Learning Stock Market Sentiment Lexicon and Sentiment-Oriented Word Vector from StockTwits

Quanzhi Li, Sameena Shah
Research and Development
Thomson Reuters
3 Times Square, New York, NY 10036
{quanzhi.li, sameena.shah@thomsonreuters.com}

Abstract

Previous studies have shown that investor sentiment indicators can predict stock market change. A domain-specific sentiment lexicon and sentiment-oriented word embedding model would help the sentiment analysis in financial domain and stock market. In this paper, we present a new approach to learning stock market lexicon from StockTwits, a popular financial social network for investors to share ideas. It learns word polarity by predicting message sentiment, using a neural network. The sentiment-oriented word embeddings are learned from tens of millions of StockTwits posts, and this is the first study presenting sentiment-oriented word embeddings for stock market. The experiments of predicting investor sentiment show that our lexicon outperformed other lexicons built by the state-of-the-art methods, and the sentiment-oriented word vector was much better than the general word embeddings.

1 Introduction

Social media has provided a rich opinion content that is valuable for diverse decision-making processes (Montoyo et al., 2012; Oliveira 2016), and sentiment analysis is being increasingly used to predict stock market variables (Antweiler and Frank 2014; Yu eta l., 2013; Schumaker et al., 2012). In particular, social media messages are a useful source for supporting stock market decisions (Bollen et al., 2011; Oliveira et al., 2013). Users of social media, such as StockTwits and Twitter, post very frequently, and this makes the real-time assessment possible, which can be exploited during the trading day. The two important sentiment data that can help sentiment analysis greatly are sentiment lexicons and word embeddings learned from large amount of data. Word embedding (word vector) has been used in many NLP tasks and noticeably improved their performance (Socher et al., 2013; Tang et al., 2014b; Vo and Zhang, 2015; Li et al., 2017). However, there has been little effort in constructing sentiment lexicon for financial domain and stock market, and in using social media as the data source. Many terms in financial market have different meanings, especially sentiment polarity, from that in other domains or sources, such as the general news articles and Twitter. For example, terms *long, short, put* and *call* have special meanings in stock market. Another example is the term *underestimate,* which is a negative term in general, but it can suggest an opportunity to buy equities when is used in stock market messages. Domain independent lexicons or general word embedding model may not perform well in financial domain. Therefore, it is necessary and important to built sentiment lexicons and word embeddings specifically for stock market.

The automatic lexicon creation approaches in previous studies are mainly based on statistic measures. There are few studies exploiting machine learning models (Tang et al, 2014a; Vo and Zhang 2016). In this study, we propose a new approach that is based on neural network, and our experiment shows that it outperforms the state-of-the-art methods. Most word embedding models only consider the syntactic and semantic information of a word, and the sentiment information is not coded in the embeddings. In this study, we extend the word vector model from (Collobert et al., 2011) by incorporating the sentiment information into the neural network to learn the embeddings; it captures the sentiment information of sentences as well as the syntactic contexts of words.

Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), pages 301–310,
Vancouver, Canada, August 3 - August 4, 2017.

The main contributions of this study are: first, we proposed a new approach based on neural network for constructing a large scale sentiment lexicon for stock market. Second, we built a sentiment-oriented word embedding (SOWE) model specifically for stock market. To our knowledge, this is the first word embedding model for stock market. The experiment shows that it outperforms the general embedding models. The lexicons and embeddings are available at https://github.com/quanzhili/stocklexicon.

2 Related Studies

There are three approaches to generating a sentiment lexicon (Liu, 2012; Al-Twairesh et al., 2016): the manual approach, dictionary-based approach, and corpus-based approach. The manual approach is usually used in conjunction with automated approaches to check the correction of the resulting lexicons.

The dictionary based method exploits the synonyms and antonyms of a word in dictionary. It usually starts with a small set of seed sentiment words, and they are looked up in the dictionary for their synonyms and antonyms, which are then added to the seed set and a new iteration process starts. Most studies adapting this approach use WordNet with different ways expanding the seed list, such as graph-based methods (Rao and Ravichandran, 2009) and distance-based measures (Kamps, 2004; Williams and Anand, 2009). The SentiWordNet lexicon created by (Esuli and Sebastiani, 2005) is the first important work based on WordNet. SentiWordNet was further expanded by (Esuli and Sebastiani, 2006; Baccianella et al., 2010) later. Bing Liu's lexicon (Hu and Liu, 2004) is also built using a dictionary based method, in which WordNet is used.

In the corpus-based approaches, the lexicon words are extracted from the corpus. Usually they also start with a set of seed sentiment words, and then expand it by discovering words with opposite or similar sentiment. For example, Turney and Littman (2002) used search engines to find a word's sentiment. They first compute the association strength between the word and a set of positive words, and then the association strength between the word and a set of negative ones. The strength with positive words minus the strength with negative ones is this word's sentiment score, which is negative if the result is negative and positive if the result is positive. Point-wise Mutual Information (PMI) is used to measure the association strength, and it is also used as one baseline method in our study. More details on PMI will be given later. Another example is the MPQA subjectivity lexicon (Wilson et al., 2005), which was built manually by annotating the subjective expressions in the MPQA corpus. As social media became popular, several studies have focused on developing sentiment lexicons from social media data, especially Twitter (Tang et al., 2014; Kiritchenko et al., 2014, Vo and Zhang, 2016; Al-Twairesh et al., 2016).

There are very few lexicons built for stock market or financial domain. A financial lexicon was manually built by (Loughran and McDonald, 2011), using documents extracted from the U.S. Securities and Exchange Commission portal from 1994 to 2008. Mao et al. (2014) proposed a procedure to automatically create a financial lexicon in Chinese, by exploiting a large news corpus, whose documents are classified as positive or negative according to the contemporaneous stock returns. Oliveira et al. (2014; 2016) used statistic measures, such as Information Gain (IG), TF.IDF and PMI, to build sentiment lexicons from StockTwits messages for stock market. The TF.IDF and PMI methods are used as two baseline methods in our study. Tang et al. (2014a) use a neural network to learn word embeddings from tweets, and then expand a set of seed sentiment words by measuring the word vector distance between seed words and other words. Vo and Zhang's approach (2016) is based on a simple neural network, to learn polarity values of a term by optimizing the prediction accuracy of message sentiment using lexicons. Our proposed approach is compared to this method. Both (Tang et al., 2014a) and (Vo and Zhang, 2016) worked on Twitter data, not stock market data.

Embeddings of a word capture both the syntactic structure and semantics of the word. The C&W model and the word2vec model are the two popular word embedding models (Collobert et al., 2011; Mikolov et al., 2013). Word embeddings have been used in many NLP tasks (Socher et al., 2014; Mass, 2012; Matt, 2015; Tang et al., 2014b; Li et al., 2016; Li et al., 2017; Vo and Zhang, 2015). Although there are quite a few studies on word embedding for Twitter data, there is no previous study on word embeddings for stock market.

Year	Number of StockTwits Messages	Number of Messages with Sentiment Label (Bullish or Bearish)	% of Messages with Sentiment Label	Number of Bullish Messages	Number of Bearish Messages	Ratio of Bullish / Bearish
2010	517,435	20,307	3.92%	17,310	2,997	5.78
2011	1,182,172	62,186	5.26%	46,823	15,363	3.05
2012	2,823,990	128,832	4.56%	95,610	33,222	2.88
2013	6,039,500	784,067	12.98%	609,709	174,358	3.50
2014	10,833,688	2,168,000	20.01%	1,774,647	393,353	4.51
2015	15,390,934	3,253,027	21.14%	2,596,182	656,845	3.95
Total	36,787,719	6,416,419	17.44%	5,140,281	1,276,138	4.03

Table 1: Statistics of StockTwits data set. Sentiment labels (bullish or bearish) are provided by message authors.

3 Learning Sentiment Lexicon and Sentiment Oriented Word Embeddings (SOWE)

In this section, we first describe how we collected the data set from StockTwits and the necessary preprocessing steps. The data set was used for both the sentiment lexicon construction and the SOWE model creation. Section 3.2 presents the proposed approach for constructing the lexicon, and Section 3.3 explains the algorithm used for building the SOWE model.

3.1 Data Collection and Preprocessing Steps

In this subsection, we describe the StockTwits data set, the basic data preprocessing steps, and how we identify phrases from StockTwits messages.

StockTwits Data Set: StockTwits is a financial social network for sharing ideas among traders. Anyone on StockTwits can contribute content – short messages limited to 140 characters that cover ideas on specific investments. Most messages have a *cashtag*, which is a stock symbol, such as $aapl, to specify the entity (stock) this message is about. We received the permission from StockTwits to access their historical message archive from year 2010 to 2015. We used this data set to build sentiment lexicons and SOWE for stock market and general financial applications.

Similar to Twitter's tweet, each StockTwits message includes a userId, number of followers, message text, timestamp, and other metadata. About 17.44% of the StockTwits messages are labeled as "bullish" or "bearish" by their authors, to show their sentiment toward the mentioned stocks. The rest of them do not have this bullish/bearish metadata. Table 1 presents the basic statistics of this data set. This table shows that the total number of messages increased greatly year by year. And we also see that the number of messages labeled as *Bullish* is much higher than that labeled as *Bearish*, with an overall ratio of 4.03. In this study, we extracted 6.4 million messages with the *Bullish* or *Bearish* sentiment label, and used them as the training data for our lexicon construction and SOWE model creation. Overall, we have about 5.1 million *Bullish* messages and 1.3 million *Bearish* messages. Below are some examples of StockTwits messages:

- *Love this company long time. $PYPL*
- *Most bullish stocks during this dip, $GOLD*
- *Another Sell Rating, Sell Rating for $AXP*
- *My favorite stock pick #in2010: $GMCR.*
- *Supermarket Stocks Rally as Tesco Plans to Axe Non-core UK Assets. $MRW*
- *Long $AMZN Oct $240 Calls*
- *for the 2009, $AXP was the $DJIA&#39;s best-performing component, having garnered a 118% gain:*
- *$f ford has not seen $10 since 9/05, but it's still a bull, be careful.*

Preprocessing Steps: Some preprocessing steps are performed to clean the messages:

- Messages that contain only cashtags, URLs, or mentions are discarded, since they do not have meaningful terms.
- Message text is converted to lower case.

- All URLs are removed. Most URLs are short URLs and located at the end of a message.
- All mentions are converted to a special symbol, for privacy reason. This includes the mentions appearing in a regular message and the user handles at the beginning of a retweet, e.g. ``RT: @bullguy''.
- All cashtags are replaced by a special symbol, to avoid cashtags to gain a polarity value related to a particular time period.
- Numbers following +, – or white space, but not followed by % (e.g. +23.3, +33, -5.52), are converted to a set of special symbols. These symbols reflect the value range of these numbers, and the range of the number determines which symbol it will be converted to. For example, +12.45 => #increase1, +20.22=> #increase2, -21.45=> #decrease2. These numbers are usually about stock price change, and so they bear sentiment information of the message. Different symbols reflect different degrees of price change.
- Similar to the above step, numbers following +, – or white space, and also followed by % (e.g. +23.34%, -5.8%), are also converted to a set of special symbols. These numbers are usually about price or volume changes. But they are based on percentage, which is different from the numbers discussed in previous step. They also convey important sentiment information.

After passing through the above preprocessing steps, the tweets are used to learn the sentiment lexicon and word embedding model.

Phrase Identification: Phrases usually convey more specific meaning than single-term words, and many phrases have a meaning that is not a simple composition of the meanings of its individual words. To identify phrases, we use the approach described in (Mikolov et al. 2013). We first find words that appear frequently together, and infrequently in other contexts. For example, “short sell” is identified as a phrase; while a bigram “they have” is not. By using this approach, we can form many reasonable phrases without greatly increasing the vocabulary size. To identify phrases, a simple data-driven approach is used, where phrases are formed based on the unigram and bigram counts, using this scoring function:

$$Score(w_i, w_j) = \frac{C(w_i, w_j) - \mu}{C(w_i) * C(w_j)} \quad (1)$$

Where *C* *(w*$_i$, *w*$_j$*)* is the frequency of word w_i *and* w_j appearing together. μ is a discounting coefficient to prevent too many phrases consisting of infrequent words to be generated. The bigrams with score above the chosen threshold are then used as phrases. Then the process is repeated a few passes over the training data with decreasing threshold value, so we can identify longer phrases having several words. For the StockTwits data set, we empirically set the maximum length of a phrase to 4 words in this study. Other parameters are set as the default values used in (Mikolov et al. 2013).

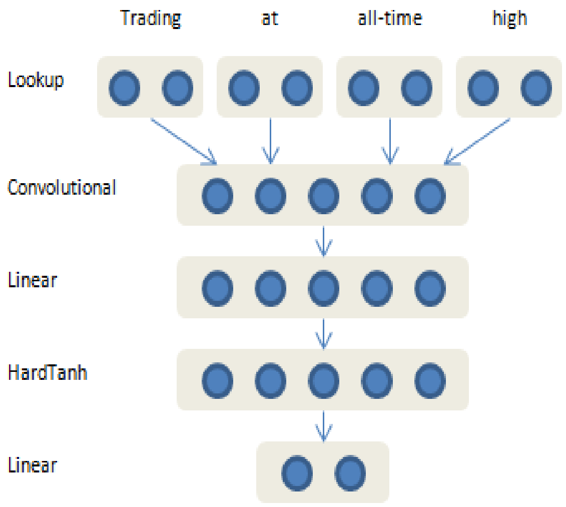

Figure 1: The neural network model for building sentiment lexicon for stock market.

3.2 Sentiment Lexicon Construction

The Proposed Approach: Most corpus-based lexicon construction approaches mainly utilize statistical measures, such as TF-IDF, GI and PMI methods. Our approach is based on a neural network model, inspired by the general network structure for processing NLP tasks (Collobert at al., 2011). Figure 1 shows the neural network we employed for learning the polarity values of a term, by predicting the sentiment value of a StockTwits message. Following (Esuli and Sebastiani, 2006; Vo and Zhang, 2016), we also use two attributes to define the sentiment of a term (word or phrase): positivity and negativity. This means each term has the form of *t* = *(p, n)*, where *p* is the positivity value and n is the negativity value. The value range is from 0 to 1 for both *p* and *n*. If the value of *p* is greater than *n*, we can say that this term has a positive sentiment, and vise versa. If *p* and *n* are close to each other, we can say that the term is neutral, bearing little sentiment information.

There are five layers in Figure 1, and from top to bottom, they are: lookup, convolutional, linear, HardTanh and linear. Using the message in this figure as an example, the words of this post are the input of this feed-forward neural network. In this example message, we assume there is no phrase identified, so there are four input terms. If there is a phrase, let's say *all-time high* is detected as a phrase, then these two words will be treated as one input term. The top layer is the lookup table for term polarity values. Because the training input is message, which varies in length, we use a convolutional layer to extract features that can be fed to standard affine layers. There are different ways to generate the representation of text segments with different lengths. In this study, we use the concatenation convolutional layer, which concatenates the layers of max, min and average of the two polarity values of all terms in the input message. This layer gives the best performance, based on our pilot experiments. The concatenation layer is expressed as follow:

$$Z(m) = [Z_p(m), Z_n(m)] \tag{2}$$

$$Z_p(m) = [Z_{p,max}(m), Z_{p,min}(m), Z_{p,ave}(m)] \tag{3}$$

$$Z_n(m) = [Z_{n,max}(m), Z_{n,min}(m), Z_{n,ave}(m)] \tag{4}$$

Where $Z(m)$ is the representation of message m, $Z_p(m)$ is for the positivity values of all the terms in this message, and $Z_n(m)$ is for negativity values of the terms. Given the convolutional layer, we can get the output of the first linear layer:

$$f^1(t) = w_1\, Z(m) + b_1 \tag{5}$$

The HardTanh layer:

$$\alpha = HardTanh(f^1(t)) \tag{6}$$

And the second linear layer, whose output,$f^2(t)$, is the sentiment score for input message m:

$$f^2(t) = w_2\alpha + b_2 \tag{7}$$

Where w_1, w_2, b_1, b_2 are the parameters of the linear layers. The non-linear HardTanh layer is to extract highly non-linear features. Without the HardTanh layer, the network would be a simple linear model. Because the hard version of the hyperbolic tangent is slightly cheaper to compute and still keep the generalization performance unchanged, it is chosen as the non-linear layer.

The *HardTanh(x)* function is defined as:

$$HardTanh(x) = \begin{array}{l} -1,\ if\ \ x <= -1 \\ x,\ if\ \ -1 <= x <= 1 \\ 1,\ if\ \ x > 1 \end{array} \tag{8}$$

Since we have just two labels for the output, negative and positive, the dimension of the second linear layer is 2. If the polarity of a StockTwits message is positive, the predicted positive score is expected to be larger than the predicted negative score, and vise versa.

The hinge loss of this model is defined as:

$$loss(m) = max(0, 1- g(m)\, f_p(m) + g(m)\, f_n(m)) \tag{9}$$

Where $g(m)$ is the gold value of message m (positive or negative), $f_p(m)$ is the predicted positive score, and $f_n(m)$ is the predicted negative score.

Model Training: The data set used for training this model is already described in previous section. To train this model, we take the derivative of the loss by back-propagation with respect to the whole set of parameters, and use AdaGrad to update the parameters (Collobert et al., 2011; Duchi et al., 2011). Each term is first initialized by randomly choosing a negative and positive value less than 0.2. The same neural network and parameters setting are used to learn the sentiment polarity for both words and phrases. A validation data set was used to tune the model hyper-parameters.

Baseline Methods for Performance Comparison: We compare our method to three other methods: TF.IDF, PMI and Vo & Zhang from (Vo and Zhang, 2016), which is based on a simple neural network. PMI and TF.IDF are the two most successful approaches building lexicons based on statistic measures. The Vo & Zhang method is the state-of-the-art approach utilizing machine learning technology. We described them briefly below.

TF.IDF is usually used for calculating the weight of a term in text analysis tasks, and it has been used in previous studies for lexicon construction (Oliveira et al. 2014; Oliveira et al., 2016; Al-Twairesh et al., 2016). To use it for computing a term's sentiment score, we first created two documents composed by all the messages of each class (bullish document and bearish document). Then, for each term, we compute its TF.IDF value for the bullish and bearish classes, respectively. And finally we can compute the sentiment score for term *t*, using the two TF.IDF values:

$$S_{TF.IDF}(t) = \frac{TF.IDF(t,p) - TF.IDF(t,n)}{TF.IDF(t,p) + TF.IDF(t,n)} \quad (10)$$

The final sentiment class depends on the value of $S_{TF.IDF}(t)$. It is bullish if the value is positive and bearish if it is negative.

PMI is a popular statistic measure used in many previous studies to develop lexicons (Mohammad et al., 2013; Oliveira et al., 2014; Oliveira et al., 2016; Al-Twairesh et al., 2016; Vo and Zhang, 2016). It is defined as:

$$PMI(x, y) = log_2 \frac{p(x,y)}{p(x)p(y)} \quad (11)$$

Where *x* and *y* are two objects, *p(x)* and *p(y)* are the probabilities of occurring *x* and *y* in the corpus, respectively, *p(x, y)* is the probability that they co-occur together. If *x* and *y* are strongly associated, PMI will be largely positive. It is highly negative if they are complementary. And if there is no significant relationship between them, it is near zero. To compute a term's sentiment score, we use both positive and negative PMI values of a term. The S_{PMI} score for term *t* is defined as follow:

$$S_{PMI}(t) = PMI(t, bullish) - PMI(t, bearish) \quad (12)$$

Where *bullish* and *bearish* refer to the sentiment label provided by the message author.

Vo & Zhang approach is a machine learning method that also optimizes the prediction accuracy of message sentiment using lexicons (Vo and Zhang, 2016). To leverage large amount of data, they use a simple neural network to train the lexicon. In this method, each term also has two polarity values: positive and negative. It uses one layer to compute the predicted sentiment probability, by adding the positive and negative values of all the terms in the input message together. Then a softmax function is used to get the predicted sentiment label for the input message. The cross-entropy error is employed as the loss function. Vo and Zhang tested their method on Twitter, using the emotions in a tweet as the indication of its polarity type. They didn't use it in the stock market domain.

3.3 Sentiment-Oriented Word Embedding

Word embedding is a dense, low-dimensional and real-valued vector for a word. The embeddings of a word capture both the syntactic structure and semantics of the word. Traditional bag-of-words and bag-of-n-grams hardly capture the semantics of words (Collobert et al., 2011; Mikolov et al. 2013).

The C&W (Collobert et al., 2011) model is a popular word embedding model. It learns word embeddings based on the syntactic contexts of words. It replaces the center word with a random word and derives a corrupted n-gram. The training objective is that the original n-gram is expected to obtain a higher language model score than the corrupted n-gram. The original and corrupted n-grams are treated as inputs of a feed-forward neural network, respectively. SOWE extends the C&W model by incorporating the sentiment information into the neural network to learn the embeddings (Collobert et al., 2011; Tang et al., 2014b); it captures the sentiment information of messages as well as the syntactic contexts of words. Given an original (or corrupted) n-gram and the polarity of a message as input, it predicts a two-dimensional vector (f_0, f_1), for each input n-gram, where (f_0, f_1) are the language model score and sentiment score of the input n-gram, respectively. There two training objectives: the original n-gram should get a higher language model score than the corrupted n-gram, and the polarity score of the original n-gram should be more aligned to the polarity label of the message than the corrupted one. The loss function is the linear combination of two losses: $loss_0$ *(t, t')* - the syntactic loss and $loss_1$ *(t, t')* - the sentiment loss:

$$loss\ (t, t') = \alpha * loss_0\ (t, t') + (1-\alpha) * loss_1\ (t, t') \quad (13)$$

The SOWE model used in this study was trained from the same 6.4 million StockTwits messages used for building sentiment lexicons; this includes 5.1 million bullish and 1.3 million bearish messages. The metadata of the SOWE model will be presented in the Experiments section

4 Experiments and Results

4.1 Evaluation of Sentiment Lexicons

In this experiment, we evaluated the lexicons built by these approaches: TF.IDF, PMI, Vo & Zhang, and our proposed approach. The same data set, which consists of 6.4 million labeled StockTwits messages, is used by these four methods. The messages are preprocessed accordingly for each method. If the difference between a term's learned positive and negative values is very small, then this term has a neutral sentiment. If we use 0.10 as the threshold to differentiate neutral terms from positive and negative terms (i.e. terms with |positive-negative| < 0.10 are neutral), our approach generated 42K sentiment words and phrases. The other three methods have slightly lower amount of sentiment terms.

Sentiment Classification: The lexicons built from these methods can be used in both unsupervised and supervised sentiment classifiers. The former is implemented by summing the sentiment scores of all tokens contained in a given message (Taboada et al., 2011; Kiritchenko et al., 2014; Vo and Zhang, 2016). If the total sentiment score is larger than 0, then the message is classified as positive. Here only one positivity attribute is required to represent a lexicon, so for lexicons with both positive and negative values for a term, the value of (*positive − negative)* is used as the score.

In this experiment, we used a supervised method for performance evaluation. There are different ways to generate features for a message using a lexicon. In this study, we follow the method used in previous studies (Zhu et al., 2014; Vo and Zhang 2016). If a lexicon has both positive and negative values for a term, then a unified score is first computed (i.e. positive – negative), and it is used to generate features described below. Given a message *m*, the features are:

- The number of sentiment tokens in *m*, where sentiment tokens are words or phrases whose sentiment scores are not zero in a lexicon.
- The total scores of negative and positive terms.
- The maximal score of all the terms in this message.
- The total sentiment score of the message.
- The sentiment score of the last term in *m*.

Data Set: we selected 30K messages that were already labeled as bullish or bearish from StockTwits's 2016 data set. They were not included in the data set used for constructing the lexicons. The amounts of bullish and bearish messages in the data set are roughly about 70% vs. 30%. We split this data set into three parts: 70% as training data, 10% as validation data and 20% for testing.

Classifier and Performance Metrics: we tried several classifiers, such as LibLiner, logistic regression and SMO. SMO gave the best results for most cases, and so we used it to compare the four lexicons. SMO is a sequential minimal optimization algorithm for training a support vector classifier. The F1 measure and accuracy are used as the performance metrics, which have been used by many previous studies.

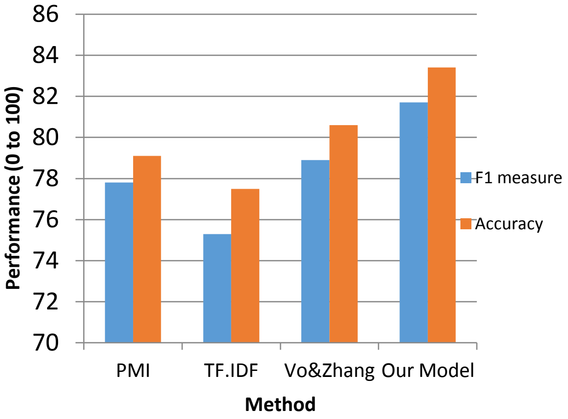

Figure 2: Sentiment classification result, based on lexicons generated by different approaches.

Result: Figure 2 presents the results. It shows that the two methods based on neural network performed better than the two statistic measures. PMI outperformed TF.IDF, which is also demonstrated by other studies (Oliveira et al. 2016). For the two models using neural networks, our proposed model outperformed the Vo & Zhang model, and the result was statistically significant at *p=0.01* using *t-test*. This result also shows that learning lexicon by predicting the accuracy of message is better than the approaches using statistic measures.

Metadata	Model		
	SOWE	StockTwitsWE	TwitterWE
Number of messages	6.4 million	37 million	200 million
Number of words in training data	87 million	505 million	2.9 billion
Number of unique words in the embedding model	165K	616K	3.5 million
Vector dimension size	300	300	300
Term frequency threshold	5	5	5
Learning context windows size	8	8	8

Table 2: Metadata of word embedding models

4.2 Evaluation of the SOWE Model

In this experiment, we evaluated the SOWE embeddings, which encode both the syntactic and sentiment information and are generated specifically for stock market. We also use sentiment classification task to do the evaluation. We compare SOWE to only embedding models, not lexicons. The reason is that they are the same type of data, and so we can use the same feature setting for them, and the experiment setting would not affect the performance comparison result. We didn't compare the SOWE to the lexicons, because they are different types of data and we need to use different approaches to generate features for them, and this will inevitably affect their performance, and make an unfair comparison. We leave this type of comparison for future research.

Word Embedding Models for Comparison: The SOWE model is compared to two types of embeddings:

StockTwitsWE: this is a general word embedding model built from StockTwits data set. This model does not have sentiment information encoded. But because the general embeddings capture both the syntactic structure and semantics of the word, it may know that the term *long* and *buy* have similar meaning in the stock market.

TwitterWE: this is a general word embedding model built from Twitter data set. This model is purely based on Twitter tweets. Although there are some tweets talking about stocks, most of the tweets are about other topics, such as sports and celebrities. We wanted to see how SOWE performs, compared to the embedding model learned from messages of a different social media platform.

These two models are built using word2vec (Mikolov et al., 2013). For StockTwitsWE, we collected 37 million StockTwits messages, which include both the labeled (bullish or bearish) and unlabeled messages. They are preprocessed using the same steps as the data set for creating sentiment lexicon. About 200 million tweets were collected from Twitter for building the TwitterWE model. The tweets date from October 2014 to October 2016. They were acquired through Twitter's public streaming API and the Decahose data (10% of Twitter's streaming data). The basic information of the three models is presented in Table 2. The embedding dimension size, word frequency threshold and window size are set based on our pilot experiments.

Experiment Settings: In this experiment, we used the same data set used in last evaluation, which consists of 30K messages. The performance metrics used are also F1 measure and accuracy. The classifier used is still SMO, which gave the best performance among several classifiers we tried.

Message Representation from Term Embeddings: In this experiment, to derive the message representation from embeddings of its terms, we use the concatenation convolutional layer, which concatenates the layers of max, min and average of message terms. This layer gave the best performance based on our pilot experiments. . The concatenation layer is expressed as follow:

$$Z(t) = [Z_{max}(t), Z_{min}(t), Z_{ave}(t)] \qquad (14)$$

where $Z(t)$ is the representation of tweet *t*.

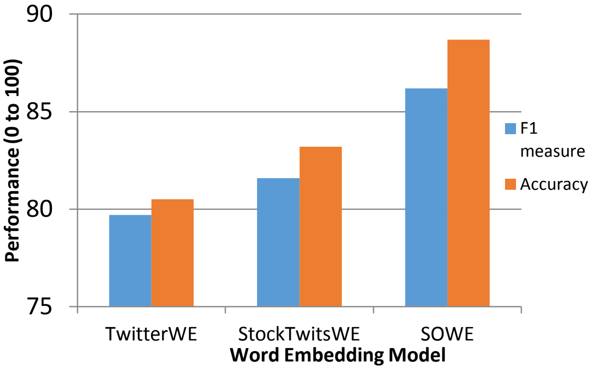

Figure 3: Comparison of the three word embedding models

Result: Figure 3 shows the comparison result of the three models. As we expected, the SOWE models performed the best, and the difference was statistically significant *at p =0.01* using *t-test*. This proves that by integrating the sentiment information into word embeddings, we can greatly improve the sentiment classification performance, although its training data set is much smaller compared to the other two models. The result also shows that the general embedding model trained on StockTwits messages performed better than the model trained on Twitter data set. One reason is that some terms in StockTwits have different sentiment meanings from the same terms in Twitter, such as *put* and *call.*

5 Conclusion

In this paper, we described a new approach based on neural network for building a large scale sentiment lexicon for stock market. We also presented a sentiment-oriented word embedding model, learned from millions of labeled messages on stock market. The experiment of measuring investor sentiment shows that our lexicon construction approach outperformed the state-of-the-art methods, and the sentiment-oriented word embedding model performed well on predicting a message's bullish/bearish polarity. The lexicons and word embedding presented in this study are available to researchers and others interested in sentiment analysis and stock market move prediction. One of our future studies is to apply our lexicon and SOWE on predicting stock market.

References

Alfred. V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling, volume 1*. Prentice-Hall, Englewood Cliffs, NJ.

Nora Al-Twairesh, Hend Al-Khalifa, AbdulMalik Al-Salman, AraSenTi: Large-Scale Twitter-Specific Arabic Sentiment Lexicons, *ACL 2016*

W. Antweiler, M. Z. Frank, Is all that talk just noise? the information content of internet stock message boards, *The Journal of Finance* 59 (3) (2004)

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. *In LREC*, volume 10

J. Bollen, H. Mao, X. Zeng, Twitter mood predicts the stock market, *Journal of Computational Science* 2 (1), 2011

Collobert, Ronan, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 2011

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*

Andrea Esuli and Fabrizio Sebastiani. 2005. Determining the semantic orientation of terms through gloss classification. *CIKM 2005.*

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. *LREC'06*, volume 6

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. *KDD 2004*

Jaap Kamps. 2004. Using Wordnet to measure semantic orientations of adjectives. *LREC 2004*

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M. Mohammad. 2014. Sentiment analysis of short informal texts. J*ournal. of Artificial Intelligence. Re*search, 50:723–762.

Quanzhi Li, Sameena Shah, Xiaomo Liu, Armineh Nourbakhsh, Word Embeddings Learned from Tweets and General Data, *The 11th International AAAI Conference on Web and Social Media (ICWSM-17).* May, 2017.

Quanzhi Li, S. Shah, X. Liu, A. Nourbakhsh & R. Fang, TweetSift: Tweet Topic Classification Based on Entity Knowledge Base and Topic Enhanced Word Embedding, *CIKM* 2016

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1)

T. Loughran, B. McDonald, When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks., *Journal of Finance,* 66 (1) (2011)

Maas, A.; Daly, R.; Pham, P.; Huang, D.; Ng, A. and Potts, C., Learning word vectors for sentiment analysis, *ACL 2012*

H. Mao, P. Gao, Y. Wang, J. Bollen, Automatic construction of financial semantic orientation lexicon from large-scale Chinese news corpus, *Financial Risks International Forum*, 2014.

Matt, T., Document Classification by Inversion of Distributed Language Representations, 2015. *ACL 2015*

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. *NIPS*, 2013.

Saif M. Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *SemEval-2013*

Montoyo, P. Martnez-Barco, A. Balahur, Subjectivity and sentiment analysis: An overview of the current state of the area and envisaged developments, *Decision Support Systems* 53 (4) (2012)

N. Oliveira, P. Cortez, N. Areal, On the predictability of stock market behavior using StockTwits sentiment and posting volume, *Progress in Artificial Intelligence, 2013*

N. Oliveira, P. Cortez, N. Areal, Automatic Creation of Stock Market Lexicons for Sentiment Analysis Using StockTwits Data, *IDES 2014*.

N. Oliveira, P. Cortez, N. Areal, stock market sentiment lexicon acquisition using microblogging data and statistical measures, *Decision Support Systems,* 2012.

Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. European Chapter of the Association for Computational Linguistics, 2009.

R. P. Schumaker, Y. Zhang, C.-N. Huang, H. Chen, Evaluating sentiment in financial news articles, Decision Support Systems 53 (3) (2012)

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C.; Ng, A. and Potts, C., Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, EMNLP 2014.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. Computational linguistics, 2011

Duyu Tang, Furu Wei, Bing Qin, Ming Zhou, and Ting Liu. 2014a. Building large-scale twitter-specific sentiment lexicon: A representation learning approach. COLING 2014

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014b. Learning sentiment specific word embedding for Twitter sentiment classification. ACL 2014

Peter Turney and Michael L Littman. 2002. Unsupervised learning of semantic orientation from a hundred-billion-word corpus. Technical report, NRC Institute for Information Technology; National Research Council Canada.

D. Vo and Y. Zhang, Target-dependant twitter sentiment classification with rich automatic features, IJCAI 2015.

D. Vo and Y. Zhang, Don’t Count, Predict! An Automatic Approach to Learning Sentiment, ACL2016.

Gbolahan K Williams and Sarabjot Singh Anand. 2009. Predicting the Polarity Strength of Adjectives Using WordNet. ICWSM 2009

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. EMNLP 2005

Xiaodan Zhu, Svetlana Kiritchenko, and Saif Mohammad. 2014. Nrc-canada-2014: Recent improvements in the sentiment analysis of tweets. In Proceedings of SemEval-2014