LSTM)	(None, 256)	295936	dropout_102[0][0]
lstm_42 (LSTM)	(None, 256)	295936	dropout_104[0][0]
add_21 (Add)	(None, 256)	0	lstm_41[0][0] lstm_42[0][0]
dense_21 (Dense)	(None, 256)	65792	add_21[0][0]
dropout_105 (Dropout)	(None, 256)	0	dense_21[0][0]
output (Dense)	(None, 1)	257	dropout_105[0][0]

Total params: 18,973,585
Trainable params: 18,973,585
Non-trainable params: 0

Figure 5: Model Used for Approach 2

ReduceLROnPlateau was used to decrease the learning rate when the validation loss stops decreasing. Along with this, both grid search as well as random search was used. This model is described in detail in the following discussion.

A common LSTM unit is composed of a cell, an Input gate, an Output gate and a Forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. This is shown in Figure 6.

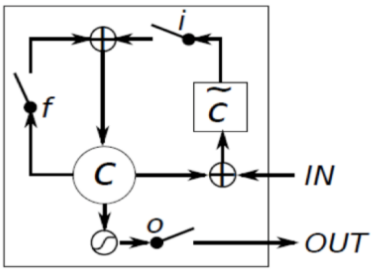


Figure 6: A typical LSTM unit (Wang et al.)

The Long Short-Term Memory (LSTM) was first proposed by Hochreiter and Schmidhuber (1997) and it can learn long-term dependencies. See Figure 6 for the graphical illustration.
Different from traditional recurrent units, LSTM units keeps the existing memory $c_t \in R^n$ at time t. The input at time t is x_t, h_{t-1}, c_{t-1}, the output is h_t, c_t, they can be updated by the following equations:

$$
\begin{aligned}
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\
g_t &= tanh(W_g x_t + U_g h_{t-1} + b_g) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot tanh(c_t)
\end{aligned}
$$

where $\sigma(\cdot)$ denotes the logistic sigmoid function. The operation $\odot$ denotes the element-wise vector product. At each time step t, there is an input gate i_t, a forget gate f_t, an output gate o_t, a memory cell c_t and a hidden unit h_t. h_0 and c_0 can be initialized to 0 and the parameters of the LSTM is W, U, b.

Convolution is widely used in sentence modeling[5][7][8]. The structure of convolution varies slightly in different research fields. The convolutional layer applies a matrix-vector operation to each window of size w of successive windows in the sentence-level representation sequence. We apply the max and average pooling operations and find that the former performs better with less computational complexity.

Next, a pairwise max pooling operation is applied over the feature map to capture the most important feature. The pooling operation can be considered as feature selection in natural language processing. The max pooling operation is shown in Figure 7. This is then fed to the neural network and the change in opening price is calculated.

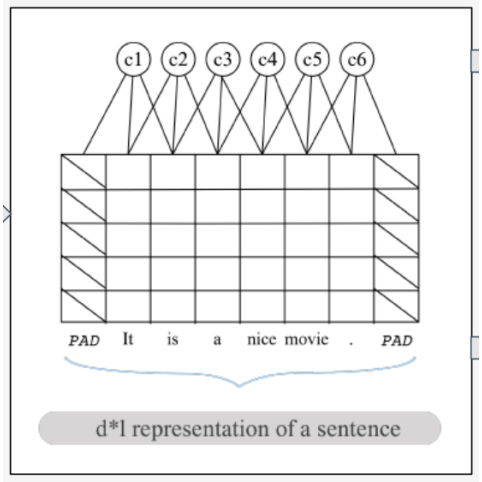


Figure 7: Convolution and Pooling (Wang et al.)

4 Results

Both the implementations were analyzed and the accuracy was predicted for the stock prices. For the implementation using the Gated Recurrent Unit, an Area Under Curve - Receiver Operating Characteristics (AUC-ROC) curve has been generated. It is one of the most important evaluation metrics for checking the performance of any classification model's performance. This curve tells how

much a model is capable of distinguishing between classes.
It can be seen from the Figure 8 that this implementation gives an accuracy of 54.4%. The state of the art implementation gives an accuracy of 60% in predicting the stock price.

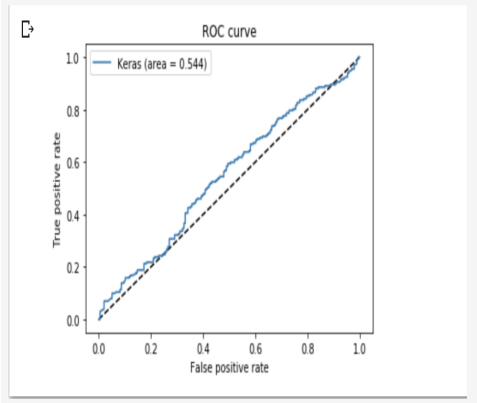


Figure 8: AUC-ROC Curve for GRU Implementation

For the implementation using the LSTM model, the first experiment involved predictions of just an increase or decrease classes in the stock price trend. This model achieved an accuracy of about 48.57%. Ground Truth vs the Predicted Value plot has been generated. All the green dots represent the ground truth value and the red ones represent the predicted value. When the dots are at 1, it shows that the price of stocks are going to increase and when the dots are at 0, it shows that the price of stocks are going to decrease.

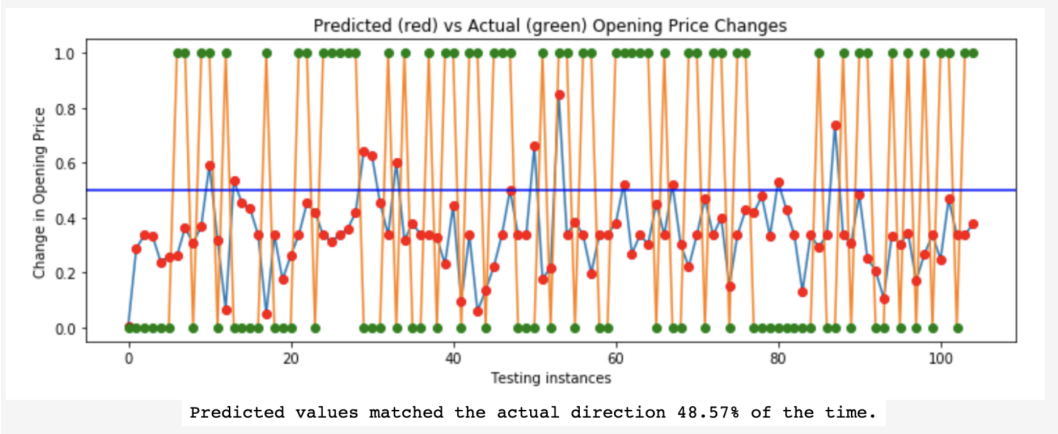
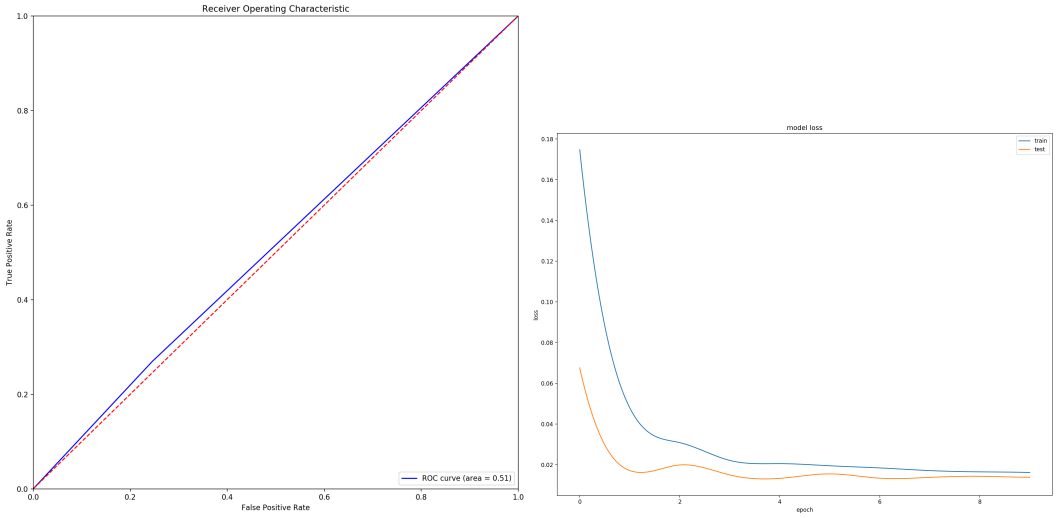


Figure 9: Preliminary results for LSTM Implementation

In subsequent experiments, the model was trained on normalized price fluctuations, to explore the quantitative effect of how much perturbations the news of a company caused in their corresponding stock price. Figure 12 shows the ground truth trend of data in blue and the trend of stock prices predicted by the model in red. This figure is taken from the best performing model, which achieves an accuracy of 51.43%.

	Actual Increase	Actual Decrease
Predicted Increase	0.133	0.124
Predicted Decrease	0.362	0.381

Figure 10: Confusion matrix for Approach 2



(a) ROC curve (blue) for Approach 2 (b) Training and validation losses for Approach 2

Figure 11: Results for Approach 2

The ROC curve and losses are shown in Figure 11. The AUC score for Approach 2 is 0.51 as shown by Figure 11 (a). Compared to the state of the art at around 60%, the accuracy of Approach 2 still lags a bit behind, which may be due to unexplored features to be considered in the future work. Figure 10 also shows the confusion matrix for the best performing model, with binary classification, for predicted vs actual increase and decrease of prices.

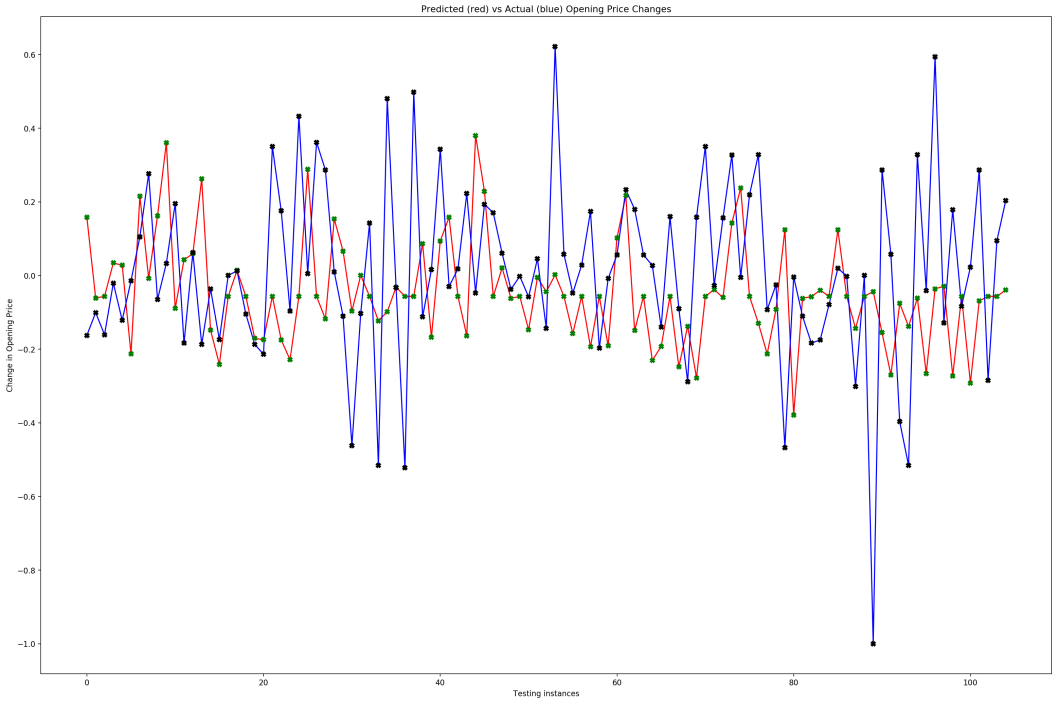


Figure 12: Actual trend (blue) VS Predicted trend (red)

It can be observed from these results that the implementation using GRU achieves better accuracy than the implementation using LSTM. The reason for achieving such low accuracy in predicting the

stock price is that a great deal of data and emotions are factored into its value that can not be covered by just using the financial news dataset.

5 Future Work

In order to improve the accuracy, the plan is to extend the work to use additional news data such as change in trade policies, development in marketing strategies etc. the aim is to introduce more features(oil supply, weather information) to try and reduce stochasticity. This may or may not improve results. Data augmentation and pre-processing can also be used for better representation and usage of relevant embeddings. Using the Keras functional API to form a multi input Keras model, to account for certain other factors that lead to fluctuation of the stock market such as the Moving Average Convergence/Divergence Oscillator.

6 Conclusion

The stock predictor system was successfully implemented using GRU and LSTM. GRU based model has been implemented in order to achieve high computational results as compared to the LSTM based approach. While, the LSTM model allows to harness the advantage of the memory unit present in the LSTM architecture which is of crucial importance in terms of financial news data[6]. It enables the system to extract and store meaningful information from the news articles while ignoring the unimportant features or words and use it as feedback to improve the model. The GRU- based model learned faster than the LSTM model considering the data set that was used for training. The accuracy achieved for implementation using GRU is 54.4% and the accuracy achieved for implementation using LSTM is 51.43%. The results show that the implementation using GRU performed better in predicting the stock price movements as compared to LSTM.
Along with the literature survey, the study and implementation of the combination of GRU and LSTM layers used in constructing the model was carried out by Soumya Kumar. The implementation of the GRU-based model was carried out by Ashish Kotian and different hyper-parameters were tested in order to observe changes in the accuracy and achieve better results[2][5]. The implementation of the LSTM-based model was carried out by Ajit Sarkaar. He implemented the combination of convolutional and LSTM layer along with designing two input branches to use different sized kernels for conv operations to obtain improved results. The study and implementation of Glove Embedding was done by Prerit Jain.

References

[1] Xingyou Wang, Weijie Jiang Zhiyong Luo (2016) Combination of Convolutional and Recurrent Neural Network for Sentiment Analysis of Short Texts. *Coling 2016*

[2] Abdalraouf Hassan (2018) Deep Neural Language Model for Text Classification Based on Convolutional and Recurrent Neural Networks.

[3] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR 2014*

[4] Hang Cui, Vibhu Mittal, and Mayur Datar (2006) Comparative experiments on sentiment classification for online product reviews. *In AAAI 2006*

[5] Yoon Kim (2014) Convolutional neural networks for sentence classification. *EMNLP 2014*

[6] Phong Le and Willem Zuidema (2015) Compositional distributional semantics with long short term memory. *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics 2015*

[7] Cıcero Nogueira dos Santos and Maira Gatti (2014) Deep convolutional neural networks for sentiment analysis of short texts. *COLING, pages 69–78*

[8] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom (2014) A convolutional neural network for modelling sentences. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics 2014*

[9] Pennington, Jeffrey Socher, Richard Manning, Christoper (2014) Glove: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*