Leverage Financial News to Predict Stock Price Movements Using Word Embeddings and Deep Neural Networks

Yangtuo Peng and **Hui Jiang**
Department of Electrical Engineering and Computer Science,
York University, 4700 Keele Street, Toronto, Ontario, M3J 1P3, Canada
emails: tim@cse.yorku.ca, hj@cse.yorku.ca

Abstract

Financial news contains useful information on public companies and the market. In this paper we apply the popular word embedding methods and deep neural networks to leverage financial news to predict stock price movements in the market. Experimental results have shown that our proposed methods are simple but very effective, which can significantly improve the stock prediction accuracy on a standard financial database over the baseline system using only the historical price information.

1 Introduction

In the past few years, deep neural networks (DNNs) have achieved huge successes in many data modelling and prediction tasks, ranging from speech recognition, computer vision to natural language processing. In this paper, we are interested in applying the powerful deep learning methods to financial data modelling to predict stock price movements.

Traditionally neural networks have been used to model stock prices as time series for the forecasting purpose, such as in (Kaastra and Boyd, 1991; Adya and Collopy, 1991; Chan et al., 2000; Skabar and Cloete, 2002; Zhu et al., 2008). In these earlier work, due to the limited training data and computing power available back then, normally shallow neural networks were used to model various types of features extracted from stock price data sets, such as historical prices, trading volumes, etc, in order to predict future stock yields and market returns.

More recently, in the community of natural language processing, many methods have been proposed to explore additional information (mainly on-line text data) for stock forecasting, such as financial news (Xie et al., 2013; Ding et al., 2014), twitter sentiments (Si et al., 2013; Si et al., 2014), financial reports (Lee et al., 2014). For example, (Xie et al., 2013) has proposed to use semantic frame parsers to generalize from sentences to scenarios to detect the roles of specific companies (positive or negative), where support vector machines with tree kernels are used as predictive models. On the other hand, (Ding et al., 2014) has proposed to use various lexical and syntactic constraints to extract event features for stock forecasting, where they have investigated both linear classifiers and deep neural networks as predictive models.

In this paper, we propose to use the recent word embedding methods (Mikolov et al., 2013b; Liu et al., 2015; Chen et al., 2015) to select features from on-line financial news corpora, and employ deep neural networks (DNNs) to predict the future stock movements based on the extracted features. Experimental results have shown that the features derived from financial news are very useful and they can significantly improve the prediction accuracy over the baseline system that only relies on the historical price information.

2 Our Approach

In this paper, we use deep neural networks (DNNs) as our predictive models, which provide us with the advantage of easily combining various types of features from different domains with the minimum

Proceedings of NAACL-HLT 2016, pages 374–379,
San Diego, California, June 12-17, 2016.

pre-processing and normalization effort. The DNN model takes as input the features extracted from both historical price information and on-line financial news to predict the stock movements in the future (either up or down) (Peng and Jiang, 2015).

2.1 Deep Neural Networks

The structure of DNNs used in this paper is a conventional multi-layer perceptron with many hidden layers. An L-layer DNN consisting of $L-1$ hidden nonlinear layers and one output layer. The output layer is used to model the posterior probability of each output target. In this paper, we use the rectified linear activation function, i.e., $f(x) = \max(0, x)$, to compute from activations to outputs in each hidden layer, which are in turn fed to the next layer as inputs. For the output layer, we use the *softmax* function to compute posterior probabilities between two nodes, standing for *stock-up* and *stock-down*.

2.2 Features from historical price data

In this paper, for each target stock on a target date, we choose the previous five days' closing prices and concatenate them to form an input feature vector for DNNs: $P = (p_{t-5}, p_{t-4}, p_{t-3}, p_{t-2}, p_{t-1})$, where t denotes the target date, and p_m denotes the closing price on the date m. We then normalize all prices by the mean and variance calculated from all closing prices of this stock in the training set. In addition, we also compute first and second order differences among the five days' closing prices, which are appended as extra feature vectors. For example, we compute the first order difference as follows: $\Delta P = (p_{t-4}, p_{t-3}, p_{t-2}, p_{t-1}) - (p_{t-5}, p_{t-4}, p_{t-3}, p_{n-2})$. In the same way, the second order difference is calculated by taking the difference between two adjacent values in each ΔP: $\Delta\Delta P = (\Delta P_{t-3}, \Delta P_{t-2}, \Delta P_{t-1}) - (\Delta P_{t-4}, \Delta P_{t-3}, \Delta P_{t-2})$. Finally, for each target stock on a particular date, the feature vector representing the historical price information consists of P, ΔP and $\Delta\Delta P$.

2.3 Financial news features

In order to extract fixed-size features suitable to DNNs from financial news corpora, we need to preprocess the text data. For all financial articles, we first split them into sentences. We only keep those sentences that mention at least a stock name or a public company. Each sentence is labelled by the publication date of the original article and the mentioned stock name. It is possible that multiple stocks are mentioned in one sentence. In this case, this sentence is labeled several times for each mentioned stock.

We then group these sentences by the publication dates and the underlying stock names to form the samples. Each sample contains a list of sentences that were published on the same date and mention the same stock or company. Moreover, each sample is labelled as *positive* ("price-up") or *negative* ("price-down") based on its next day's closing price consulted from the CRSP financial database (Booth, 2012). In the following, we introduce our method to extract three types of features from each sample.

(1) Bag of keywords (BoK): We first select the keywords based on the recent word embedding methods in (Mikolov et al., 2013a; Mikolov et al., 2013b). Using the popular word2vec method [1], we first compute the vector representations for all words occurring in the training set. Secondly, we manually select a small set of seed words, namely, nine seed words of {*surge*, *rise*, *shrink*, *jump*, *drop*, *fall*, *plunge*, *gain*, *slump*} in this work, which are believed to have a strong indication to the stock price movements. Next, we will repeat an iterative searching process to collect other useful keywords. In each iteration, we compute the cosine distance between other words occurring in the training set and each seed word. The cosine distance represents the similarity between two words in the word vector space. For example, based on the pre-calculated word vectors, we have found other words, such as *rebound*, *decline*, *tumble*, *slowdown*, *climb*, which are very close to at least one of the above seed words. The top 10 most similar words are chosen and added back into the set of seed words at the end of each iteration. The updated seed words will be used to repeat the searching process again to find another top 10 most similar words, the size of the seed words will increase as we repeat the procedure. In this way, we have searched all words occurring in training set and finally selected 1,000 words (including the nine initial seed words) as the keywords for our prediction

[1] https://code.google.com/p/word2vec/

task. In this iterative process to collect keywords, we have found that the final set of the derived keywords is usually very similar as long as we start from a small set of seed words that all strongly indicate the stock price movements.

Finally, a 1000-dimension feature vector, called *bag-of-keywords* or *BoK*, is generated for each sample. Each dimension of the *BoK* vector is the *TFIDF* score computed for each selected keyword from the whole training corpus.

(2) Polarity score (PS): We further compute so-called *polarity* scores (Turney and Littman, 2003; Turney and Pantel, 2010) to measure how each keyword is related to stock movements and how each keyword applies to a target stock in each sentence. To do this, we first compute the point-wise mutual information for each keyword w:

$$\text{PMI}(w, pos) = \log \frac{\text{freq}(w, pos) \times N}{\text{freq}(w) \times \text{freq}(pos)},$$

where $\text{freq}(w, pos)$ denotes the frequency of the keyword w occurring in all positive samples, N denotes the total number of samples in the training set, $\text{freq}(w)$ denotes the total number of keyword w occurring in the whole training set and $\text{freq}(pos)$ denotes the total number of positive samples in the training set. Furthermore, we calculate the polarity score for each keyword w as:

$$\text{PS}(w) = \text{PMI}(w, pos) - \text{PMI}(w, neg).$$

Obviously, the above polarity score $\text{PS}(w)$ measures how each keyword is related to stock movements (either positively or negatively) and by how much.

Next, for each sentence in all samples, we need to detect how each keyword is related to the mentioned stock. To do this, we use the Stanford parser (Marneffe et al., 2006) to detect whether the target stock is a subject of the keyword or not. If the target stock is the direct object of the keyword, we assume the keyword is *oppositely* related to the underlying stock. As a result, we need to flip the sign of the polarity score. Otherwise, if the target stock is the subject of the keyword, we keep the keyword's polarity score as it is. For example, in a sentence like *"Apple slipped behind Samsung and Microsoft in a 2013 customer experience survey from Forrester Research"*, which contains an identified keyword *slip*. Based on the parsing result, we know *Apple* is the subject of *slip* while *Samsung* and *Microsoft* are the object of *slip*. Therefore, if this sentence is used as a sample for *Apple*, the above polarity score of *slip* is directly used. However, if this sentence is used as a sample for *Samsung* or *Microsoft*, the polarity score of *slipped* is flipped by multiplying -1. Finally, the resultant polarity scores are multiplied to the *TFIDF* scores to generate another 1000-dimension feature vector for each sample.

(3) Category tag (CT): During the preprocessing of the financial news data, we have discovered that certain type of events are frequently described in the financial news, and the stock price will change significantly after the publication of such financial news. In order to discover the impact of these specific events on the stock price, we further define a list of categories that may indicate the specific events or activities of a public company, which we call as category tags. In this paper, the defined category tags include: *new-product*, *acquisition*, *price-rise*, *price-drop*, *law-suit*, *fiscal-report*, *investment*, *bankrupt*, *government*, *analyst-highlights*. Each category is first manually assigned with a few words that are closely related to the category. For example, we have chosen *released*, *publish*, *presented*, *unveil* as a list of seed words for the category *new-product*, which indicates the company's announcement of new products. Similarly, we use the above word embedding model to automatically expand the word list by searching for more words that have closer cosine distances with the selected seed words. At last, we choose the top 100 words to assign to each category tag.

After we have collected all key words for each category, for each sample, we count the total number of occurrences of all words under each category, and then we take the logarithm to obtain a feature vector as $V = (\log N_1, \log N_2, \log N_3, ..., \log N_c)$, where N_c denotes the total number of times the words in category c appear in the sample. In the case where N_c is zero, it is replaced by a large negative number, for example -99.0 in this work.

2.4 Predicting Unseen Stocks via Correlation Graph

There are a large number of stocks trading in the market. However, we normally can only find a

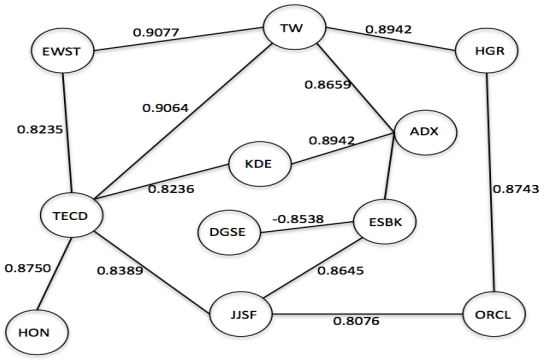

Figure 1: Illustration of a part of correlation graph

fraction of them mentioned in daily financial news. Hence, for each date, the above method can only predict those stocks mentioned in the news. In this section, we propose a new method to extend to predict more stocks that may not be directly mentioned in the financial news. Here we propose to use a stock correlation graph, shown in Figure 1, to predict those unseen stocks. The stock correlation graph is an undirected graph, where each node represents a stock and the arc between two nodes represents the correlation between these two stocks. In this way, if some stocks in the graph are mentioned in the news on a particular day, we first use the above method to predict these mentioned stocks. Afterwards, the predictions are propagated along the arcs in the graph to generate predictions for those unseen stocks.

(1) Build the graph: We choose the top 5,000 stocks from the CRSP database (Booth, 2012) to construct the correlation graph. At each time, any two stocks in the collection are selected to align their closing prices based on the related dates (between 2006/01/01 - 2012/12/31), and we only keep the stock pairs that have an overlapped trading period of at least 252 days (number of trading days in one year). Then we calculate the correlation coefficients between the closing prices of these two stocks. The computed correlation coefficients (between -1 and 1) are attached to the arc connecting these two stocks in the graph, indicating their price correlation. The correlation coefficients are calculated for all stock pairs from the collection of 5,000 stocks. In this paper, we only keep the arcs with an absolute correlation value greater than 0.8, all other edges are considered to be unreliable and pruned from the graph.

(2) Predict unseen stocks: In order to predict price movements of unseen stocks, we first take the prediction results of those mentioned stocks from the DNN outputs, by which we construct a 5000-dimension vector $\mathbf{x}$. The value of each dimension in this vector is set to indicate the probability of price moving up or down. For the dimensions corresponding to the stocks that are mentioned in the financial news, we set the values using the prediction outputs of the DNN. Since the DNN has two outputs, each of which represents the probabilities of two categories, i.e. stock price moving up or down. If a sample is recognized as price-up, we set this dimension as its probability. Otherwise, if it is recognized as price-down, we set this dimension as its probability multiplied by -1.0. Next, we set zeros for all other dimensions corresponding to unseen stocks. The graph propagation process can be mathematically represented as a matrix multiplication:

$$\mathbf{x}' = \mathbf{A}\mathbf{x}$$

where $\mathbf{A}$ is a symmetric 5000-by-5000 matrix denoting all arc correlation weights in the graph. Of course, this graph propagation may be repeated for several times until the prediction $\mathbf{x}'$ converges.

3 Dataset

The financial news data we used in this paper are provided by (Ding et al., 2014), which contains 106,521 articles from Reuters and 447,145 from Bloomberg. The news articles were published in the time period from October 2006 to December 2013. The historical stock security data are obtained from the Centre for Research in Security Prices (CRSP) database (Booth, 2012), which is published by the Chicago Business School and is widely used in the financial modelling. The CRSP database is properly adjusted for all special price events such as stock splits as well as dividend yields. We only use the security data from 2006 to 2013 to match the time period of the financial news. Base on the samples' publication dates, we split the dataset into three sets: a training set (all samples between 2006-10-01 and 2012-12-31), a validation set (2013-01-01 and 2013-06-15) and a test set (2013-06-16 to 2013-12-31). The training set contains 65,646 samples, the validation set contains 10,941 samples, and the test set contains 9,911 samples.

4 Experiments

4.1 Stock Prediction using DNNs

In the first set of experiments, we use DNNs to predict stock price movements based on a variety of features, namely producing a polar prediction of the price movement on next day (either *price-up* or *price-down*). Here we have trained a set of DNNs using different combinations of feature vectors and found that the DNN structure of 4 hidden layers (with 1024 hidden nodes in each layer) yields the best performance in the validation set. We use the historical price feature alone to create the baseline and various features derived from the financial news are added on top of it. We measure the final performance by calculating the error rate on the test set. As shown in Table 1, the features derived from financial news can significantly improve the prediction accuracy and we have obtained the best performance (an error rate of 43.13%) by using all the features discussed in Sections 2.2 and 2.3. We have also applied the structured event features proposed in (Ding et al., 2014) to our samples and the result is also listed in Table 1, which shows that our proposed features produce better performance in predicting a pool of individual stock prices.

feature combination	*error rate*
price	48.12%
price + BoK	46.02%
price + BoK + PS	43.96%
price + BoK + CT	45.86%
price + PS	45.00%
price + CT	46.10%
price + PS +CT	46.03%
price + BoK + PS + CT	**43.13%**
structured events (Ding et al., 2014)	44.79%

Table 1: Stock prediction error rates on the test set.

4.2 Predict Unseen Stocks via Correlation

Here we group all outputs from DNNs based on the dates of all samples on the test set. For each date, we create a vector $\mathbf{x}$ based on the DNN prediction results for all observed stocks and zeros for all unseen stocks, as described in section 2.4. Then, the vector is propagated through the correlation graph to generate another set of stock movement prediction. During the propagation, we compute the results from different iterations by multiplying the vector with the correlation matrix ($\mathbf{x}' = \mathbf{A}\mathbf{x}$). Our experimental results show that the prediction accuracy stops to increase after the 4th iteration. After the propagation converges, we may apply a threshold on the propagated vector to prune all low-confidence predictions. The remaining ones may be used to predict some stocks unseen on the test set. The prediction of all unseen stocks is compared with the actual stock movement on next day. Experimental results are shown in Figure 2, where the left y-axis denotes the prediction accuracy and the right y-axis denotes the percentage of stocks predicted out of all 5000 per day under various pruning thresholds. For example, using a large threshold (0.9), we may predict with an accuracy of 52.44% on 354 extra unseen stocks per day, in addition to predicting only 110 stocks per day on the test set.

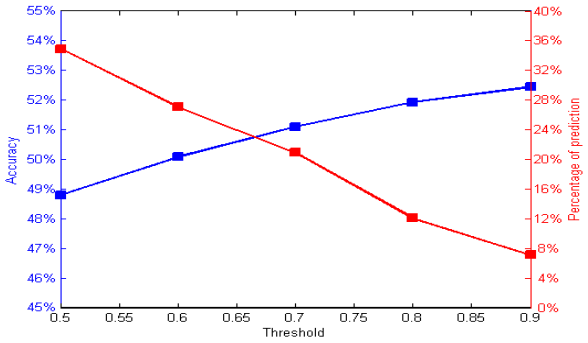

Figure 2: Predict unseen stocks via correlation

5 Conclusion

In this paper, we have proposed a simple method to leverage financial news to predict stock movements based on the popular word embedding and deep learning techniques. Our experiments have shown that the financial news is very useful in stock prediction and the proposed methods can improve the prediction accuracy on a standard financial data set.

Acknowledgement

This work is partially supported by a Discovery Grant and an Engage Grant from Natural Sciences and Engineering Research Council (NSERC) of Canada.

References

Monica Adya and Fred Collopy. 1991. How effective are neural networks at forecasting and prediction? a review and evaluation. *Journal of Forecasting*, 17:481–495.

Chicago Booth. 2012. *CRSP Data Description Guide for the CRSP US Stock Database and CRSP US Indices Database*. Center for Research in Security Prices, The University of Chicago Graduate School of Business.

Man-Chung Chan, Chi-Cheong Wong, and Chi-Chung Lam. 2000. Financial time series forecasting by neural network using conjugate gradient learning algorithm and multiple linear regression weight initialization. *Computing in Economics and Finance*, 61.

Zhigang Chen, Wei Lin, Qian Chen, Xiaoping Chen, Si Wei, Xiaodan Zhu, and Hui Jiang. 2015. Revisiting word embedding for contrasting meaning. In *Proceedings of the 53th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics.

Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2014. Using structured events to predict stock price movement: An empirical investigation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1415–1425. Association for Computational Linguistics.

Iebeling Kaastra and Milton Boyd. 1991. Designing a neural network for forecasting financial and economic time series. *Neurocomputing*, 10:215–236.

Heeyoung Lee, Mihai Surdeanu, Bill Maccartney, and Dan Jurafsky. 2014. On the importance of text analysis for stock price prediction. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*.

Quan Liu, Hui Jiang, Si Wei, Zhenhua Ling, and Yu Hu. 2015. Learning semantic word embeddings based on ordinal knowledge constraints. In *Proceedings of the 53th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics.

Marie-Catherine Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of ICLR Workshop*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, pages 3111–3119.

Yangtuo Peng and Hui Jiang. 2015. Leverage financial news to predict stock price movements using word embeddings and deep neural networks. In *arXiv:1506.07220*.

Jianfeng Si, Arjun Mukherjee, Bing Liu, Qing Li, Huayi Li, and Xiaotie Deng. 2013. Exploiting topic based twitter sentiment for stock prediction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 24–29. Association for Computational Linguistics.

Jianfeng Si, Arjun Mukherjee, Bing Liu, Sinno Jialin Pan, Qing Li, and Huayi Li. 2014. Exploiting social relations and sentiment for stock prediction. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1139–1145. Association for Computational Linguistics.

Andrew Skabar and Ian Cloete. 2002. Neural networks, financial trading and the efficient markets hypothesis. In *Proc. of the Twenty-Fifth Australasian Computer Science Conference (ACSC)*.

Peter D. Turney and Michael L. Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Trans. Inf. Syst.*, 21(4):315–346.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

Boyi Xie, Rebecca Passonneau, Leon Wu, and Germán G. Creamer. 2013. Semantic frames to predict stock price movement. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 873–883. Association for Computational Linguistics.

Xiaotian Zhu, Hong Wang, Li Xu, and Huaizu Li. 2008. Predicting stock index increments by neural networks: The role of trading volume under different horizons. *Expert Systems with Applications*, 34:3043–3054.