SENTIMENT ANALYSIS FOR FINANCIAL MARKET PREDICTION

T. H. H Methmal

168246D

Degree of Master of Science in Computer Science

Department of Computer Science and Engineering

University of Moratuwa
Sri Lanka

April 2020

SENTIMENT ANALYSIS FOR FINANCIAL MARKET PREDICTION

Thommaya Hewa Hasanga Methmal

168246D

Dissertation submitted in partial fulfillment of the requirements for the degree Master of Science in Computer Science specializing in Data Science

Department of Computer Science and Engineering

University of Moratuwa
Sri Lanka

April 2020

DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: …………......... Date: …………..........

Name: T. H .H Methmal

I certify that the declaration above by the candidate is true to the best of my knowledge and that this report is acceptable for evaluation for the CS6997 - MSc Research Project.

The above candidate has carried out research for the Masters Dissertation under my supervision.

Signature: …………......... Date: …………..........

Name: Dr. Surangika Ranathunga

ABSTRACT

Today's world is highly dependent on financial markets. Financial markets are very dynamic, making it difficult to predict prices. If we can make good predictions, we can make financial gains without any risks. There are two main categories of data that we can use to predict the market price; historic market data and textual data. Most importantly, textual analysis on sources like news, social media and reports is more popular today among researchers. This process can be introduced as sentiment analysis. The whole idea behind sentiment analysis is checking the opinion behind the text; whether it is a positive, negative, or neutral polarity.

This research focuses on sentiment-analysis-based financial market prediction using deep leaning. Market prediction using sentiment analysis is a very challenging task. There are complex linguistic issues to solve and using a microblog dataset like Twitter for the prediction task makes it even more difficult. However, the current prediction approach rarely exceeds the seventy percent accuracy mark. This research is based on the SEMEVAL 2017 fifth task and will use the same dataset shared by the SEMEVAL team.

This thesis presents an improved version for above mention reported baseline. We experiment with different techniques in both machine leaning and deep learning domains. Lexicon based dictionaries are heavily used here in each model since this is a small dataset with train set (1693) and test set (793). We had to enlarge the dataset as much as possible to achieve good accuracy. We created mainly four models which are based on machine learning and deep leaning techniques. Support vector regression algorithm is used for the machine leaning model. Also we used convolutional neural network (CNN) , long short term memory (LSTM) and gated recurrent unit (GRU) as deep leaning architectures which are performed better than any of the baseline models on this dataset. Our deep leaning models are achieved maximum similarity scores than any of the single system.

Finally, we experiment three main ensemble techniques which are Averaging Linear Regression and multilayer perceptron. We achieved best results from averaging ensemble model.

Keywords: Financial Sentiment Analysis; Opinion Mining; Reviews; Text Analysis; Deep learning; Sentiment Analysis Challenges; Semeval.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Dr Surangika Ranathunga for the continuous support from the first day of my M.Sc. study, for her patience, motivation, enthusiasm, and immense knowledge. Besides my supervisor, I would like to thank the rest of my teachers throughout the journey for guiding me towards the right path in life. Last but not least, I would like to thank my mother, without whom I would not be here, and for her support in guiding me spiritually throughout my life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

GDP	Gross Domestic Product
FOREX	Foreign Exchange Market
MSH	Morgan Stanley High-Tech Index
LDA	Latent Dirichlet Allocation
IG	Information Gain
CHI	Chi-square Statistics
DF	Document Frequency
TF	Term Frequency
IDF	Inverse Document Frequency
SVM	Support Vector Machine
CATS	Categorization And Trading System
SVC	Support Vector Classification
SVR	Support Vector Regression
RBF	Radial Basis Function Kernel
OLS	Ordinary Least Square
GI	The General Inquirer
LMFSD	Loughran-McDonald financial sentiment dictionary
CBOW	continuous bag-of-words
SG	skip-gram
GLOVE	Global Vectors
ABR	AdaBoost Regressor
BR	Bagging Regressor
RF	Random Forest
CNN	Convolutional Neural Network

Chapter 1 : Introduction

1.1 Background

The modern world today highly depends on the financial market. So it is very important to have an idea about the financial market and its movements. If we can predict the financial movements correctly, we can make financial gains without financial losses. Unfortunately, it is very difficult to predict the nature of the financial market.

We can categorize predictive measures as either a technical or fundamental analysis, based on the input data provided. For example, previously, historic market data was used, whereas later, news based on the market is increasingly being used. In early days of market prediction, researchers mostly used the technical analysis. Early research was done using technical quantitative methods on historical market data. However, fundamental analysis on unstructured data like news, reports or social media textual analysis is more challenging. Nowadays, research focuses on textual data available online, such as social media, news, blogs, and forums.

1.2 Sentiment Analysis

Sentiment analysis is determining the emotional tone behind the text, used to understand the attitude, opinions, and emotions of a series of words. This is also known as opinion mining [1]. This is extremely useful for analyzing public opinion on certain topics that are mainly discussed across social media. Nowadays, there are tools introduced by researches to do real time sentiment analysis, which is quicker and easier. The applications of sentiment analysis are of a wide range. The most popular application is where most organizations extract opinions from social media to be more successful in the market. They consistently look at the customer satisfaction of their products and services[2].

The basic task of the sentiment analysis is to classify texts based on their polarity. It is to check whether the opinion of the text is of a positive, negative, or neutral polarity. Polarity is mainly measured by words such as happy, sad, angry, etc. For example, we can get a general idea about a particular movie, based on the sentiment analysis of the reviews.

Financial data is more challenging because the presented sentiments and opinions can affect market dynamics [3][4]. News articles frequently include market specific information such as projects, company merges, acquisitions, etc. Good news can lift market values. Analyzing the public sentiment on market movement is a very powerful method to predict the market reaction [4][5].

1.3 Motivation

Accuracy of recently used machine learning based and deep learning sentiment analysis approaches for financial market prediction rarely exceeds seventy percent [6][7]. It is very challenging because there are not well structured messages most of the time. As well as most of the time they are full of sarcasm and irony. Furthermore, short length texts like microblogs can be very informative and challenging to extract features, due to the different vocabularies used [7][8]. It is beneficial to enhance the quality of the sentiment analysis on financial domain for large audiences such as investors, mangers, and brokers. This highly interesting and valuable research will motivate the public to develop more innovative services and products.

1.4 Objective

The SEMEVAL 2017 fifth task has provided a well-defined dataset. This is a microblog dataset collected from Twitter and Stocktwits social media platforms. The proposed task is to predict a sentiment score for the microblog messages. Scores are in range of -1 and 1. Neutral messages are assigned zero as their sentiment score [9].

The main objective is to use the SEMEVAL 2017 fifth task data set and develop state of the art classification model for sentiment analysis in stock market prediction.

1.5 Thesis Contribution

Most of the researches are framed into either a machine learning model with handcraft features or a deep learning model with word embedding features. This research is combined both word embedding features and handcraft features and introduce state of art hybrid feature set. It is performing well with deep learning models like GRU , LSTM and MLP and beat all existing baselines.

1.6 Thesis Organization

The chapter two described the related literature and overview of a past research work. All the challenges and trends discuss there. Such as how researchers handled input data, preprocessing, feature extraction and training in the past. The chapter three is discussed the methodology which we implement. In chapter four presents all the experiment results which we achieve from our model by comparing them. Chapter five discuss all the results and methodology which we use by comparing existing baselines and possible directions to further improvements.

Chapter 2 : Literature Review

Supervised learning is a popular method in past researches. All of the existing systems for stock market predictions based on sentiment analysis represent some of the components in Figure 2.1. When we input textual data to the system from one side, some market predictive scores are produced as the output [10]. After finding a well-defined annotated dataset, we need to preprocess the dataset according to the problem. First of all, selecting the appropriate feature set is an important step. Then, we need to pick the key features from the selected features. We can use a feature representation method to input to the machine learning algorithm. After that ,the model is going to be trained using training dataset. After that, we can test the model using a test data set.

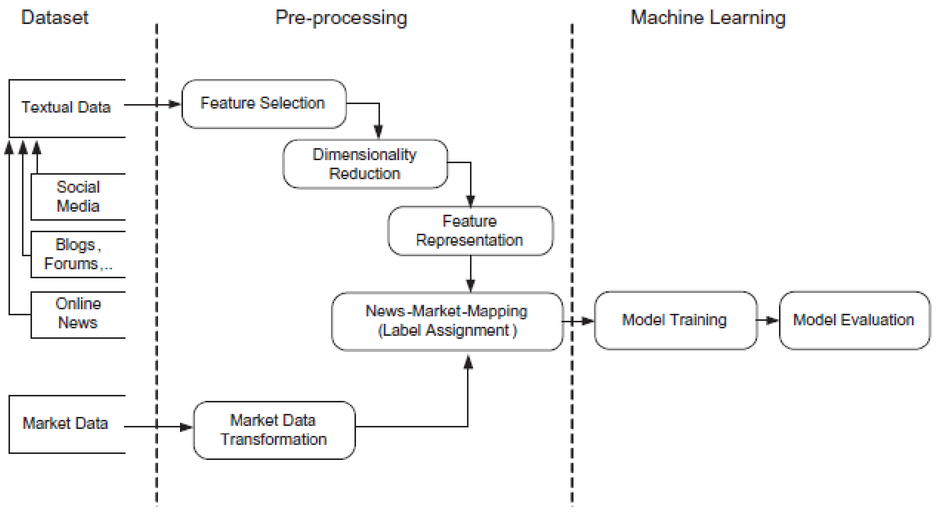


Figure 2.1. System components diagram for machine leaning models [9].

2.1 Input dataset

Mainly there are two sources for data such as textual data and market historical data.

2.1.1 Textual Data

Mainly textual data extracted from popular websites like Wall Street Journal [11], Financial Times [12], Reuters [13], Dow Jones, Bloomberg [14], Forbes, Yahoo! Finance [15], etc. News could be both general news and financial news. However, financial news is preferred due to less noise when compared to general news. Again, we could have extracted either news texts or headlines, but headlines are easier to analyze because they are less noise and on point.

More recently, the most popular way to do a sentiment analysis was based on social media like Twitter, stocktwist, blog posts and market predictions [16]. Most researchers have chosen only Twitter because it express the public mood in a very efficient way.

The next category of textual resources are company reports and press releases. An important fact about these sources is that they are limited to a preschedule time[10]. For example, if the first quarter of a company report is used to predict the results of the third quarter, it is not a very accurate analysis. Apart from that, these sources may be unstructured or semi-structured, such as certain government announcements made regarding unemployment percentages or the Gross Domestic Product.

2.1.2 Market Data

Market data is numeric values which is in price points or indexes from the financial market. Those datasets are mostly used to directly train the machine leaning algorithms where one variable is considered as a single feature. Even though they use both stock market prediction and the foreign exchange market (FOREX), past researches mostly focused on stock market. Prediction can consist of an index like Dow Jones Industrial Average [17], the US NASDAQ Index[18] , Morgan Stanley High-Tech Index (MSH) [19], the Indian Sensex Index [20], and S&P 500 [21]. We can also directly predict the stock market price of a company like Apple, Google, or Intel. We can make a solid prediction based on the stock market rather than foreign exchange market, as the FOREX market is very dynamic.

2.2 Pre-processing

Once we have all the textual data needed to create a module in a format that can be input to all machine learning algorithms, we need to restructure the data in a structural manner. Usually, in data mining and specifically in text mining, preprocessing data is crucial in obtaining a good outcome of the results. The preprocessing process can be divided into at least three sub processes, which show as past work [10].

2.2.1 Feature Selection

These are representative of the text documents and should be selected based on how it is best conveyed with regard to the document content. This is crucial, and if incorrect features are input into the model, the expected output would be meaningless.

2.2.1.1 Bag of words

This is the most popular feature and almost seventy percent of text mining based research uses it. It consists of splitting the text into words and consider each term as a feature. All stop words (such as e.g. a, and, the) are ignored, like determiners, prepositions, etc. The order of word and their co-occurrence within the context is completely ignore. One of the advantage of the bag of word we can treat it as a subset of terms which is more helpful to express the meaning of an article. [22].

2.2.1.2 Noun Phrases

A noun and any word in the sentence that modifies the noun can be named as a noun phrase. Noun phrases are creating with the help of POS tags. Firstly identified the noun from the POS tag and then concatenate surrounding parts following grammar rules. [23].

2.2.1.3 Named Entities

Named entities are the more abstract view of the bag of word model and noun phrases. Using semantic hierarchy noun and noun phrases can be classified as a person, organization, location, date, time money and percent. Likewise, the first name entity set had seven types. This is a predetermined entity classification method and is known as the MUC-7 framework [24]. Later, it has been extended to include 200 categories [15]. Entities can be tagged in different ways. Taggers have lexical and syntax information. They are using sample entities and words patterns to tag. Based on this information, we can introduce a machine learning or rule based approach to the tagger. When the input text matches saved sample patterns, entity tag will be assigned to that text. The advantage of having name entity tags is that we can get an idea of the converge of a text by identifying special phrase types. Those noun entities and phrases may not independently performing well in the microblog messages. But they will definitely helpful to improve the accuracies with other combination of feature set. [24].

2.2.1.4 N-Gram

This model is very popular in speech recognition. However, it is currently used in other application areas as well, such as informational retrieval, machine translation, optical character recognition, and spelling correction. The objective is to predict the probability of naturally occurring word sequences. Probabilities are high when a set of words naturally occur together often. We can use simple n-grams to identify the correlated words in the documents. It is not just a bag of words anymore [25].

2.2.1.5 Latent Dirichlet Allocation (LDA)

This method is not very popular. This is a conceptual technique which is categorizing word into concepts. If we want to find themes or topics in documents and do not need any supervised topic classification, we can use LDA. This is essentially a clustering problem of both words and documents. Typically, one document may exhibit many topics. When we input a document to this technique, LDA try to extract set of topics and surrounding words from that topic and it will create a specific collection of that topic based on the information that it extracted. For example, the LDA model may have topics that are classified as Bearish and Bullish. This means that stock prices are either going up or down. Those topics have probabilities of generating words as “high, increase”, which can be classified as bullish related. Likewise, bearish topic has probabilities of generating each word as “low, decrease”. Unigram bag-of words is used as the model for LDA. Jin et al [14] have discussed the currency trend modeling based on the news article. Mahajan et al [26] show how the market events are impacted based on the stock prices.

2.2.1.6 Discussion

This is a major step in the sentiment analysis because it is very important to have an idea about the performance of the feature set. As discussed earlier, many researchers used the bag of words feature, which is not that effective in prediction. As isolated words may not be capable of expressing the underlying meaning of the text, selecting a more complex and expressive feature would be more accurate in prediction. So according to the past review work, the N-gram model with word combinations, which is present in Hagenau et al [27], gave up to 76 % accuracy.

2.2.2 Feature Reduction

We should keep well performed feature set aside. If we are overloaded by many features, it is very hard to identify the most important features of the data. Most algorithms will not perform well in that situation. This is known as the curse of dimensionality. Let's discuss what the feature reduction methods and their applications are, based on past review work.

2.2.2.1 Wordnet Thesaurus

Wordnet shows the hierarchical relations between words. This is one of the expectations of dimension reduction as well. Zhai et al [28] use the binary vector space to model the feature set. First, stop words are removed from the document and the remaining word tags are replaced with part-of-speech tags. All the higher level concepts replace. POS taggers help to disambiguate the words when assigning word phrases to WordNet. TF and IDF used to weight the concepts. If term occurs more often on rarer in the document, its assign a higher weight. For example, unique concepts for good news are: developing, increasing, sustaining etc. Bad news words are: reducing, down etc.

2.2.2.2 Selecting Top N Words

Term Frequency (TF), Inverse Document Frequency (IDF), and their product (TF* IDF) are commonly used to measure the importance of a word. Most important feature set is top n words with highest score. Mittermayer [29] does the feature reduction for his bag of word model by selecting the 1000 meaningful terms. Then, he builds the vector representation of local dictionaries for three categories in the feature extraction phase.

2.2.2.3 Minimum Occurrence per Document

This is common technique which is used by most of the researchers. They define minimum occurrence value for terms. (Butler & Kešelj[14]; Schumaker & Chen [24]). This is done to cut down the noise of rarely used words.

2.2.2.4 Pre-defined Dictionaries

The next common approach is using dictionaries predefined by experts who have the domain knowledge. In Wuthrich et al's [12] paper they used a set of keyword tuples such as "bond strong", "dollar falter", "property weak", "dowrebound", and "technology rebound strongly". Then, they got the number of the occurrence of the keyword tuples in the news of each date. Those values are converted into weights and based on that, rules are generated to predict the market trend. Rachlin et al [23] used an automatic keyword extractor, which is a engine to extract all important keywords. It analyzes those keywords and their behavior in the domain and specific document. It then finds the most influential words in each document. Sometimes, dictionaries are more specific to the fields of use. In the work of Tetlock et al. [30], the psychology field related dictionary named Harvard-IV-4 is used.

2.2.2.5 Other Dimensionality Reduction Methods

There is another usual method used by most of the past review work for feature reduction. Activities like "url" removing , transform all abbreviations and punctuation in to different formats, slangs and elongated words to their special format , lemmatizing , stemming character encoding etc.

2.2.3 Feature representation

Once the feature reduction of each feature is done, it should be presented as a numeric value in order to input that value to machine learning algorithms. The numeric value assigned to the feature acts as the weight or score. There are a few popular techniques discussed below.

2.2.3.1 Binary Representation

This is one of the most basic and popular techniques out there. Here, we represent the feature in binary form (0 or 1) based on the presence or absence of the feature in a particular document. That kind of feature representation is named as a binary vector. If your features are in a bag of model form and if your sample is for example, 1000 words, you can define a binary vector v [I, j] =1 if the document i contains words j and v [i, j] =0 otherwise. Peramunetilleke and Wong [31] defined a time period where all news headlines appear if a keyword record appears at least once where the variable is set as one, otherwise zero.(Werner and Myrray [11] ; Zhai et al [28])

2.2.3.2 Information Gain (IG)

This is defined as the term goodness criterion in machine learning. It is calculating the number of bites in an information which is extracted for predict the category. In the training corpus, we compute the information gain for each unique term. If information gain is less than the threshold that was predefined earlier, that term is removed from the feature space. Computation is doing using the estimation of the conditional probability of category for given term. In Groth and Muntermann [32], this technique has been used for the management of financial risks.

2.2.3.3 Chi-square Statistics (CHI)

This method calculates the independence between category and term. For each category, they computed the distance between each u term in the corpus and particular category. The complexity of CHI is quadratic and the distance is comparable within each category since it is a normalized value. However, this normalization cannot be considered if terms are lightly populated in case of low frequency terms. If we are dealing with low frequency terms, it is not suitable to use the CHI -square method to represent the features. [27][32]

2.2.3.4 Document Frequency

If one particular word appears in many documents , sum of those documents consider as document frequency. We count all document frequencies with respect to all terms in the feature space. If the document frequency is less than a particular predefined threshold, those are removed from the feature space. Those rare words are considering as negligible when doing the category prediction due to the small impact for the performance and overall results. When we take out the rare words, it reduces the dimensionality and an improvement in the performance can be expected as well. This method was used in Groth & Muntermann,[32] , Hagenau et al [27].

2.2.3.5 Term Frequency-Inverse Document Frequency (TF-IDF)

This feature is used to calculate the importance of a word in a document or corpus. If the frequency is high proportionally the importance of the document getting high. To calculate the TF-IDF, we are using two terms. We calculate term frequency and Inverse document frequency separately and get the product of those output values.

2.3 Regular Machine Learning Approach

After preprocessing and dimensionality reduction occurs and features are converted into numerical values, we can input that data into the machine learning algorithms. Below We have summarized all the machine learning algorithm usage of past researches. Most algorithms are used for classification into the classes, which are like market movement of up, down or steady. There is also some work needed to predict the market prices. Regression modules are used for prediction. We can use 6 categories to divide all learning algorithms:

2.3.1 Support Vector Machines (SVM).

SVM is a non-probabilistic supervised learning algorithm used to separate two classes with a maximum margin. This is a very popular algorithm due to its ease of use, good performance and the ability to apply it to solve a variety of problems. It is successful in many fields like bioinformatics, text, and image recognition. This algorithm is used to learn from input data.

A common use of SVM is named SVM light. This is developing based on the SVM using the C language. SVM light was created by Joachims [33]. He mentions the training problem as a quadratic programming optimization problem. It has two constraints named bound and linear equality constraints. Even though these problems are well understood, there are issues when designing an SVM learner for this. When we are learning from large training samples, optimization techniques for general quadric programs run out of memory and time requirements. To address that issue, Joachims [33] introduced a method called SVM light that makes large-scale SVM more accurate and efficient. When developing an algorithm for this, a support vector decompose method was followed in Osuna et al [34] where a large training data set was divided in to many small datasets. Large tasks are then divided into smaller chunks. One chunk is known as a “Working set”. This decomposition suggests the memory requirement should be linear in both number of training examples and number of support vector machine. That is one advantage of using working set. The disadvantage is that that this will take more time .Joachims [33] works towards making an algorithm more effective by selecting more efficient working sets, keeping less support vectors than the training.

This SVM light method is directly used in the Mittermayer [29] and Pui Cheong Fung et al [13] review works. Mittermayer [29] introduces a software called NewsCATS (News categorization and Trading system). It recognizes the stock market trend right after the publication press release. 200 trading days were used as training set.Also 6602 press release used as test set. Then a single dictionary consisting of 1000s of terms is used. The TF * IDF calculated for each word and a m* n feature vector is prepared. After that, the SVM algorithm can be applied. SVM needs both positive and negative documents as the training set. Then, SVM searches for the best decision surface that can separate positives from negatives in the multi-dimensional space. Support vectors are closest to the margin. The advantage of SVM is again, it’s better performance. The disadvantage is that the document could be categorized into many categories. Since similarity is measured independently for each category.

LIBSVM is another implementation version of the SVM, which is used in both review works of Chen, and Lin [35] and Soni et al [36]. LIBSVM is a library that supports SVM not only in stock market prediction but also using several domains like computer vision, Natural Language processing, Neuroimaging and bioinformatics. It comes with some interesting functions such as support vector classification both two class and multi class (SVC), support vector regression (SVR), One class SVM. This is a very popular library in many domains and there are more than 250000 downloads from worldwide researches.

Also, SVM can be extended to be a non-leaner classifier using the kernel mapping, which is commonly used as a kernel trick. The idea is to transfer the classification problem into a higher dimensional space. A leaner classifier in the high dimensional space can be assumed as the nonlinear classifier in the low dimensional feature space. In the study of Zhai et al [28] Gaussian RBF (Radial basis function kernel) kernel and polynomial kernel are used . The effort was to improve the prediction of the stock market trend using news releases and technical indicators. Twelve months of data used for training and only two moths of data use for validation. This study used controlling parameter, feature width for BRF and power for polynomial kernels respectively as parameters. Even though Gaussian RBF performs slightly better than the polynomial kernel, BRF is biased to one class, and all data is classified to one class. To reduce the bias factor, they used the polynomial kernel for the SVM.

2.3.2 Regression Algorithms

There are a few major categories in regression algorithms that we are going to discuss. Support Vector Regression (SVR) is a commonly used regression algorithm, which is a regression based implementation of support vector machine. Obviously, SVR is the enhanced version of the SVM. Earlier, SVM only returned a binary measurement. However, SVR can predict even the discrete values that stocks return and can predict the squared correlation coefficient between the predicted value and actual return value. Optimization is very similar to the SVM. While SVM gives us true or false values , SVR predicted continuous values. Higher deviations like profit or losses are more meaningful to realize. Since the output is a real number, it is very difficult to predict the algorithm as it is very complex (Drucker, Burges,Kaufman, Smola, & Vapnik [37]). The basic idea is the same as SVM and all we have to do is minimize the error by individualizing the hyperplane, which maximizes the margin.

The review work of Schumacher et al [15] uses SVR. They chose the SVR Sequential Minimal Optimization function in Weka. They selected n-fold cross validation and linear kernel for the validation task. So they ended up with a module that can predict the price predictions of each financial news article encountered.

The linear regression model is also a very popular technique in regression algorithms. This is predicting an equation based on the Y value and X value. When weights are properly trained we can predict y value for given x.

If the goal is prediction, we can predict the Y value of any X. That is the brief idea of the leaner regression. Linear regression was used by Jin et al [14]. They modelled a currency forecasting system based on news topics. The explanatory variables were interest rate, inflation, stock and unanticipated events and the dependent variable was the change in currency value.

The next regression model is multivariate regression, wherein an output is calculated for the same value at multiple time points (repeated measures) or the modeling of clustered data. Past work of the Chatrath et al. [38] uses stepwise multivariate regression model in the probability model.

The OLS (Ordinary Least Square) method is also another regression model used in papers in the past. The goal here is to minimize the sum of the squares of the differences between the observed responses in the given dataset. First, take the difference between the dependent variable and its estimation. Square that difference and get the squired sum for all the data. This method is used by Tetlock et al [30] for estimate the ability of negative words to predict earnings.

2.3.3 Naïve Bayes

This is one of the oldest algorithm in the machine learning domain. However, it is still very famous and is used in many works. It is based on an assumption that considers complete independence among the text features. This algorithm created on Bayes theorem. It performs well in the field of text classification. It can be applied for linear separable function. Following is the Bayes theorem.

$$posteriorprobability = conditionalprobability * priorprobability / evidence$$

we need to create a decision rule maximizing the posterior probability in a given training dataset. Usually in text classification tasks we assume all the words in the documents are conditionally independent. There are mainly two naïve Bayes models named Multinomial model and Multi variate Bernoulli model.

Let’s looking into Multi-variate Bernoulli model. This is based on binary data. Every value in the feature vector either of value zero or one.The size of the feature vector equal to the size of the vocabulary in the document . If the value is 1, that means the word is present in the input document.

The other approach classifies text documents based on the term frequency. The number of time a word is in a document called as term frequency. Usually, term frequency is normalized by dividing the document length. Using the normalized word frequencies, we can calculate the maximum likelihood estimate based on the training dataset. This is using for estimate class conditional probabilities in the multinomial model. From the below mention equation we can compute the class conditional probabilities for each word.

$$P(x|\omega j) = P(x1\ |\omega j).P(x2\ |\omega j).....P(xn|\omega j) = \prod i = 1mP(xi|\omega j)$$

If the vocabulary size large, the multinomial model is better than multi-variate Bernoulli model. Usually, the performance depends on the feature selection and selected features. It is recommended to proceed the text classification after doing multiple experiments with different feature selection and extraction. Then, you can always select the best model suitable for the problem.

In Yu, Duan, and Cao [16], this algorithm was applied to understand the impact of social media such as twitter for the stock market performance. Li [39] used a naïve Bayes module with Perl language to conduct the prediction. He converted the vector of words for each sentences after preprocessing into a hash variable. He proceeded with that hash variable collected from manually-coded 30000 sentences and fed into the Bayesian classifier for training. The algorithm then predicted the tone and category for the test set, which consist of 13 million sentences.

2.3.4 Decision Rules and Trees

These algorithms used mostly for the rule based classification system. Peramunetilleke and Wong [31] used a set of keywords that were provided by the domain experts. News headlines were taken as the input for each consistent time period to match with existing keywords and get counts for each. With the help of the TD* IDF calculation, all frequencies were converted into weights. There is a correlation between a keyword and one of the outcomes of the ruleset. They have created three rulesets, namely UP, STEADY and DOWN. So a particular probability state shows certain events consisting of keywords falls into the three classes. This is just a bag of words module that considers single keywords. To increase the accuracy Huang, et al [40] used two or more combinations of words. As they have a relationship among the keywords, they applied the weighted association rules algorithm to detect the important compound terms in the news headlines.

A popular decision tree induction algorithm is C4.5. This does not assume the independence of an attribute. Rachlin et al [23] used this algorithm to predict the trend of the stock market. He used both textual and numerical data to present the effect of the combination of both sources. Vu et al [41] also used the C4.5 algorithm to predict the daily fluctuation of the market.

Obviously, all rules depend on the set of words, and words have meanings. So rules themselves have insights. Rules can include recognizing meaningful patterns and sequences that are helpful in categorizing the text. If the rules are complex, we have to expect less accuracy from them. Decisions trees are a special kind of decision rules that represent a tree structure and predictions are made at each leaf node.

2.3.5 Combinatory Algorithms

Machine learning algorithm are also can be used together. Das and Chen [22] have used different algorithms such as Naive Classifier, Bayesian Classifier and Bayesian classifier. They used various hybrid methods to implement a voting system to extract investors sentiments. Among all five classifiers, the best accuracy was given by the widely-used Bayesian Classifier.

Mahajan et al. [26] presented a financial news classification system which is based on text mining . Topic extraction is done through the Latent Dirichlet Allocation (LDA) mechanism to identify all topics that can have an impact on the market prices. They used a stack classifier to predict the market trend based on the topics in newspapers. This classifier is creating by combining some generic classifiers based on generalized voting procedure. The voting part consider as a separate classification problem. Stacked classifier uses the decision tree upon information gain numerical attributes coupled with a SVM with sigmoid kernel. Finally, they could achieve 60% accuracy with this hybrid system.

Butler and Kešelj [14] introduced two novel methods to analyze annual reports, which was helpful to assess the performance of the stock price during the next year. For the first model , they created character N- gram for each report and labeled based on CNG classifier. Next model is readability scores combined with performance inputs and then input to a support vector machine for classification. When they are concatenated, those two methods got better results than any method.

2.4 Lexicon – Based Approach

There are lot of past researches based on the lexicon approaches [42]. Most of the researches were using bag of words those days. The lexicons used for classifying phrases and sentences based on their positivity and negativity. They used pre-defined dictionaries for that [39]. Word weighting and lexicon / dictionary is most important in this approach [43].

There are some oldest lexica like The General Inquirer (GI) and DICITION which used for the financial sentiment analysis. The general inquirer is a built in dictionary [44]. DICTION is a textual analyze program. But these lexica were not showing better accuracy.

Therefore, many researches tend to introduce better lexica for financial sentiment analysis. A successful lexica was finally developed by Loughran and McDonald which consist of finance wordlist [45]. U.S. Security and Exchange Commission portal (1994 - 2008) was used to build this lexicon. After that many financial sentiment researches used this lexica and it was clearly enhanced the accuracy of them [43] [46]. It called Loughran-McDonald financial sentiment dictionary (LMFSD).

There are many different lexicons are using along with LMFSD nowadays. General lexica like MPQA (Multi-Perspective Question Answering) is a Subjectivity Lexicon [47]. In sake of a better domain adaptation some of the researches are even extended the LMFSD [48].

The top ranked teams in the SEMEVAL 2017 task 5 were using mostly common set of lexica like Loughran and McDonald [49], Opinion lexicon [50] and MPQA subjectivity lexicon [51].

	Lexica
L1	Loughran and McDonald (Loughran and McDonald) [49]
L2	Stock Market Lexicon [52]
L3	SentiWordNet [53]
L4	SenticNet4 [54]
L5	VADER (Hutto and Gilbert) [55]
L6	Opinion Lexicon(Hu and Liu) [50]

L7	MPQA Subjectivity Lexicon (Wilson et al) [47]
L8	NRC Hashtag Sentiment Lexicon (Kiritchenko et al) [56]
L9	NRC Hashtag Emotion Lexicon (Kiritchenko et al) [56]
L10	NRC Hashtag Affirmative Context Sentiment Lexicon (Kiritchenko et al) [56]
L11	NRC Hashtag Negated Context Sentiment Lexicon (Kiritchenko et al)[56]
L12	NRC Word-Emotion Association Lexicon / NRC Emotion Lexicon (Kiritchenko et al) [56]
L13	NRC Word-Emotion Association Lexicon /NRC Emotion Lexicon (Kiritchenko et al) [56]
L14	Sentiment140 Affirmative Context Lexicon (Kiritchenko et al) [56]
L15	Yelp Restaurant Sentiment Lexicon [57]
L16	Amazon Laptop Sentiment Lexicon [58]
L17	Macquarie Semantic Orientation Lexicon [59]

Table 2.1. Semeval 2017 task 5 lexicons table.

Rank	Research	lexica
1	Jiang et al (2017) [60]	L3 + L6 +L7 + L8
2	Ghosal et al. (2017) [61]	L1 + L6 + L7 + L8 +L10 + L14
4	Cabanski et al.(2017) [62]	L3 + L5 + L6 + L8
5	Kumar et al. (2017) [63]	L6 + L7 + L8
6	Kar et al(2017) [64]	L4
7	Nasim (2017) [65]	L1 + L2
8	Seyeditabari et al (2017) [66]	L1
9	Saleiro et al(2017) [67]	L1 + L7 + L9
13	Chen et al (2017) [68]	L3 + L4 + L10 + L11 + L12 + L13 + L14 + L15 + L16 + L17

Table 2.2. Usage of lexica by Semeval 2017 task 5 teams.

The advantage of the lexica is it provides good performance data in a high dimensionally space. This is very useful when you have the small dataset which includes less texts to classify. We can gather more text features to our feature list by using different lexica. But unfortunately most lexicon based method are coarse-grained methods and they are not sufficient to detect the polarity of classification in a fine – grained way. There for most researches not used lexicon methods lonely. Most of them used in a hybrid way along with machine learning or deep leaning techniques.

2.5 Word Embedding

Word embedding is a vector that represent word in a text document. This is a very important feature that can learn from text data. The vector depends on the context that word appears. This numerical vector is providing the facility of representation any word in the vector space. Semantically or syntactically matching words are nearby each other in the vector space[69]. This similarity is depending on the corpus that embedding are trained. In the early stages of sentiment analysis is used numerical functions to represent words [70].

There are different methods to learn word embedding from a corpus. But there are three methods that researches widely used.

2.6.1 Word2vec

This technique has been introduce by Mikolov et al [71]. In that time, he was working at google. This is the widely used word embedding method by researches. There are two ways to implement word2vec such as a continuous bag-of-words (CBOW) or as a skip-gram (SG) [72][73]. Both CBOW and SG methods use small neural network to train word embedding [74]. There are two key parameters to train word2vec. 1) the dimension of the embedding (between 50 and 500) 2) the length of the window of skip gram model. (usually 5 or 10 words) [75].

The system is trying to predict the probability of a word in a given context in the CBOW method. Sometime context may be a word or group of word. Simple neural network can be used for map words to the target variables which are also words. System will learn the weights in the learning process according to the input and output vectors. One hot encoded vector was used as input layer and output layer. Following architectures can be used for predicting the single context (Figure 2.2) word and multiple context words (Figure 2.3).

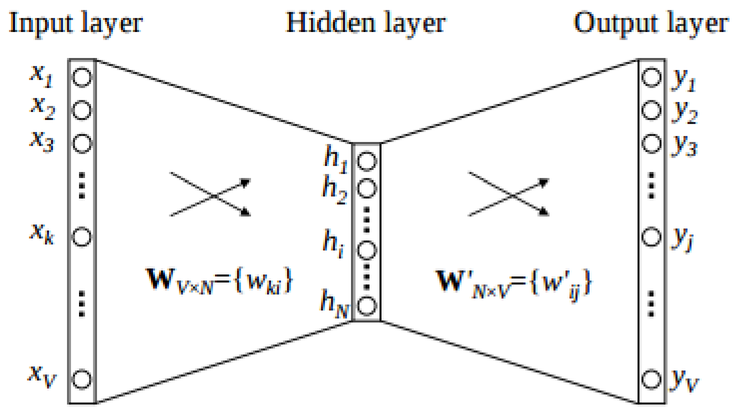


Figure 2.2. Neural network architectures for CBOW single word context [70].

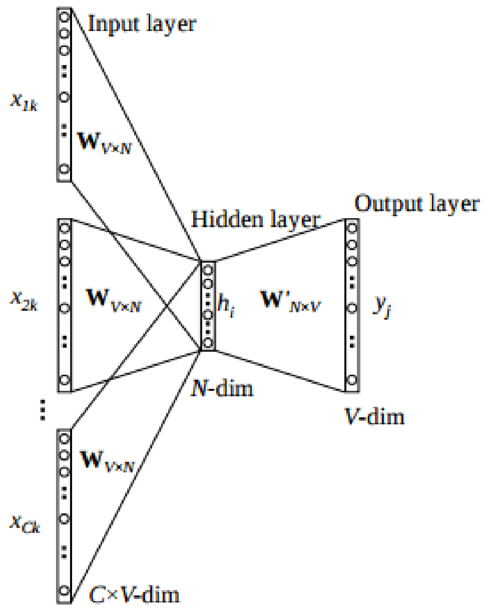


Figure 2.3. Neural network architectures for CBOW group of word context [70].

One of the main advantage of CBOW is low memory usage. Usually CBOW is taking the average of the context of the word. For an example, bank can be having two meanings depends on the context. It may be river bank or money bank. But CBOW will get the average of both contexts. That is a disadvantage of CBOW method.

Skip gram model is just an opposite of CBOW method architecture. Here objective is predicting the context given the word. The input layer is the remaining same as CBOW. But there will be two target vectors and two corresponding output vectors since context window is consider both the sides.

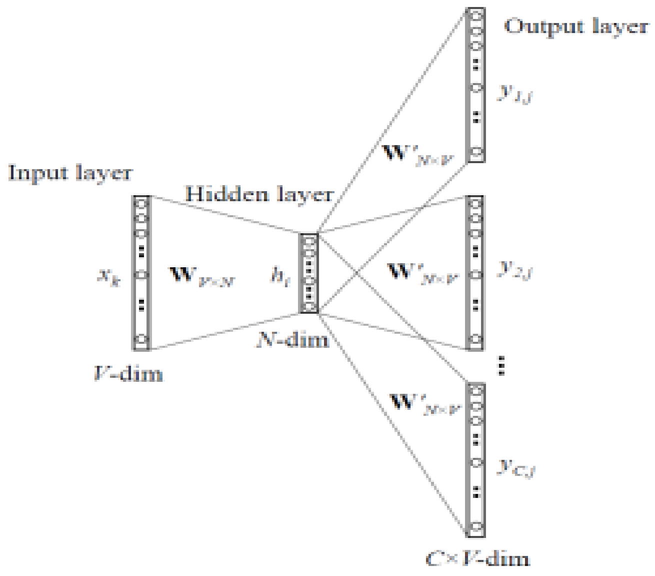


Figure 2.4. Neural network architectures for skip gram model [70]

Skip gram model can represent two semantics for single a word. For an example it can represent two vectors for both meaning of “bank” which we discussed earlier. Skip gram with negative sampling is perform well than every other methods. [76]

The winning team of the Semeval 2017 has proved the impact of the word embedding [60]. They used publicly available google word2vec [73] in two forms of features. First way of representing the sentence as vectors and then they created the feature called word clusters. First of all, they used the google word2vec pre-trained model to get 300-dimensional feature vector for each word in the sentence. Then they used simple min, max and average pooling strategies to concatenated sentence embedding feature. Finally, they used 900-dimensional vector for represent each message.

Since the effective features are always represent in law dimensional space the Jiang et al [60] introduce new feature called word clusters. After extract word vectors for each word they used the k-means algorithm (k =50) to cluster all word vectors in 300-

dimensional vector space. After that corresponding cluster was used to represent the words in each sentence. That way they created very effective 50-dimensional feature set.

Apart from that some teams in Semeval 2017 used different pre-trained word2vec models. Ghosal et al [61] who are in second place in the competition used twitter model trained on 400 million tweets which was founded by Godin et al [77]. Also two word2vec models used by Cabanski et al [62]. First model was a pre-trained model which is in spacy library called Levy and Goldberg dependency model [78]. As well as by Cabanski et al used self-trained model which was constructed based on the same dataset. Self-trained model has whole vocabulary in the dataset but it trained on smaller corpus.

By studding recent past literatures, we can understand that the word2vec feature plays a major role in the sentiment analysis domain. It helps to increase the accuracy in most of the models. Word2vec does not support for word order and global information. That is the major weakness of the word2vec. The global vector (Glove) [79] which was introduced by Stanford university later resolved this gap.

2.6.2 Global Vectors (GloVe)

The global vector model can support global statistical information. Also it creates meaningful substructure in the vector space with help of matrix factorization methods. Considering these facts it also outperform word2vec on named entity recognition and similarity tasks [79]. There are two ways of learning word embedding using glove model. Either we can use matrix factorization or shallow-window techniques.

In the matrix factorization they were created a co-occurrence matrix by consolidating all word occurrences of the word in the context. Normalization was done to overcome the problem of high frequency words.

Shallow – window technique is used the same techniques that were used by word2vec. It will be scan the context with a local context window to learn word embedding. But this method suffers with the same issue that it cannot grab the global information in the entire corpus. As well as it will fail to take advantage on a corpus which having cast amount of repetitive data.

Following figure 4 shows a comparison between glove and word2vec on word analogy task [79] . It can clearly see accuracy of glove clearly outperform the word2vec CBOW and skip-gram models.

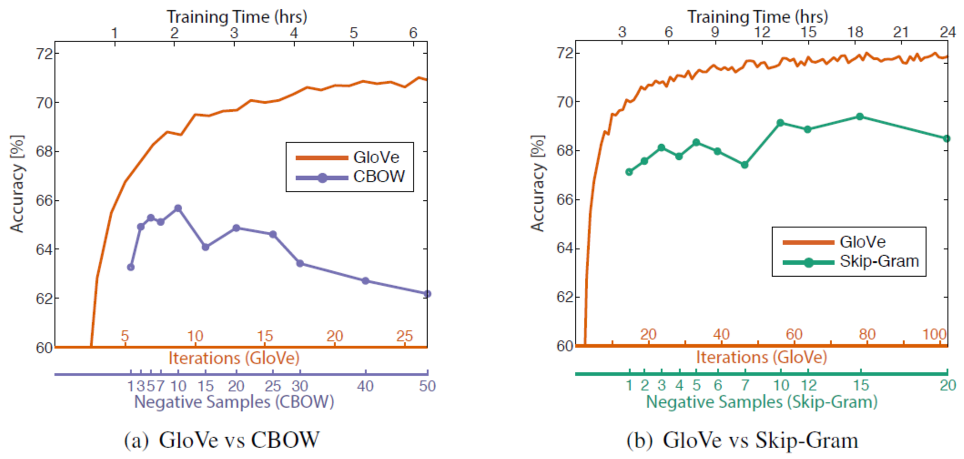


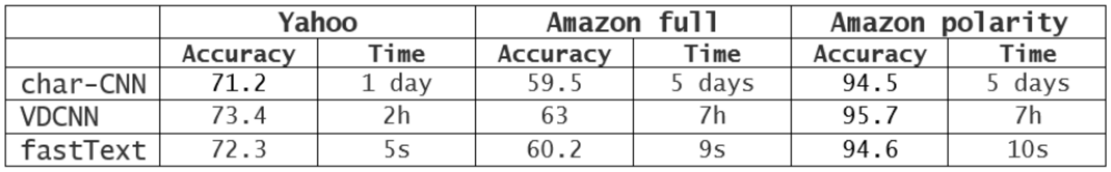
Figure 2.5. Overall accuracy on word analogy task between glove and word2vec. [73]

Glove vectors were used for sentiment analysis by researches in recent past. Even Semeval 2017 teams are tend to use as a major feature set in their models. Ghosal et al [61] , Kumar et al [63] used 2 billion pre-trained twitter model for their system.

2.6.3 FastText

FastText was introduced by facebook. This method can be used to address the limitations of word2vec and Glove methods. Specially it can handle the words which are out of the vocabulary by extending the word2vec skip gram model with internal information [80]. FastText is perform well better than word2vec and glove because it is a character level representation. Each word represents as a bag of character n-grams in addition to the word itself. Then it will create a vector based on these n-grams composition.

Fasttext is extremely light compare to the other models. Fasttext can even reduce it size to match on mobile devices. Most importantly fastText maintain the local word order when it uses vectors to represent word n-grams. Fasttext able to cut down several days of training time into few seconds on standard datasets. [81]

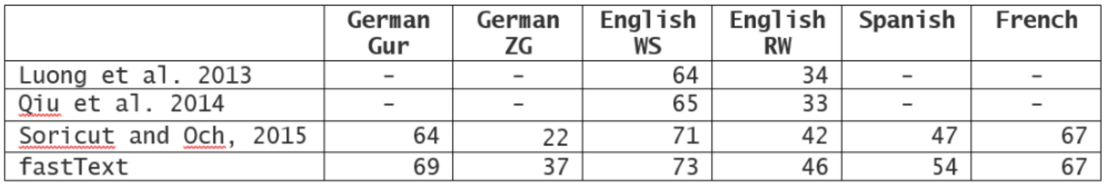


	Yahoo		Amazon full		Amazon polarity	
	Accuracy	Time	Accuracy	Time	Accuracy	Time
char-CNN	71.2	1 day	59.5	5 days	94.5	5 days
VDCNN	73.4	2h	63	7h	95.7	7h
fastText	72.3	5s	60.2	9s	94.6	10s

Figure 2.6. Comparison between other deep learning methods and fastest on tag prediction[75].

Fast text is dedicated tool for text classification which is very important in the commercial word. Even though there are generic tools like Vowpal , libSVM exists in the market for spam or clickbait filtering , FastText is outperform them by its power to train large datasets quickly. Fasttext can train more than 1 billion words within 10 minutes in a standard multicore CPU. Also it can classify a half-million sentences in 5 minutes which is spread across 300000 categories [81].

Apart from text classification, fastText work is design to work with various languages like Czech , French , Spanish and German. FastText can achieve significant better results over word2vec and other state of the art word representations.



	German Gur	German ZG	English WS	English RW	Spanish	French
Luong et al. 2013	-	-	64	34	-	-
Qiu et al. 2014	-	-	65	33	-	-
Soricut and Och, 2015	64	22	71	42	47	67
fastText	69	37	73	46	54	67

Figure 2.7. Comparison between other word representation methods and fastest on different languages [75].

2.6 Deep Learning Approach

Deep learning algorithms have already made huge impact on domain like pattern recognition and image processing. Following this path recent research work on NLP now more focusing on deep learning techniques. In the beginning of sentiment analysis researches vastly used shallow machine learning models like SVM, random forest and logistic regressions. The main disadvantage of these models are those algorithms trained on very high dimensional and sparse features. Specially in the last two years, deep learning method beat most of the machine learning baselines with help of word embedding and various neural network structures [82]. Deep learning can be defined as a “deep” neural network and it includes group of multilayers of nonlinear processing units for feature extraction. Deep learning models depends on multilevel automatic feature learning. While lower layers of deep neural network are learning simple features, higher layers learn complex features. With comparing traditional machine learning approach that they highly depend on handcrafted feature sets which are most likely incomplete and time consuming to create [83].

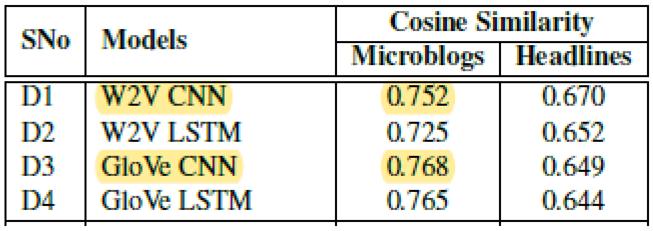
We reviewed most important deep learning models applied to sentiment analysis tasks such as convolutional neural network s (CNNs), recurrent neural networks (RNNs) and long short term memory (LSTM)

2.7.1 Convolutional Neural Network (CNN)

Convolutional neural networks originally applied for computer vision. CNNs models have been shown exceptional results in information retrieval, semantic parsing sentence modeling and other various NLP tasks.

Kim [84] used CNN for sentence level sentiment classification. He used datasets like MR, SST-1 , SST-2 ,MPQA etc . He presented series experiments with CNN on top of google word2vec. With compare to other models he used, one layer of convolutional performs significantly well. Document level sentiment classification remains a challenge for longtime. Tang et al [85] introduced a neural network model Conv- GRNN for document level classification which is deriving from traditional CNN. This approach encodes the semantics of the sentence into document level. Other researchers also applied CNN models on character level [86] and word level classification [87] tasks. CNN model has been outstanding on benchmark datasets with respect to the other baseline models.

As well as the runners up paper by Ghosal et al [61] in Semeval 2017 also used a CNN model and it was outperform LSTM model.



SNo	Models	Cosine Similarity	
		Microblogs	Headlines
D1	W2V CNN	0.752	0.670
D2	W2V LSTM	0.725	0.652
D3	GloVe CNN	0.768	0.649
D4	GloVe LSTM	0.765	0.644

Figure 2.8. Ghosal et al [56] cosine similarity comparison between CNN and LSTM on validation set.

2.7.2 Recurrent Neural Networks (RNN)

RNNs are making connections as directed cycles. This is very important model when we are processing sequential data like texts. Most of the time in text the output is depends on the previous computations or results. Usually fix sixed vector is created to feed the RNN and it use the memory of the previous computation to process current computation. This process can be applied many of NLP task such as speech recognition, machine translation and language modeling etc. RNNs tremendously popular in recent past in NLP domain. [82]

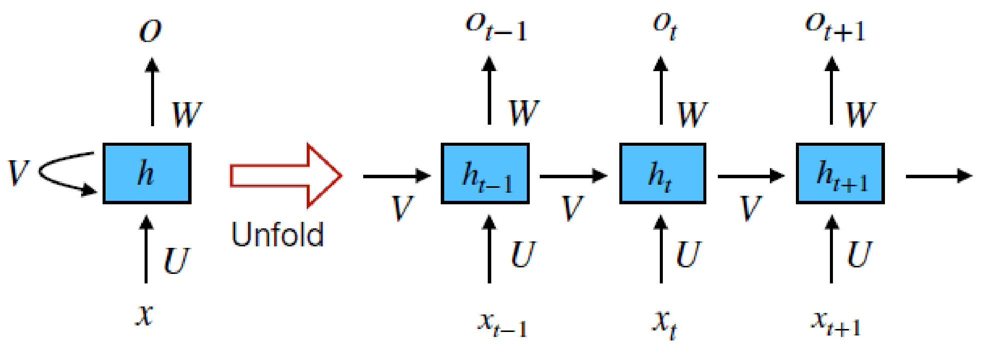


Figure 2.9. simple RNN model. [76]

RNNs has the capability to process inherent information from previous sequences where units are in characters, words even sentences. RNN can learn semantical meaning of a word based on the previous word. For a instance it can be clearly recognized the difference between “dog” and “hot dog”. RNNs are very useful to handle such similar sequence modeling tasks and context dependencies. These are the strong reasons for researchers to choose   RNNs over CNNs in classification tasks.

2.7.3 Long Short Term Memory (LSTM)

LSTM solves the vanishing exploiting gradient problems and save more information than RNNs. The main different between RNN and LSTM is the gated cell. Capability of saving additional information is given to the LSTM by these gated cells. All information can be read from cell or write to the cell. Process of removing or storing data decided by gates opening and closing. Every cell consists of four components named: input gate, forget gate, output gate and a neuron with a self-recurrent connection. LSTM cell can forget its previous state using forget gate.

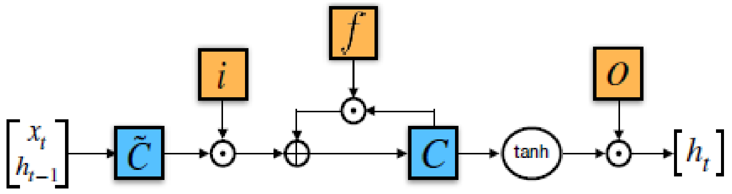


Figure 2.10. Long Short-Term Memory [76].

Cabanski et al [62] used two models on individual features in semEval 2017. By analyzing the results, we can see the LSTM model clearly outperform the SVM model.

2.7.4 Gated Recurrent Units (GRU)

As discussed earlier compare to LSTM. Main difference is GRU has only two gates while LSTM having RNNs are suffer from short term memory. If sequence are too long RNNs are fail to carry out information from earlier time steps. To solve this issue Chung et al [88] introduce another model called gated recurrent units. This is very similar to famous LSTM model. But GRU has slight differences three (figure 2.11). GRU cell has only reset gate and update gate. GRU model perform better in smaller dataset over LSTM. Also GRUs train faster compare to LSTM. The update gate in GRU more similar to forget gate and input gate in LSTM. It determines whether which data should be retained or throw away. Reset gate decide how much past data should be forget. GRUs are not perfect solution for longer sequences. But very useful in short text formats like microblogs.

Also in Akhtar et al [89] has been used a GRU model as its one of four models.

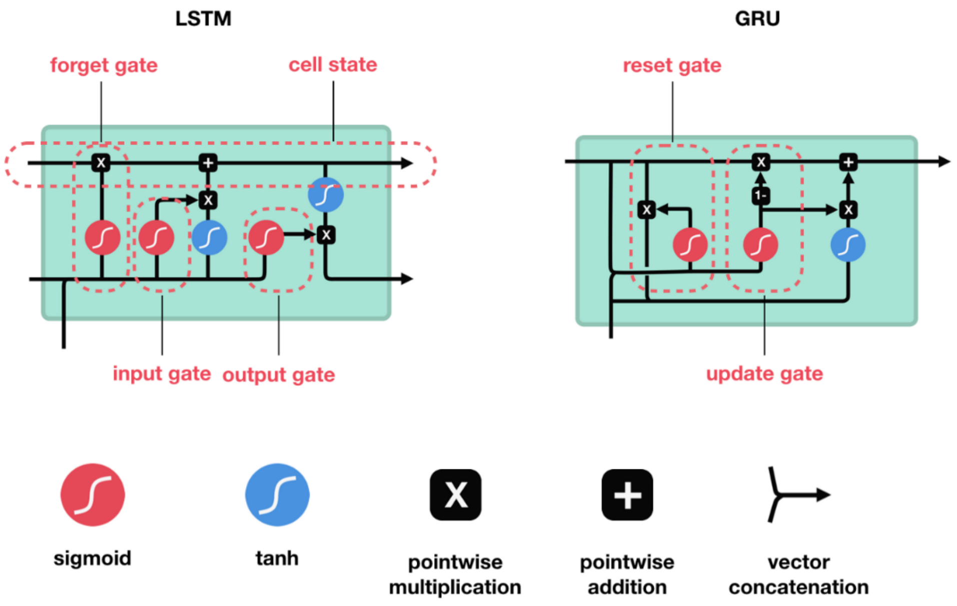


Figure 2.11. LSTM cell vs GRU cell [90]

2.7 Ensemble Approach

Most of the recent past researchers used ensemble models to improve results among their models. They concatenate multiple models and gain improved overall performance. Ensembling usually reduce the overfitting by reducing the generalization error. Even in SemEval 2017 competition, more teams tend to use ensemble method to beat other models. Jiang et al [60] which was first in the Semeval 2017 task 5 also used many ensemble algorithms. There are four main techniques for ensemble models named: stacking , Blending , bagging and boosting. As well as algorithm like random forest can be categorized as bagging ensemble approaches. Algorithms like Adaboost , GBM , XGBM can be categorized as boosting algorithms.

Ghosal et al [61] achieved best results by ensembling their all four systems (CNN ,LSTM , vector averaging , Feature driven) to create new feature vector and then input them to a multilayer perceptron (MLP) network for training. Figure 2.12 shows the proposed system.

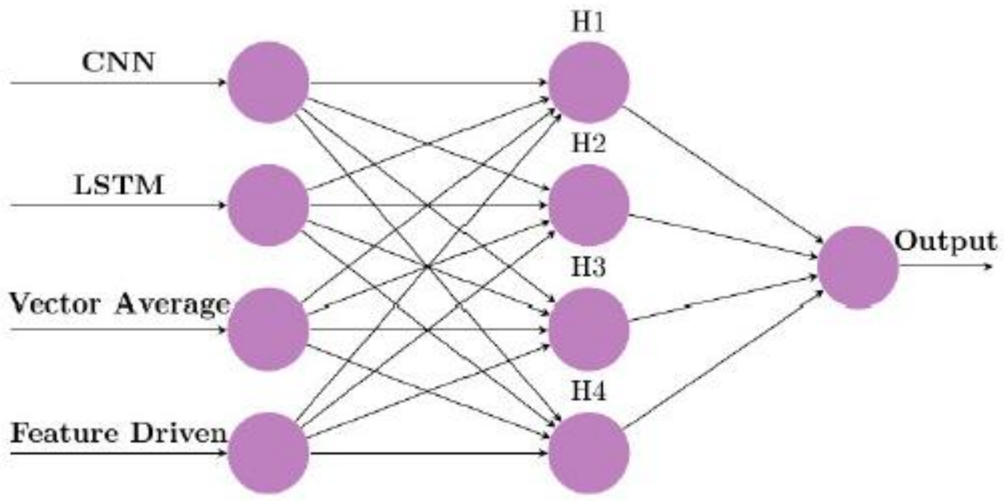


Figure 2.12. Ensemble MLP model [56].

Further working on the same architecture which was introduce by Ghosal et al [57], Akhtar et al [89] proposed an ensemble model which is better than the existing state of the art systems for the SemEval-2017 task 5 microblog dataset. They created four individual systems mainly and ensemble those models using a MLP based ensemble architecture. (Figure 2.13)

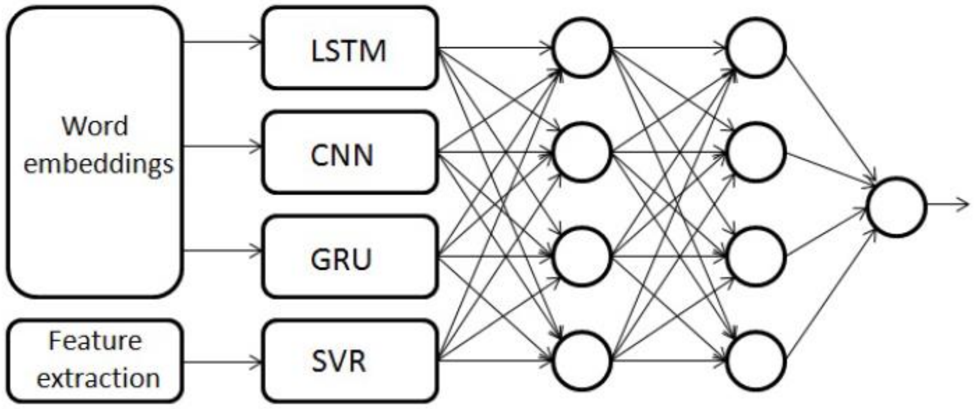


Figure 2.13. MLP based ensemble architecture [82]

As network parameters for LSTM, CNN and GRU models, they used fully connected layers which having 50 neurons and occupy two hidden layers. Also used Relu activation for immediate layers and tanh activation for in the final layer. Akhtar et al [89] introduced novel ensemble method (figure 2.13). They use MLP consist of two layers that each layer has four neurons. they were applied 25% dropout in the immediate layers and use Adam optimizer during backpropagation. Final prediction value has been given as the output of the model.

Chapter 3 : Methodology

3.1 Data

The dataset we have used for this research taken from the Semeval 2017 Task 5 competition [91] . Mainly dataset contains microblog messages with their manually annotated sentiment values. Microblog messages were collected from two social media platforms. StockTwist social media platform is widely used by investors and traders for give their prospective about the stocks. Each message in the stocktwist has a reference to the company which is called as cashtags. (cashtags start with "$" symbol e.g. $AAPL cashtag referring Apple Inc.) Apart from that message contain with short supporting text, url links and images. Images containing stock value analysis graphs. They also used twitter platform to improve the diversity of the dataset. Twitter messages are also containing stock related tweets. Those tweets have been added to the dataset.

There are 1693 messages in the training set and 793 messages in the test set. Those messages are selected over a large pool of messages after going through an initial filtering and random sampling process. While random sampling ensures a good unbiased set of message, filtering process eliminated the spam messages. Then again resampling for different time units to get most random and balance dataset which is represent entire time span. Stocktwist data refer to the period of October 2011 to june 2015.Twitter data were sampled between March 11th and 18th 2016.Both datasets were streaming using their own official APIs.

Final sample of the dataset was annotated by for independent financial experts. Each instance of the dataset contains following information.

Cashtags: Stock company symbol for a company.

Sentiment score: A numeric sentiment value between -1(very negative) and 1 (very positive)

Span: Part of extracted text from the message which sentiment is expressed

Message: Text which is expressed by sentiment.

Source: Twitter or stocktwist where the message extracted from.

Train set	1693
Positive messages in training set	1086
Negative messages in training set	581
Test set	793
Positive messages in test set	523
Negative messages in test set	257
Sentiment score (mean)	0.077250
Sentiment score (standard deviation)	0.318269
Sentiment score (minimum)	-0.866000
Sentiment score (maximum)	1.0

Table 3.1. Dataset Statistics

3.2 Data Preprocessing

It is very important to have a high quality input for increasing the overall quality of the system. The raw messages do not provide useful properties to the system. So we should go through several steps to filter out most important properties of messages. Even the small prepressing step may be lead for massive progress in the final score. Most of the preprocessing steps are adopted from Jiang et al [60] and Cabanski et al [62].

Firstly, replaced all the URLs by "url" and transform the abbreviations, punctuations with a special format, slangs, and elongated words to their normal format. NLTK python library is used for tokenization, POS tagging, named entity recognition and parsing. Potter stemming algorithm which is implemented in NLTK library is used to remove affixes lemmatized words. WordNet based lemmatizer in NLTK library is used for converting lemmatize words to their base forms. All the characters are converted into lowercase and white spaces between words are normalized to length one. Hashtags which followed by "#" , replaced by their associate characters of strings. For example, #investing replaced by "investing". Also some messages contain several spans. All such spans concatenated to one unified string. For empty spans we considered as the whole microblog message text for feature extraction. All words are converted into UTF8 encoding and standard English stopwords are removed from messages. All the cashtags and company names strings are replaced by the "company" and doller signs and euro signs followed numbers replaced by “cash_amount”.All other numbers percentages are replaced by “positive_number” , “negative_number” , “positive_percentage” and “negative_percentage.”

3.3 Feature Engineering

It is important to have a good feature vector which can impact to the accuracy of the system. All the features that we extracted can be divided into three categories. Sentiment lexicon features, word embedding features and linguistic features are those main subtypes of the feature vector. There are different techniques to follow in the process of feature extraction.

3.3.1 Sentiment lexicon features.

Since our dataset is too small it was decided to gain more additional information from publicly available lexica. These lexicons are mostly pre-defined dictionaries such as in positive or negative categories.

3.3.1.1 SentiWordNet

Using wordnet’s synset corpus which introduce by Fellbaum [92] , Baccianella et al [93] created sentiWordNet lexica. For all synset terms has a positive and negative score between 0 and 1. Also we calculated objectivity of a word by subtracting one from sum of Positive score and Negative score. Most of the time one word has a different sentiment scores. In that case we calculated the mean of all possible sentiment scores in the sentiWordNet.

3.3.1.2 Opinion Lexicon

First paper was published by Hu and Liu [94] in 2004. Importance of this lexica is it is continuously updating since 2004. Lexica comes as two files which for positive word list and negative word list. In the source code we concatenated this two files into one list and classified binary as 1 (positive) and -1 (negative). This lexicon was trained on social media data.

3.3.1.3 MaxDiff twitter sentiment Lexicon

This corpus is mainly trained on twitter by kiritchenko et al [95]. Corpus represents set of unigrams with associative strength towards positive sentiment. Each entry of the lexicon there are words with relevant score ranging between -1 and +1.

3.3.1.4 VADER

This is a gold standard sentiment lexicon that was introduced by Hutto and Gilbert [55] which is specially attuned to sentiment expressed in social media. Hence words and symbols that are not part of other traditional sentiment lexicon resources are included and scored with continuous values.

3.3.1.5 Financial sentiment lexicon

Word polarity is highly depending on the domain. Hence we created lexicon based on training dataset and do the sentiment scoring of words according to the sentiment score provided in the train set. All words in the text is assigned its sentiment score. SO we created whole vocabulary of train data with corresponding their sentiment values. Finally, we grouped by similar words and averaged their scores.

3.3.1.6 Loughran and McDonald Sentiment Word Lists

Using other general lexica on financial sentiment analysis is misclassify the certain words in financial domain. So Loughran et al [49] developed a list of positive and negative words which are using in financial domain. For created this feature , positive words and negative words are count for each microblog message.

3.3.1.7 Stock Market Lexicon

This lexica was created based on labeled stocktwist messages by Oliveira et al [96]. This is a large word list with corresponding part of speech (POS) tag and sentiment score. In order to compute the sentiment of positivity in a sentence we get the sum of the total positive values and negative values for each word.

3.3.2 Linguistic Features

3.3.2.1 N-grams

First of all, spans are needs to be cleared until remaining words of the sentence. Then we extracted three types of bag of words features where N = {1 , 2,3} (unigram , bigram, trigram). We used the "CountVectorizer" class in sklearn library to extract this feature. The fit_transform() method in that class returns a binary array as a feature for presents of n-grams in each message.

3.3.2.2 RF N-gram

This is a different approach from N-gram. In the N-gram feature we assigned the same weight for each word. Here we calculate the word weights according to the method of Lan et al [97]. First of all, we calculate the occurrence of the words in each positive and negative messages of the training data. Then calculate the 'rf' weight for each word in unigram, bigram, and trigram. Rf = max {ln(2 + (a/max(1,c) , ln (2+ c/max(1,a)}. Where 'a' is the total occurrence of the token in positive messages and 'c' is the number of sentences in the negative category that contain this word.

3.3.2.3 Word cluster

Due to high dimensionality of the N -gram feature we used word cluster to reduce the dimensions of the feature vector. Firstly we used the Google word2vec that trained 100 billion of words in google news [71], [72] to extracted the word embedding of each word in the vocabulary. Then all the word embedding vectors were clustered using k – means algorithm (k = 50). After that all the words in the messages replace by its cluster value and created a binary matrix around it.

3.3.2.4 Verb

We used NLTK POS tags to extract all part of speech tags in the vocabulary. Select all words that tags part of VB , VBD , VBG ,VBN , VBP and VBA categories. Then filter column vectors and created a binary vector to represent as bag of words.

3.3.3.5 Named Entity features

Considering that the numbers, money , percentages information are helpful for the predicting score we have gathered name entities that provided by spaCy library named entity recognition feature. There are PERSON , NORP , FAC , ORG , GPE , LOC , PRODUCT , EVENT , WORK_OF_ART , LAW , LANGUAGE , DATE , TIME , PERCENT , MONEY , QUANTITY , ORDINAL , CARDINAL built in name entities in the spacy library. We extracted these features from the words in messages represent as the binary array. [98]

3.3.3.6 Term Frequency - Inverse Document Frequency (TF-IDF)

TF-IDF feature determines how important is the word is for document. If a word occurring less frequency in the corpus TF-IDF assigns higher weights. With the help of this feature we can reduce the importance of common words. Feature vector for TF-IDF is created by the sklearn python library class called TfidfVectorizer.

3.3.3 Word embedding features

3.3.3.1 Google Word2vec

Google word2vec is one of the popular feature set for researches. It is performing well in the text classification domain. Here we used pre- trained google word2vec to extract word embedding for each words in the vocabulary we applied min , max and average pooling techniques when aggregating done towards messages from word level. Each word vector that extracted from google word2vec has 300-dimensional vector space. So the concatenated feature became 900- dimensional vector.

3.3.3.2 SpaCy Word Embedding

Python library call spaCy has provided us a pre-train Levy and Goldberg dependency-based model [78] to extract word embedding from words. We have used that for take feature vector of word embedding. It was 96-dimentioanl vector.

3.3.3.3 Genism Self Trained word2vec

Regardless of the size of our dataset we tried to extract self-train word embedding feature. The Word2vec class of python library call genism, facilitates to create word vectors. We have input our vocabulary to it and configure to get 300 – dimension vector as its output.

3.3.3.4 Glove

We have used glove pre-trained twitter model as a feature set [79]. First of all extract 200-dimentional word vector for each word in the sentence. Then concatenate using min, max, average pooing methods to represent sentence embedding. Final feature vector for each microblog message is 600-dimentional.

3.3.3.5 FastText

We used fast text pre-trained model called English word vectors [99]. Genism python library is used to load fastText model. As similar to the word embedding features we extracted min , max ,average sentence embedding for each message. Also we created self-trained word vectors using the dataset. Advantage of the self-trained model is it consist all the words in the dataset.

3.3.3.6 Genism Doc2vec

Again genism library facilitate to learn paragraph embeddings via the distributed memory and bag of words models from Mikolov et al [72]. These are self-trained vectors from the dataset.

3.3.4 Domain – Specific Features

Particularly the numbers are very important to determine whether it is “bullish” or “bearish” in the financial sentiment analysis. As well as put and call words often use in financial microblog domain and it related to sentiment score in the message [60].

3.3.4.1 Number

We have extracted 14 binary features to determine whether there are the different types of numbers in the messages like Positives , Negatives , percentages , currency values etc.

3.3.4.2 Number + keyword

As mentioned above “puts” and “call” keyword significant in the financial domain. We extracted 4-dimensional binary feature to represent “-num%” and “+num%” with “put” and “call” keyword.

3.3.4.3 Punctuation (Punc)

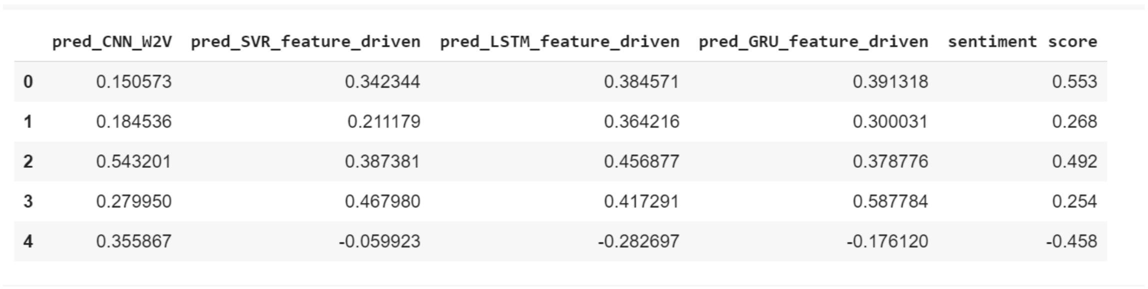
Exclamation marks (“!”) or Question marks(“?”) are often used in microblogs to show the excitement . Hence we have extracted following some of the binary features which is including those symbols.

3.4 Evaluation Measure

We have evaluated the models using cosine similarity score which was the same measure used by Akhtar et al [89] and the teams that participate in the SEMEVAL 2017 task 5. When our models predicted the sentiment scores for microblog messages of test set, calculated the cosine similarity predicted sentiment score and actual sentiment score. Cosine similarity score is calculating between those two vectors.

Cosine similarity score (G , P) $= \frac{\sum_{i=1}^{n} G_i * P_i}{\sqrt{\sum_{i=1}^{n} G_i^2} * \sqrt{\sum_{i=1}^{n} P_i^2}}$

In above equation G is the vector of gold standard scores and P is the scores that is predicted by the model. In the following table 3.2 shows the chunk of results we get for the test set comparing gold scores in the test set.



	pred_CNN_W2V	pred_SVR_feature_driven	pred_LSTM_feature_driven	pred_GRU_feature_driven	sentiment score
0	0.150573	0.342344	0.384571	0.391318	0.553
1	0.184536	0.211179	0.364216	0.300031	0.268
2	0.543201	0.387381	0.456877	0.378776	0.492
3	0.279950	0.467980	0.417291	0.587784	0.254
4	0.355867	-0.059923	-0.282697	-0.176120	-0.458

Table 3.2.Prediction scores of each model with sentiment scores of test set

3.5 Proposed Technique

We followed similar architecture which was proposed by Akhtar et al [89] with additional MLP model. We applied more advance features to improve results from baseline. We can divide our system into four major models and ensemble model.

3.5.1 Convolutional Neural Network model(CNN)

In this system we used only word embeddings to feed the convolutional neural network. This is fully connected layer which is having two hidden layers. First hidden layer has 50 neurons and second layer has 10 neurons. As well as we used Relu activation function in the intermediate layer and tanh activation in the final layer. We employed 20 % dropout for both fully connected layers. Also used Adam optimizer in tensorflow as the optimizer. As mentioned earlier we used only word2vec and glove word vectors as feature set.

3.5.2 Recurrent Neural Network (BI -LSTM)

We used "Keras" deep learning library to create our RNN models. Most importantly we focused on building special kind of RNNs called Bidirectional LSTM. The specialty of this method is output can get information from both directions. Passing information both backward and forward directions will be increased the accuracy of the output. Define two Bidirectional LSTM layers and there are 16 neurons for each layer. Also each layer employed 0.4 dropout. After that on top of BI-LSTM layers, dense layer is connected to the neural network which is consisted of 8 neurons.

3.5.3 Gated recurrent Units (GRU)

As discussed in CHAPTER 2, GRU is perform better on small datasets. Hence we decided to create a GRU model to experiment with the dataset. We have created a neural network with two hidden layers using LSTM and GRU layers. Each layer has consisted of 16 neurons each.

3.5.4 Multilayer Perceptron (MLP)

Simple MLP architecture has been created with two hidden dense layers which are having 50 and 10 neurons respectively.

3.5.5 Regular machine Learning

Apart from the deep learning models we used machine leaning algorithms also to compare the results between both architectures. Mainly we used support vector regression algorithm since it has proved the best results gainer in the past. Both handcraft features and word embedding features are fed into model. Not only SVR we have used few other ensemble machine learning algorithms just to compare the performance of SVR. Some of the machine learning ensemble techniques are AdaBoost Regressor (ABR) , Bagging Regressor (BR) , XGBoost Regressor (XGB) and Random Forest (RF) .But based on the experiments we could clearly see the SVR is outperform any other algorithm that we used.

3.5.6 Ensemble Model

The importance of the ensemble models already been discussed in our literature review. We have used mainly three types of ensemble models.

3.5.6.1 Average Ensemble Model

This is the most basic ensemble model. What we did was sum up each predictions scores for test set and divided by four since we used four models. Then calculate the cosine similarity score among averaged prediction vector and score vector of test set.

3.5.6.2 Linear Regression Ensemble Model

For this we have used simple linear regression algorithm which is trained only train data. All four models are trained on 70% of the train data and then predict 30% data using the same models. Linear regression is trained on top of 30% of predictions. Then linear regression and all models outputs are used to predict test dataset. Then calculate cosine similarity between test gold scores and predictions of linear regression.

3.5.6.3 MultiLayer Perceptron (MLP) Based Ensemble Model

Akhtar et al [89] proposed a new ensemble technique based on MLP which is learned on top of the predictions of above ML and DL models. MLP model has two hidden layers which is having 4 neurons in each and an output layer. We employ 15% dropout in the immediate layer and use Adam optimizer during backpropagation. We used three activation functions in hidden layers which are called “Relu” ,”tanh” and “elu”. Also used “tanh” function for final output layer.

Following figure 3.1 shows that present in this work. Figure 3.2 shows the model presented by Akhtar et al [89]. Main difference in the two system is Akhatar et al used only the word embedding features for their deep learning models. But we tried the combination of handcrafts features and word vectors which lead into the significant improvement of the results. Also Akhtar et al used SVR model along with LSTM , GRU and CNN models. But in the proposed method here is used all deep learning models which are CNN , MLP , LSTM and GRU. Please see Figure 3.1 and Figure 3.2 for the model architecture.

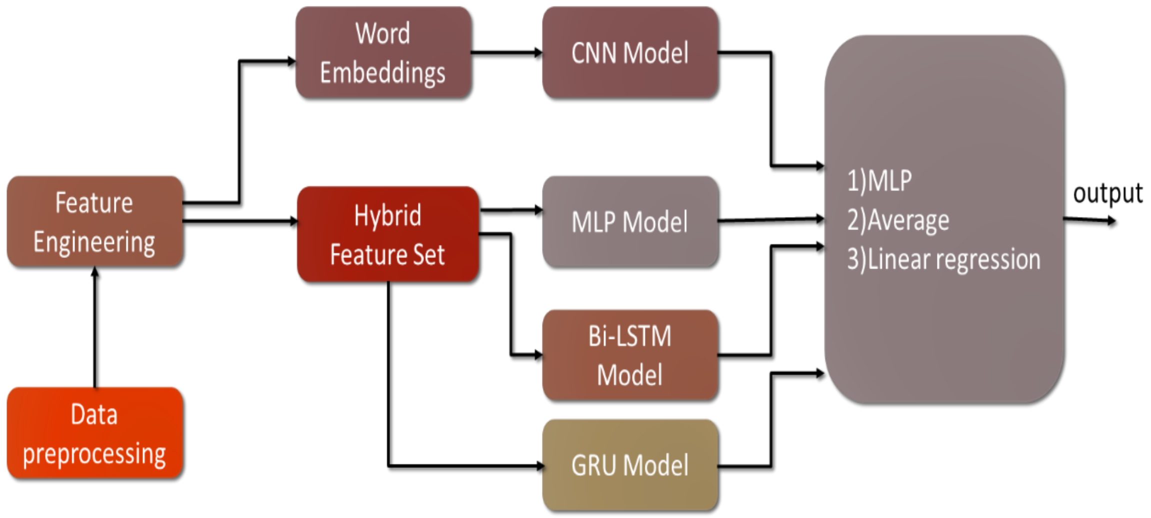


Figure 3.1. Proposed ensemble model

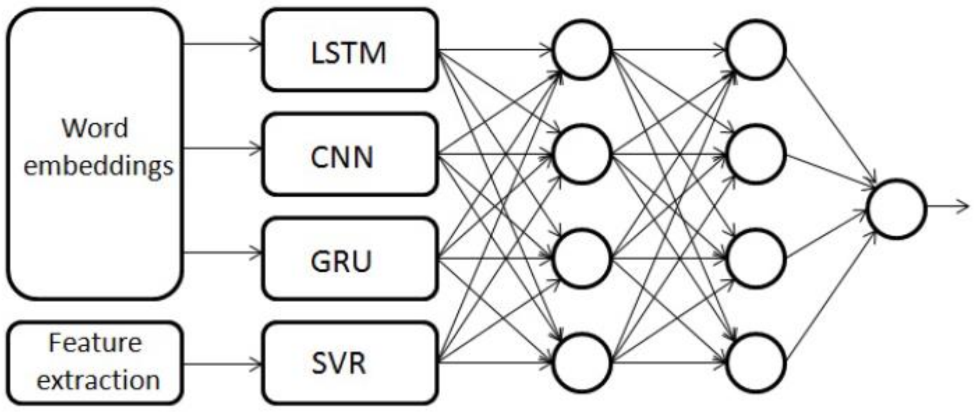


Figure 3.2 Proposed method by Akhtar et al.

Chapter 4 : Experiments

This chapter describe various experimentation we have done with the Semeval dataset which describe in the chapter 3. We have used different machine learning techniques, deep learning techniques and ensemble models to gain the cosine score with respect to the test data set. Also those all models will experiment against the features sets mention in chapter 3. For evaluating the experiments, we used cosine similarity on 5-fold cross validation. Also final experiments were done on test set. The obtain results during experiments are discussed in this chapter.

4.1 Regular Machine Learning Algorithms

We explore 5 algorithms on the dataset. Some are ensemble algorithms like AdaBoost , XGBoost. All these algorithms are used with default parameters.

4.1.1 Support Vector Regression (SVR)

Firstly, we tested each feature individually against the SVR algorithm and check what is the best performing features that give the best cosine score. These results are compared with validation set in the cross validation task.

Features	Cosine similarity
w2v spacy	0.5957
Lexicon	**0.7327**
w2v gensim	0.6835
tfidf	0.3629
Verbs	0.3446
Name entity	0.3705
Word cluster	0.5408
Google Word2vec	0.5390
Doc2vec genism	0.6637
Unigram	0.3324

Bigram	0.2938
Trigram	0.2913
Rf unigram	0.4621
Rf Bigram	0.3073
Rf Trigram	0.3016

Table 4.1.Cross validation score on individual feature sets

From Table 4.1 we found that our best performing feature was the lexical feature. It may be because we used 7 different dictionaries and large set of external data bring to the dataset from that. Second best performing feature set was doc2vec that we trained using the same messages in training dataset. Next best results are given by word2vec which also from training dataset. “Rf_unigram” feature is performing well better than other N- grams features.

Since we used different lexica, we did experiments on different lexica to recognized the best lexica feature.

Lexica models	Cosine similarity
Loughran and McDonald	0.4645
SentiWordNet	0.4477
Opinion Lexicon	0.4477
MaxDiff	0.4518
Vadar	0.5200
Financial sentiment lexicon	0.6379
Stock Market Lexicon	**0.6636**

Table 4.2. Cross validation score on individual lexicon features

According to the Table 4.2 We can clearly see the stock market lexicon feature is outperform other lexica. Because it was trained using the stocktwist financial microblogs. Financial sentiment lexical features are also performed well because it is created based on financial texts.

	Feature set	Cosine similarity
S1	Lexicon	0.7303
S2	Lexicon + Word cluster	0.7519
S3	Lexicon + Word cluster + rf-ngram	0.7575
S4	Lexicon + Word cluster + rf-ngram + tf- idf	0.7558
S5	**Lexicon + Word cluster + rf-ngram + verb**	**0.7587**
S6	Lexicon + Word cluster + rf-ngram + verb + number	0.7564
S7	Lexicon + Word cluster + rf-ngram + verb + number + keyword_number	0.7583
S8	Lexicon + Word cluster + rf-ngram + verb + number + keyword_number + punctuation	0.7578

Table 4.3. Cosine similarity score on different features with SVR on test set.

According to the above results (Table 4.3) with handcraft features we can see the highest cosine similarity score is given by S5 set. As per the results most performing handcraft features are lexicon, Word cluster , rf-ngram and verb.
Once we found the best feature set, it was tested with each word vector.

	Feature set	Cosine similarity
W1	Lexicon + Word cluster + rf-ngram + verb + glove	**0.7491**
W2	Lexicon + Word cluster + rf-ngram + verb + google word2vec	0.7331
W3	Lexicon + Word cluster + rf-ngram + verb + fastText	0.7185
W4	Lexicon + Word cluster + rf-ngram + verb + fastText self train	0.7236
W5	Lexicon + Word cluster + rf-ngram + verb + w2v_spacy	0.7432
W6	Lexicon + Word cluster + rf-ngram + verb + w2v_genism	0.7287
W7	Lexicon + Word cluster + rf-ngram + verb + d2v_gensim	0.7291

Table 4.4.Cosine similarity score on different word vectors with SVR on test set.

According to the above table (Table 4.4) it shows the glove [79] vectors are more effective than other embeddings. Reason for that we used the twitter pre-trained model for glove. It was trained on shorts text similar to our dataset. Second best results was showed from spacy word2vec which is also derived from pre-trained Levy and Goldberg [78]. But we combine word vectors with handcraft features, it will reduce the accuracy. Reason for that is the higher dimensionality.

4.1.2 Other Machine Learning Algorithms

Here we are going to experiment with few other machine learning algorithms which is having good track records with sentiment analysis tasks. Our first algorithm is ensemble AdaBoost algorithm. Also it is based on adaptive boosting technique. This algorithm is the first boosting algorithm that founded by Freund and Schapire in 1996 [100] . As well as we consider to test with XGBoost algorithm which is shorts for "Extreme Gradient Boosting". This is an implementation of gradient boosting machines. This is fast computing algorithm which is highly support for parallelization.

Also We used a decision tree as the base estimator for bagging regression algorithm. It trains the base model on random subset of training data and aggregate the predictions.

We used the same feature combinations to experiment with other models. Hence it was eased the comparison.

	SVR score as baseline	AdaBoost	XGBoost	Bagging Regressor	Random Forest
S1	**0.7303**	0.6639	0.5315	0.7225	0.6410
S2	**0.7519**	0.6833	0.5422	0.7205	0.6410
S3	**0.7575**	0.6833	0.5375	0.7399	0.6411
S4	**0.7558**	0.6893	0.5400	0.7294	0.6411
S5	**0.7587**	0.6838	0.5344	0.7381	0.6411
S6	**0.7564**	0.6793	0.5356	0.7376	0.6411
S7	**0.7583**	0.6838	0.5344	0.7381	0.6411
S8	**0.7578**	0.6838	0.5344	0.7381	0.6411

Table 4.5. Cosine similarity score on different features with other machine learning algorithms.

We can see the SVR algorithm outperform other algorithms clearly in above Table 4.5. Past researches are also prove this factor.

4.2 Long Short Term Memory Network (Bi-LSTM)

As described in the chapter 2, LSTM models are performing better than traditional recurrent neural network. LSTMs are powerful sequential models. In this case we used special flavor of LSTM called Bidirectional LSTM .We have used “Keras” library to create neural network architecture. We have created 3 different architectures and experiment with glove and word2vec word embeddings. Following table 4.6 shows the results. As per the following results in Table 4.6 , we can see the smaller neural architecture performing better than others.

	Feature set	Glove and Handcraft Features	Word2vec and Handcraft Features
L1	Bi – LSTM layer (64 neurons) Bi – LSTM layer (64 neurons) Dense layer (32 neurons) Output layer	0.7572	0.7628
L2	Bi – LSTM layer (32 neurons) Bi – LSTM layer (32 neurons) Dense layer (16 neurons) Output layer	0.7605	0.7614
L3	Bi – LSTM layer (16 neurons) Bi – LSTM layer (16 neurons) Dense layer (8 neurons) Output layer	0.7682	**0.7766**

Table 4.6. Cosine similarity score on different features with Bi-LSTM

4.3 Gated Recurrent Units (GRU)

Same experiments carried out with GRU and LSTM combined architecture. Results shows in following table. GRU model shows the best results with google word2vec feature on smaller neural network with less neurons.

	Feature set	Glove and Handcraft Features	Word2vec and Handcraft Features
G1	LSTM layer (64 neurons) GRU layer (64 neurons) Output layer	0.7733	0.7640
G2	LSTM layer (32 neurons) GRU layer (32 neurons) Output layer	0.7705	0.7752
G3	LSTM layer (16 neurons) GRU layer (16 neurons) Output layer	0.7761	**0.7808**

Table 4.7. Cosine similarity score on different features with GRU

4.4 Multilayer Perceptron (MLP)

We have used three main MLP architectures to carried out the experiments. All the architectures with results shows in Table 4.8. Glove word vectors on a smaller neural network shows the best cosine similarity score.

	Feature set	Glove and Handcraft Features	Word2vec and Handcraft Features
M1	Dense layer (200 neurons) Dense layer (30 neurons) Output layer	0.7664	0.7730
M2	Dense layer (100 neurons) Dense layer (15 neurons) Output layer	0.7794	0.7666
M3	Dense layer (50 neurons) Dense layer (10 neurons) Output layer	**0.7770**	0.7667

Table 4.8 Cosine similarity score on different features with MLP

4.5 Convolutional Neural Network (CNN)

Using the experimental setup that we explained in 3.5.1 , experiments have been done with our best performing word2vec and glove word embeddings. Results shows in Table 4.9 that Word2vec feature outperform glove by high margin.

	Feature set	Cosine similarity
C1	Word2vec	**0.7604**
C2	Glove	0.7152

Table 4.9 Cosine similarity score with word embeddings with CNN

4.6 Ensemble Model

We explained three ensemble models in Chapter 03.5.4. Following Table shows the results of all three ensemble models.

	Model	Cosine similarity
E1	MLP model	0.7993
E2	Average model	**0.8011**
E3	Linear regression	0.7953

Table 4.10 Experiment results with Ensemble models

According to the above Table 4.10, best performing ensemble model is average ensemble model. Other ensemble models also perform better than the individual models.

Chapter 5 : Discussion

The goal is this research was to find a best model to do a sentiment analysis for short texts like twitter and stocktwist. Also we focused how the domain features are behaving in terms of their sentiment. As we discussed earlier this is a unique domain to work with as there are domain specific features in these messages.

In the previous chapter, we experiment five models which involved different set of features. We started experiments with traditional machine leaning algorithms like support vector machines (SVM) and random forest (RF). Then used ensemble machine learning algorithms like AdaBoost , XGBoost and Bagging Regressor. Even though we fed models with both handcraft and word embedding features, noticed that handcraft features performing well in machine learning models. We found that the best combination of handcraft features area lexicon, rf ngram , verb and word cluster. Lexicon feature set is the best handcraft feature set, that we ever had. We used over 25 handcraft features from seven plus lexicon dictionaries for training. Because our training set very small with 1693 tuples. So we tried as much as possible to enlarge the dataset. Even though we used some of the domain specific features like numbers, keywords and punctuations, they did not perform well enough to improve the results with best performing features. Most importantly, even word embeddings also not improved the benchmark result (0.7587) that set by the best features. Baseline score for machine leaning model which set by Akhtar et al [89] is 0.765.

Apart from the machine learning models, we created three deep learning models which perform better than traditional machine leaning algorithms. Firstly, we used BI-LSTM model which was derived from LSTM. LSTMs are the new trend in text classification. The results of LSTMs are surprised more existing baselines in machine learning. We fed BI-LSTM with combination of handcraft features and word embeddings. Then we used a GRU model along with a LSTM layer. GRUs are better performing models in small datasets. Also we used traditional MLP and a CNN networks. Our all deep learning models are outperforming Akhtar et al baseline results in each model.

5.1 Data preprocessing

Here we focused on three main objectives in data preprocessing phase. Select the most important data and reducing the noise of the data. Last objective is keeping the most important data as much as possible and refrain from important data loss in the preprocessing step. As explain in the CHAPTER 3 messages consist of two forms called message and span. Span is a part of the message which is expressed message sentiment. There are unwanted data are included in the message which are nothing to do with sentiment score. We have chosen span obviously as text data since it is well formed and dense in important data. By using the correct data column, we could gain a 5% of accuracy.

Data column	Cosine similarity with ML model
Span	0.7587 (Accuracy gain 5%)
Message	0.7229

Table 5.1 Selecting the data column

5.2 Feature Engineering

As per the main thesis contribution we have introduced a new model of hybrid feature set which includes both word embeddings and handcraft feature set. Hybrid features are used to train our three deep learning models and gain significant accuracy gain from that. In Table 5.2 described the results achievement of using hybrid features.

	Model	Word Embedding Features	Hybrid Features	Accuracy Gain
L3	Bi – LSTM model	0.7114	0.7766	9.2%
G3	GRU model	0.7398	0.7808	5.5%
M3	MLP model	0.6926	0.7770	12.2%

Table 5.2 Comparison between word embedding and hybrid features

5.3 Parameter Optimization

As discussed earlier it is very important that select the parameters which are most suitable for the dataset. Table 4.6, Table 4.7, and Table 4.8 shows the increasing of the cosine score when the parameters are changing. Below Table 5.3 shows the summary of parameter optimization on different experiment models. Mainly two models extracted which are large models with higher layer count and neurons. Small models means less hidden layers with less neurons. We can clearly see how the score is increasing while model size decreasing.

	Feature set	Cosine score	Accuracy gain
L1	Bi – LSTM Large model	0.7628	
L3	Bi – LSTM small model	**0.7766**	**+1.8%**
G1	GRU large model	0.7640	
G3	GRU small model	**0.7808**	**+2.2%**
M1	MLP Large model	0.7664	
M3	MLP small model	**0.7770**	**+1.4%**

Table 5.3 Summary of accuracy gain w.r.t parameter Optimization

5.4 Comparison with Baseline

There is significant improvement when comparing this work with baseline model by Akhtar et al in all models. There are few major improvements in this work than the baseline. Mainly we used state of art hybrid feature set to train the models. Also used the well optimized four deep learning models to get the final ensemble results. We did not use the SVR model for the final ensemble model. Instead we used MLP model for final model. Akhtar et al used MLP based architecture as their ensemble model. Our both MLP ensemble and average ensemble model beat the baseline. But average ensemble model gains higher cosine similarity score. MLP is trained very limited number of train data and since it is less performing than average ensemble

Baseline (Akhtar et al)		Proposed model		Improvement
Models	Cosine score	Models	Cosine score	
LSTM	0.727	BI-LSTM	**0.7766**	**6.82%**
GRU	0.721	GRU+ LSTM	**0.7808**	**8.29%**
CNN	0.724	CNN	**0.7604**	**5.02%**
SVR	0.765	MLP	**0.7770**	**1.16%**
Ensemble Model (MLP)	0.797	Ensemble Model (MLP)	**0.7993**	0.28%
		Average Ensemble model	**0.8011**	**0.51%**

Table 5.4 Comparison between baseline (Akhtar et al) and proposed model

5.5 Future Works

Transfer learning is the one of domain we could not consider in this work. There are lot of pre trained models introduced by researchers' like BERT, XLNET and Transformer-XL for text classification. Even though we experimented with LSTM and BI-LSTM recurrent neural network architectures, there are various kind of LSTM model architectures out there like Tree -LSTM , S-LSTM etc. Also we can extract additional meta data from tweets like, conversation / replies counts, likes, hashtags, emotions etc. Tweets are full of negations, sarcasm, irony and double negatives. We can propose a method to handle those scenarios. The evaluation criteria which we used is cosine similarity. As discussed all the researches evaluating this dataset using cosine similarity score. As this proposed method is beat the ensemble baseline of Akhtar et al and set new benchmark for this dataset, we can introduced new evaluation method for future researches.

Chapter 6 : Conclusion

This research presents the behavior of machine learning model and various deep leaning models on financial microblogs messages. Shared dataset was very small which shared by Semeval task. That was the major challenge we faced. Second was this microblog messages are in financial domain. We had to extract like numbers, keywords which is unique for financial domain. Also preprocessing task was done with extra care because there is a high possibility to missed important data. We extracted 20 plus features. Some features are handcraft and some features are word embeddings. When we created word embeddings we use both pre trained models and two self-trained models.

Comparing both machine learning models with deep learning models, we got best results from DL models beside take much longer time to train than ML models. We achieve best results on DL models and even could beat ensemble baseline model of Akhtar et al with good margin.

A new way of feature engineering method is introducing in this research. That is combined feature set of word embedding and handcraft features. It could beat the baseline models. Also we used three ensemble models named MLP, Average and Linear. Best performing ensemble model was Average model while both MLP and Average models beat the baseline.

REFERENCE

[1] Wikipedia contributors, “Sentiment analysis.” 2020.

[2] B. Liu, *Sentiment analysis and opinion mining*. 2012.

[3] S. Goonatilake, R. and Herath, “The volatility of the stock market and news,” *Int. Res. J. Financ. Econ.*, vol. 3, no. 11, pp. 53–65, 2007.

[4] M. Van De Kauter, D. Breesch, and V. Hoste, “Fine-grained analysis of explicit and implicit sentiment in financial news articles,” *Expert Syst. Appl.*, vol. 42, no. 11, pp. 4999–5010, 2015.

[5] T. Schuster, “Meta-Communication and Market Dynamics. Reflexive Interactions of Financial Markets and the Mass Media,” *SSRN eLibrary*, no. July, 2003.

[6] P. Takala, P. Malo, A. Sinha, and O. Ahlgren, “Gold-standard for Topic-specific Sentiment Analysis of Economic Texts,” *Lrec*, pp. 2152–2157, 2014.

[7] Eagle Alpha, “Sentiment analysis in the financial domain. does it work?,” 2016. .

[8] N. Sinha, “Using big data in finance: Example of sentiment-extraction from news articles.” [Online]. Available: http: //www.federalreserve.gov/econresdata/notes/feds-notes/2014/using-big-data-in-financeexample-of-sentiment-extraction-from-news-articles-20140326.html. [Accessed: 29-Jan-2016].

[9] “Fine-Grained Sentiment Analysis on Financial Microblogs and News , SemEval-2017 Task 5.” [Online]. Available: http://alt.qcri.org/semeval2017/task5/. [Accessed: 15-Dec-2016].

[10] A. Khadjeh Nassirtoussi, S. Aghabozorgi, T. Ying Wah, and D. C. L. Ngo, “Text mining for market prediction: A systematic review,” *Expert Syst. Appl.*, vol. 41, no. 16, pp. 7653–7670, 2014.

[11] Z. F. Werner, A., & Myrray, “Is all that talk just noise?,” *Inf. content internet Stock Messag. boards. J. Financ.*, no. 10, pp. 1259–1294, 2004.

[12] B. Wuthrich, V. Cho, S. Leung, D. Permunetilleke, K. Sankaran, and J. Zhang, “*Daily stock market forecast from textual web data,” *SMC’98 Conf. Proceedings. 1998 IEEE Int. Conf. Syst. Man, Cybern. (Cat. No.98CH36218)*, vol. 3, pp. 1–6, 1998.

[13] G. Pui Cheong Fung, J. Xu Yu, and W. Lam, “Stock prediction: Integrating text

mining approach using real-time news," *IEEE/IAFE Conf. Comput. Intell. Financ. Eng. Proc.*, vol. 2003-Janua, pp. 395–402, 2003.

[14] F. Jin, N. Self, P. Saraf, P. Butler, W. Wang, and N. Ramakrishnan, "Forex-Foreteller : Currency Trend Modeling using News Articles," *KDD Demo,* pp. 1470–1473, 2013.

[15] R. P. Schumaker, Y. Zhang, C. Huang, and H. Chen, "Evaluating sentiment in fi nancial news articles," *Decis. Support Syst.*, vol. 53, no. 3, pp. 458–464, 2012.

[16] Y. Yu, W. Duan, and Q. Cao, "The impact of social and conventional media on firm equity value: A sentiment analysis approach," *Decis. Support Syst.*, vol. 55, no. 4, pp. 919–926, 2013.

[17] "Dow Jones Industrial Average." [Online]. Available: https://www.marketwatch.com/investing/index/djia.

[18] "Nasdaq Composite." [Online]. Available: https://www.nasdaq.com/market-activity/index/ixic.

[19] "Morgan Stanley High-Tech Index (MSH)." [Online]. Available: https://finance.yahoo.com/quote/%5EMSH/.

[20] "Indian Sensex Index."

[21] "S&P 500 Index." [Online]. Available: https://www.marketwatch.com/investing/index/spx.

[22] S. R. Das and M. Y. Chen, "Yahoo! for Amazon: Sentiment Extraction from Small Talk on the Web," *Manage. Sci.*, vol. 53, no. 9, pp. 1375–1388, 2007.

[23] G. Rachlin, M. Last, D. Alberg, and A. Kandel, "ADM RAL: A data mining based financial trading system," *2007 IEEE Symp. Comput. Intell. Data Mining, Vols 1 2,* no. Cidm, pp. 720–725, 2007.

[24] R. P. Schumaker and H. Chen, "Textual Analysis of Stock Market Prediction Using Financial News Articles," *12th Am. Conf. Inf. Syst.*, vol. 27, no. 2, pp. 1–29, 2006.

[25] F. Peng, D. Schuurmans, S. Wang, and V. Keselj, "Language independent authorship attribution using character level language models," *Proc. tenth Conf. Eur. chapter Assoc. Comput. Linguist. 1*, pp. 267–274, 2003.

[26] A. Mahajan, L. Dey, and S. M. Haque, "Mining financial news for major events

and their impacts on the market," *Proc. - 2008 IEEE/WIC/ACM Int. Conf. Web Intell. WI 2008*, pp. 423–426, 2008.

[27] M. Hagenau, M. Liebmann, and D. Neumann, "Automated news reading: Stock price prediction based on financial news using context-capturing features," *Decis. Support Syst.*, vol. 55, no. 3, pp. 685–697, 2013.

[28] Y. Zhai, A. Hsu, and S. K. Halgamuge, "Daily Stock Price Trends Prediction," pp. 1087–1096, 2007.

[29] M.-A. Mittermayer, "Forecasting Intraday stock price trends with text mining techniques," *37th Annu. Hawaii Int. Conf. Syst. Sci. 2004. Proc.*, vol. 00, no. C, pp. 1–10, 2004.

[30] P. C. Tetlock, M. Saar-Tsechansky, and S. Macskassy, "More Than Words: Quantifying Language to Measure Firms' Fundamentals," *J. Finance*, vol. 63, no. 3, pp. 1437–1467, 2008.

[31] D. Peramunetilleke and R. K. Wong, "Currency exchange rate forecasting from news headlines," *Aust. Comput. Sci. Commun.*, vol. 24, no. 2, pp. 131–139, 2002.

[32] S. S. Groth and J. Muntermann, "An intraday market risk management approach based on textual analysis," *Decis. Support Syst.*, vol. 50, no. 4, pp. 680–691, 2011.

[33] T. Joachims, "Making large-scale support vector machine learning practical," *Adv. Kernel Methods - Support Vector Learn.*, pp. 169–184, 1999.

[34] E.~Osuna, R.~Freund, and F.~Girosi, "An Improved Training Algorithm for Support Vector Machines," *Neural Networks Signal Process. VII --- Proc. 1997 {IEEE} Work.*, pp. 276~-~285, 1997.

[35] C. Chang and C. Lin, "LIBSVM : A Library for Support Vector Machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 1–39, 2013.

[36] A. Soni, N. J. van Eck, and U. Kaymak, "Prediction of {S}tock {P}rice {M}ovements {B}ased on {C}oncept {M}ap {I}nformation," *Proc. 2007 IEEE Symp. Comput. Intell. Multicriteria Decis. Mak.*, no. Mcdm, pp. 205–211, 2007.

[37] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," *Adv. Neural Inf. Process. Dystems*, vol. 1, pp. 155–161, 1997.

[38] A. Chatrath, H. Miao, S. Ramchander, and S. Villupuram, "Currency jumps, cojumps and the role of macro news," *J. Int. Money Financ.*, vol. 40, pp. 42–62, 2014.

[39] F. Li, "The information content of forward- looking statements in corporate filings-A na??ve bayesian machine learning approach," *J. Account. Res.*, vol. 48, no. 5, pp. 1049–1102, 2010.

[40] S. C. Huang, P. J. Chuang, C. F. Wu, and H. J. Lai, "Chaos-based support vector regressions for exchange rate forecasting," *Expert Syst. Appl.*, vol. 37, no. 12, pp. 8590–8598, 2010.

[41] Tien Thanh Vu1 Shu Chang2 Quang Thuy Ha1 Nigel Col l ier3, "An experiment in integrating sentiment features for tech stock prediction in Twitter," *Open Univ. Repos. Res. Publ. other Res. outputs*, 2012.

[42] P. C. Tetlock, "Giving content to investor sentiment: The role of media in the stock market," *J. Finance*, vol. 62, no. 3, pp. 1139–1168, 2007.

[43] S. Kearney, Colm; Liu, "Textual sentiment in finance : A survey of methods and models Keywords : JEL Classification :," *Int. Rev. Financ. Anal.*, 2014.

[44] P. J. Stone, D. C. Dunphy, and M. S. Smith, *The general inquirer: A computer approach to content analysis.* Oxford, England: M.I.T. Press, 1966.

[45] Y. Ding, C. Yu, and J. Jiang, "A neural network model for semi-supervised review aspect identification," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2017, pp. 668–680.

[46] H. Jangid, S. Singhal, R. R. Shah, and R. Zimmermann, "Aspect-based financial sentiment analysis using deep learning," in *Companion Proceedings of the The Web Conference 2018*, 2018, pp. 1961–1966.

[47] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis," *Comput. Linguist.*, vol. 35, no. 3, pp. 399–433, 2009.

[48] A. Moore and P. Rayson, "Lancaster A at SemEval-2017 Task 5: Evaluation metrics matter: predicting sentiment from financial news headlines," *arXiv Prepr. arXiv1705.00571*, 2017.

[49] T. Loughran and B. McDonald, "Measuring readability in financial disclosures,"

J. Finance, vol. 69, no. 4, pp. 1643–1671, 2014.

[50] M. Hu and B. Liu, “Opinion lexicon,” *Tersedia pada http//www. cs. uic. edu/~ liub/FBS/opinion-lexicon-English. rar*, 2004.

[51] MITRE Corporation, “MPQA Opinion Corpus Release Page,” *2015 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies, Denver, Colorado, USA.*, 2015. [Online]. Available: http://mpqa.cs.pitt.edu/corpora/mpqa_corpus/.

[52] “Stock market lexicon.” [Online]. Available: https://github.com/nunomroliveira/%0Astock_market_lexicon.

[53] “Sentiwordnet.” [Online]. Available: http://sentiwordnet.isti.cnr.it/.

[54] E. Cambria, S. Poria, R. Bajpai, and B. Schuller, “SenticNet 4: A semantic resource for sentiment analysis based on conceptual primitives,” in *Proceedings of COLING 2016, the 26th international conference on computational linguistics: Technical papers*, 2016, pp. 2666–2677.

[55] E. Gilbert, “VADER : A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text.”

[56] S. Kiritchenko, X. Zhu, and S. M. Mohammad, “Sentiment analysis of short informal texts,” *J. Artif. Intell. Res.*, vol. 50, pp. 723–762, 2014.

[57] “Yelp Restaurant Sentiment Lexicon.” [Online]. Available: http://saifmohammad.com/Lexicons/Yelprestaurant-%0Areviews.zip.

[58] “Amazon Laptop Sentiment Lexicon.” [Online]. Available: http://saifmohammad.com/Lexicons/Amazon-laptop-electronics-reviews.zip.

[59] “Macquarie Semantic Orientation Lexicon.” [Online]. Available: http://saifmohammad.com/Lexicons/MSOLJune15-%0A09.txt.zip.

[60] M. Jiang, M. Lan, and Y. Wu, “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),* 2017, pp. 888–893.

[61] D. Ghosal, S. Bhatnagar, M. S. Akhtar, A. Ekbal, and P. Bhattacharyya, “IITP at SemEval-2017 task 5: an ensemble of deep learning and feature based models for financial sentiment analysis,” in *Proceedings of the 11th International Workshop*

on Semantic Evaluation (SemEval-2017), 2017, pp. 899–903.

[62] T. Cabanski, J. Romberg, and S. Conrad, “HHU at SemEval-2017 Task 5: Fine-Grained Sentiment Analysis on Financial Data using Machine Learning Methods,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017),* 2017, pp. 832–836.

[63] A. Kumar, A. Sethi, M. S. Akhtar, A. Ekbal, C. Biemann, and P. Bhattacharyya, “IITPB at SemEval-2017 task 5: Sentiment prediction in financial text,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017),* 2017, pp. 894–898.

[64] S. Kar, S. Maharjan, and T. Solorio, “RiTUAL-UH at SemEval-2017 task 5: Sentiment analysis on financial data using neural networks,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017),* 2017, pp. 877–882.

[65] Z. Nasim, “IBA-Sys at SemEval-2017 Task 5: Fine-grained sentiment analysis on financial microblogs and news,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017),* 2017, pp. 827–831.

[66] N. Tabari, A. Seyeditabari, and W. Zadrozny, “SentiHeros at SemEval-2017 Task 5: an application of sentiment analysis on financial tweets,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017),* 2017, pp. 857–860.

[67] P. Saleiro, E. M. Rodrigues, C. Soares, and E. Oliveira, “Feup at semeval-2017 task 5: Predicting sentiment polarity and intensity with financial word embeddings,” *arXiv Prepr. arXiv1704.05091,* 2017.

[68] C.-C. Chen, H.-H. Huang, and H.-H. Chen, “NLG301 at SemEval-2017 Task 5: fine-grained sentiment analysis on financial microblogs and news,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017),* 2017, pp. 847–851.

[69] S. McDonald and M. Ramscar, “Testing the distributioanl hypothesis: The influence of context on judgements of semantic similarity,” in *Proceedings of the Annual Meeting of the Cognitive Science Society,* 2001, vol. 23, no. 23.

[70] R. Schumaker and H. Chen, “Textual Analysis of Stock Market Prediction Using

Financial News Articles," *AMCIS 2006 Proc.*, p. paper 185, 2006.

[71] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv Prepr. arXiv1301.3781*, 2013.

[72] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[73] T. Mikolov, W. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 2013, pp. 746–751.

[74] X. Rong, "word2vec parameter learning explained," *arXiv Prepr. arXiv1411.2738*, 2014.

[75] F. K. Khattak, S. Jeblee, C. Pou-Prom, M. Abdalla, C. Meaney, and F. Rudzicz, "A survey of word embeddings for clinical text," *J. Biomed. Informatics X*, p. 100057, 2019.

[76] "An Intuitive Understanding of Word Embeddings: From Count Vectors to Word2Vec." [Online]. Available: https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/.

[77] F. Godin, B. Vandersmissen, W. De Neve, and R. Van de Walle, "Multimedia lab@ acl wnut ner shared task: Named entity recognition for twitter microposts using distributed word representations," in *Proceedings of the workshop on noisy user-generated text*, 2015, pp. 146–153.

[78] O. Levy and Y. Goldberg, "Dependency-based word embeddings," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2014, pp. 302–308.

[79] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[80] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguist.*, vol. 5, pp. 135–146,

2017.

[81] "FastText : facebook research blog." [Online]. Available: https://research.fb.com/fasttext/.

[82] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *ieee Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, 2018.

[83] X. Man, T. Luo, and J. Lin, "Financial Sentiment Analysis (FSA): A Survey," in *2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*, 2019, pp. 617–622.

[84] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv Prepr. arXiv1408.5882*, 2014.

[85] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 1422–1432.

[86] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[87] R. Johnson and T. Zhang, "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)*, 2017, pp. 562–570.

[88] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv Prepr. arXiv1412.3555*, 2014.

[89] M. S. Akhtar, A. Kumar, D. Ghosal, A. Ekbal, and P. Bhattacharyya, "A multilayer perceptron based ensemble technique for fine-grained financial sentiment analysis," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 540–546.

[90] M. Nguyen, "Illustrated Guide to LSTM's and GRU's: A step by step explanation."

[91] K. Cortis, T. Daudert, H. Manuela, M. Zarrouk, S. Handschuh, and B. Davis, "SemEval-2017 Task 5 : Fine-Grained Sentiment Analysis on Financial

Microblogs and News," pp. 519–535, 2017.

[92] Christiane Fellbaum, *WordNet:An Electronic lecxical Databse*. Bradford books, 1998.

[93] S. Baccianella, A. Esuli, and F. Sebastiani, "S ENTI W ORD N ET 3 . 0 : An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining," vol. 0, pp. 2200–2204, 2008.

[94] M. Hu, B. Liu, and S. M. Street, "Mining and Summarizing Customer Reviews," 2004.

[95] S. Kiritchenko, "Sentiment Analysis of Short Informal Texts," vol. 50, pp. 723–762, 2014.

[96] N. Oliveira, P. Cortez, and N. Areal, "Stock market sentiment lexicon acquisition using microblogging data and statistical measures," *Decis. Support Syst.*, vol. 85, pp. 62–73, 2016.

[97] M. Lan, C. L. Tan, S. Member, J. Su, and Y. Lu, "Supervised and Traditional Term Weighting Methods for Automatic Text Categorization," vol. 31, no. 4, pp. 721–735, 2009.

[98] Spacy documenation, "Named Entity Recognition," *Spacy documenation*, 2020. [Online]. Available: https://spacy.io/api/annotation#section-named-entities.

[99] "FastText english word vectors." [Online]. Available: https://fasttext.cc/docs/en/english-vectors.html.

[100] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *icml*, 1996, vol. 96, pp. 148–156.