Attention-based Event Relevance Model for Stock Price Movement Prediction

Jian Liu[1,2], Yubo Chen[1], Kang Liu[1,2], and Jun Zhao[1,2]

[1] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[2] University of Chinese Academy of Sciences, Beijing 100049, China
{jian.liu,yubo.chen,kliu,jzhao}@nlpr.ia.ac.cn

Abstract. Stock prices, in general, can be affected by world events such as wars, natural disasters, government policies, etc. However, the correlations between events and stock prices are often implicit and the influences of events on stock prices can be in indirect ways and act in chain reactions, which brings essential difficulties for precise market prediction. In this paper, we propose an attention-based event relevance model (ATT-ERNN) to explicitly model event relevance for predicting stock price movement. Specifically, in our model, we use long short-term memory neural network (LSTM) and convolution neural network (CNN) to encode event information and stock information to distributional representations. After that, we employ attention mechanism to find related events for each stock to do price movement prediction. Attention weights in our model have a quantitative interpretation as the relevance degree of events affecting the price of a specific stock. We have conduct extensive experiments on a manually collected real-world dataset. Experimental results show the superiority of our model over many baselines, which proves the effectiveness of our model in this prediction problem.

Keywords: stock price movement prediction, attention-based neural network, event relevance modeling, deep learning

1 Introduction

Stock market prediction is the act of trying to determine the future value of a company stock [16]. The successful prediction of a stock's future price can yield significant profit obviously, making the prediction problem almost the hottest topic in finance field and artificial intelligence area.

The well-known efficient-market hypothesis (EMH) [9, 15] suggests that stock prices reflect all currently available information and any price changes based on the newly revealed relevant information. However, finding relevant information that contribute to the change of price for each individual stock faces essential difficulties because the correlations between daily events and their effect on the stock prices are usually implicit. Besides, the influences of events to stock prices can occur in indirect ways and act in chain reactions, which sets obstacles for precise market prediction.

Consider a real-world example: America was hit by hurricane Sandy in 2012, which was one of the worst storms on record that battered the Eastern United States, causing up to $20 billion in damage. This disaster drove many stocks down, including companies of the insurance industry, transportation industry, etc. However, according to financial reports, some companies benefit from this disaster! The famous Internet-based company Facebook was one of the examples. At the first glance, there seems no correlation between "hurricane causes damage" and "stock price of Facebook goes up", but when taking a deeper inspections, we can find a plausible explanation: During a storm, people will want to check in on family, friends and loved ones, as well as share photos of storm and its impact. People can't go out, so they have to use Facebook, a popular message delivery platform to communicate with other people. That is, the storm makes people gain a higher dependence on Facebook than usual, and then makes a higher expectation of investors on Facebook, causing the stock price to boost.

We argue that, for making precise prediction of a specific stock, the core question is to find which events provide related information can affect the price. Traditional approaches usually depend on complex features [1, 4, 5, 22, 21, 23] to capture the correlations, which requires a lot of domain knowledge and takes huge human ingenuity. Different from these approaches, in this paper, we propose an end-to-end attention-based event relevance model (ATT-ERNN) to learn the hidden corellations from data. Our model uses neural network based encoder to encode event information and stock information to unified semantic representations, and leverages attention mechanism to model the correlations between event-stock pairs. For a specific stock, higher attention weights show that the events are more important for affecting the price. We have conduct extensive experiments, and the experimental results show the effectiveness of the proposed model.

We summary our contributions as follows:

(1) We propose a novel end-to-end attention-based event relevance model to do stock price movement prediction. To our best knowledge, this is the first work to introduce attention mechanism to stock price movement prediction.

(2) We conduct extensive experiments on a real-world dataset. We compare our model with several baselines, and experimental results show the superiority of our proposed model.

(3) We also exploit the effect of events count and short-term, medium-term, long-term influence for stock price movement prediction.

The remaining of this paper is organized as this: Section 2 reviews the related works; Section 3 runs into the architecture of the proposed ATT-ERNN model; Section 4 illustrates the dataset construction and experimental results and Section 5 concludes the whole paper.

2 Related Works

Over the years, various methods have been proposed for stock price prediction. Traditional approaches include using historical stock prices to predict the current

price [21, 1], using user sentiment to predict stock price [5, 22, 4, 23], etc. Indeed, price history and user sentiment provide important clues for forecasting the movement of stock price. However, only use these sole evidence and completely ignoring the finance news makes these approaches behave poorly. Meanwhile, traditional approaches rely heavily on human ingenuity to design features, which often requires expert and domain knowledge.

Recently, with the development of deep learning, many neural network based models start to appear in stock price prediction. Ronnqvist et al. [17] use news document to predict potential bank crisis. In their work, they train a combined neural network to predict whether a sentence describing a bank crisis. Ding et al. [6, 7] extract structured event information from news headlines to predict the S&P 500 index. Their structure representation of events combining with CNN achieves an improvement of 6% over state-of-the-art methods. Feuerriegel et al. [8] publish a recurrent autoencoder model to predict Germany stock profit. Their model outperforms random forests as a benchmark with the accuracy by 5.66%. Deep learning based methods achieve better performance for their automatic feature extraction ability and powerful function fitting ability.

In deep learning community, attention-based neural networks gain huge popularity [3, 18, 25], which have made successful applications in neural machine translation [3], abstractive document summarization [18], image caption generation [25], etc. The advantage of the attention mechanism is that it can find the hidden correlations between two different modalities, and make the model focus only on some particular parts.

Guided by the intuition of attention mechanism, for stock price movement prediction problem, we take advantage of it to find the hidden correlations between events and stock prices, and the experiment results show its effectiveness. We believe that if combined with traditional feature-based methods, our model can get more better performance.

3 Methodology

We present the architecture of ATT-RENN model in this section. The ATT-RENN model can be further separated into three sub models: event encoding model, event relevance computing model, and price movement prediction model. In the following, we illustrate the details of each sub model.

3.1 Problem Formulation

We treat stock price movement prediction problem as a **binary classification problem**. Specifically, a stock s with its meta-information (meta-information includes company description, company location, output products, etc.) is given and the current stock price is p. Meanwhile a set of events represented as $E = \{e_i\}$ happens. The stock price changes to q after a fixed period. We assume the change of price is the consequence of the events. Comparing q with p, we can figure out whether the price goes up or down.

Thus, the problem is simplified as binary classification given s and E. We denote the result as $t \in \{0, 1\}$, which indicates the stock price goes down (t equals 0) or goes up (t equals 1). Note that, for different stocks s_j and s_k, the events set $E = \{e_i\}$ can be same. While the price movements of s_j and s_k can behave completely differently, which calls for exploiting correlations between events and stocks. We make the assumption that the influences of the events will last for a fixed period and are independent of other factors. In our experiments, we set the fixed period as one day, one week, one month to exploit the short-term, medium-term, long-term influence.

3.2 Attention-Based Event Relevance Model

The ATT-RENN model consists of three sub models: event encoding model, event relevance computing model, and prediction model. We summary the function of each sub models as below:

Event encoding model aims to encode event content to unified semantic representation. It utilizes word vectors [14] as basic lexical features and uses long short-term memory neural network (LSTM) as the encoder cell. The end-to-end structure avoids complex feature engineering for representing events.

Event relevance computing model aims to model the relevance of each event e_i with respect to a specific stock s. The attention weights have a interpretation as relevance weights of events attribute to the change of stock price.

Prediction model is a fast-forward neural network with nonlinear hidden layers. The nonlinear structure makes the model have the ability to learn complex interactions between different factors.

The overall architecture of the model is plotted in Fig. 1.

Event Encoding Model The same world event can be described in different expression ways. This variety makes the representation space for events very sparse. Several approaches [6, 7] represent the event as tuple $< S, V, O >$ to gain generalization. However, we argue that this representation is oversimplified and might lose a lot of valuable information. Instead, in our model, we propose an LSTM-based encoder to encode the entire event content to a distributional representation to tackle with sparsity problem.

LSTM belongs to Recurrent Neural Networks (RNN), and it contains three gates to maintain the information passing to capture long-term dependency [11]. LSTM shows the promising result in sentence encoding in many NLP applications [20, 10]. The computations of LSTM cells are:

$$f_t = \sigma(W_f z_t + U_f h_{t-1} + b_f) \tag{1}$$
$$i_t = \sigma(W_i z_t + U_i h_{t-1} + b_i) \tag{2}$$
$$o_t = \sigma(W_o z_t + U_o h_{t-1} + b_o) \tag{3}$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma(W_c z_t + U_c h_{t-1} + b_c) \tag{4}$$
$$h_t = o_t \circ \sigma(c_t) \tag{5}$$

Where z_t is the input of time t, and h_t is the hidden state of time t. In event encoding model, the content of each event can be represented as $e_i = \{w_{i1}, w_{i2}, ..., w_{in}\}$, where in is the content length of event e_i.

We convert all of the words to word vectors as the inputs to LSTM event encoder, and after encoding, we use the final hidden state as the event representation. Thus we transfer event set $E = \{e_i\}$ to unified semantic representations $X = \{x_i\}$, where x_i is the corresponding representation of e_i.

We prefer LSTM as event encoder for two main considerations: 1) LSTM has the ability to model long sequence and semantic of sentence. 2) The content lengths of events are usually different, and RNN is good at dealing with the variable length problem.

Event Relevance Computing Model The same events might cause different effects on different stocks. If we want to predict the stock price for a specific company, we must find which events are relevant. Note that, the relevance of events are different from companies to companies, which implies that to decide event relevance we must take stock meta-information (company information) into consideration.

Guided by this intuition, we firstly get stock meta-information representation using a convolution neural network (CNN). CNN is another type of neural network exhibits good performance in computer vision and NLP [13, 12]. We prefer CNN for its ability at extracting salient features [13]. As the meta-information of a stock is usually long and redundant, convolution neural network is the perfect tools to get salient features of the stock meta-information. We use c_i to denote the salient feature representation of stock s_i.

Then, we employ attention mechanism to find the correlations between events and stocks. The attention-based neural network has the ability to find the hidden relations between two different modalities, and we use it here to model the correlations between events and stock prices. Relevance degree of each event e_j to stock s_i (represented as c_i) is computed as:

$$r_{i,j} = \frac{exp(p_{i,j})}{\sum_j exp(p_{i,j})} \tag{6}$$

where

$$p_{i,j} = W_r[c_i; x_j] + b_r \tag{7}$$

The comprehensive information from all events is computed as:

$$a_i = \sum_j r_{i,j} x_j \tag{8}$$

We prefer linear attention (Eq. 7) for its good performance in our experiments. In Eq. 9, we use a weighted sum to get comprehensive information from all events according to their relevance weights with respect to stock s_i.

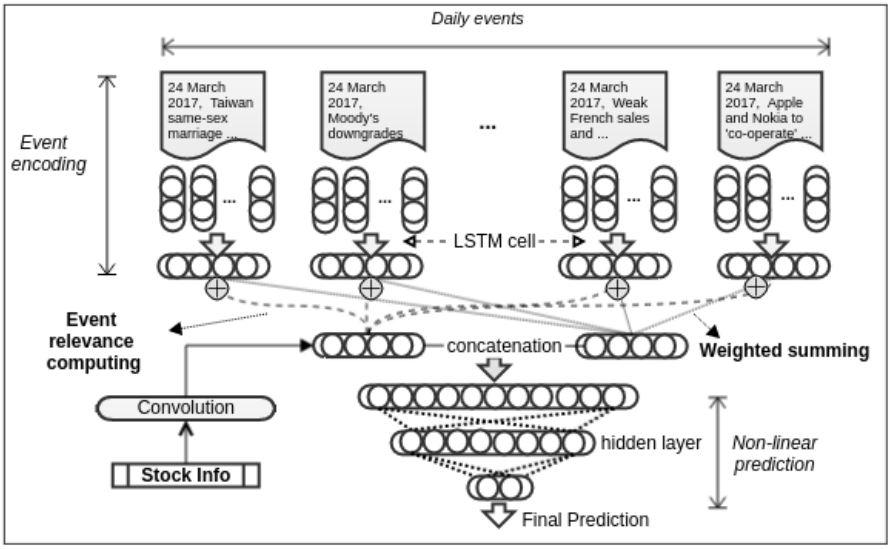

Fig. 1. The overall architecture of the ATT-RENN model

Prediction Model Before making the final prediction, we concatenate the comprehensive information a_i with the stock meta-information representation c_i. Then, we add a nonlinear hidden layer (with ReLu activation function) to let the model learn complex interactions. Finally, a softmax output layer yields the final prediction probability:

$$H_i = ReLu(W_H[a_i; c_i] + b_H) \tag{9}$$

$$O_i = Softmax(W_O H_i + b_o) \tag{10}$$

H_i is the non-linear transformation, and O_i is a two-value vector indicates the probability of price down or up of stock s_i. O_{i0} represents the probability of "goes down", and O_{i1} represents the probability of "goes up" respectively.

3.3 Model Training

We use the cross-entry loss as the loss function for model optimization. The overall loss can be represented as:

$$J(\Theta) = -\sum_i \sum_j \log(o_{i,j}) + \lambda(\Theta) \tag{11}$$

Where Θ represents all the parameters of our model, $o_{i,j}$ is the corresponding probability output by the model for stock s_i in a specific date j. λ is the

regularization weight. We also add dropout layer [19] to prevent over fitting on training phase. We use Adam as the optimization method, which gives the best result among all optimization methods in this dataset.

4 Experiments and Results

4.1 Dataset Construction

Stock price movement prediction has a large practical application value. To make the evaluations and conclusions convincing and close to the real market, we use real-world data to conduct our experiments. The collecting process can be further divided into two parallel branches:

1) **Event Data**. We obtain news articles from news publishing websites (in our experiments Baidu News[3] and Sina News[4]). We denote each event as $e_i =< tc_i, d_i >$, where tc_i is the text describing an event happens, and d_i is the date associated with the event. We use ICTCLAS [5] to do data preprocessing (word segmentation, stop words removal, etc). The following shows an event instance (we translate it to English):

$$e_i = \text{<China Internet Finance Association was founded, 2016-10-21>}$$

2) **Stock Price History**. We obtain stock price history from finance related website. Associated with each stock are the price history and the stock meta-information (company description and output products in our experiments). We use the closing price for comparison because in general closing price represents the most up-to-date valuation of a stock.

The statistics of the dateset are summarized in Table 1.

Data Statistics	**value**
Count of events	163867
Avg count of events per day	212
Avg length of event title	13
Avg length of event document	196
Count of companies (stocks)	2964
Avg length of stock meta-information	218
Count of industries	35

Table 1. Statistics of the dataset

We align stock price history with events according to time annotations. To conduct our experiments, for each industry, we sample out 20 stocks. Along with individual stocks, we also test the performance on *SH Composite Index*, which

[3] http://news.baidu.com/
[4] http://news.sina.com.cn/
[5] http://ictclas.nlpir.org/

can be seen as the average performance of all stocks. And we use data between 2015 and 2017 for training, data after 2017 for testing. One training instance consists of stock meta-information, events happen in a specific day and the up or down trend after a fixed period. We set the period to day (short-term), week (medium-term), or month (long-term), and the experiments about this setting is in section 4.4. In medium-term setting, we get a training set of size 511400 and testing set of size 81200.

4.2 Implementation and Hyper-parameter Setting

The performance of deep learning models dependent on the chosen of hyper-parameters heavily. In our experiments, we tuning our model by use grid search with 5-cross validation to find the best-fitted hyper-parameters.

We initialize word vector values from gaussian distribution, and we set the vector dimension w_d to 200 to achieve the speed and performance balance. For other hyper-parameters, we chose LSTM hidden dimension h_{lstm} from {100, 200, 300}; CNN window size w_d from {2, 3, 4, 5}; CNN filter map f_n from {64, 128, 256, 512}; non-linear hidden dimension h_p from {100, 200, 300}; learning rate l from {0.001, 0.01, 0.1, 1}; regularization parameter λ from {0.0001, 0.001, 0.01, 0.1}, and dropout rate ρ from 0.1 to 0.7. We set batch size bz to 512 without tuning because its minor influential to the final performance. The best fitted hyper-parameters are summarized in table 2.

Hyper-parameter names	**value**
word embedding w_d	200
LSTM hidden dimension h_{lstm}	300
CNN window size w_d	4
CNN filter maps f_n	128
non-linear hidden size h_p	200
learning rate l	0.001
batch size bz	512
dropout rate ρ	0.6
regularization parameter λ	0.1

Table 2. Hyper-parameter setting

4.3 Stock Price Movement Prediction Experiments

We compare ATT-RENN model with several baselines, note that all the reported performances are under the medium-term setting, namely, we set the fixed period to one week (In fact, all the models show similar consistent performance with respect to different time periods). We illustrate different models below:

RandomGus is a basic baseline model that ignores all the information totally. It simply outputs 0 (*down*) or 1 (*up*) at a chance of 50%. We also use this model to check the balance of the dataset.

StructureAvg is the method proposed in Ding [7]. Events are extracted from syntax trees by heuristic rules and are represented as $< S, V, O >$. "Avg" means we don't compute relevance for each event but we assume that all events contribute equally for the change of prices.

StructureAtt represents event the same structure as StructureAvg. Additionally, it computes events relevance use attention mechanism and combines information according to events relevance weights.

ATT-ERNNAvg uses LSTM encoder to encode events. Compared with the full ATT-TRNN model, it doesn't compute event relevance, and averaged sum strategy is used.

ATT-ERNNTtl is the full model we proposed in this paper, it only uses news titles to get event representations.

ATT-ERNNTtlCnt is the full model we proposed in this paper, it uses news titles and news contents information to compute event representations.

Performances of all these models are listed in table 3. We use accuracy as the evaluation metric.

Models	SH Composite Index	Individual stocks
RandomGus	49.6	50.7
StructureAvg	63.4	64.9
StructureAtt	63.2	65.6
ATT-ERNNAvg	**66.5**	66.8
ATT-ERNNTtl	65.9	**69.3**
ATT-ERNNTtlCnt	65.7	68.7

Table 3. Performance of different models

The nearly 50% accuracy of RandomGus shows the non-bias of the dataset. Besides, we can draw three important conclusions:

1) ATT-ERNN achieves the best result (69.3%) in individual stock price prediction when only using news titles to represent events. It outperforms the model without event relevance computing (ATT-ERNNAvg) by 2.5%. For *SH Composite Index*, ATT-ERNNAvg achieves the best result (66.5%). Since *SH Composite Index* can be seen as the ***average*** of all stocks, which just prove the reasonableness of using attention mechanism in the opposite way.

2) Model with distributed representations of events gets better performance (+3.7%) compared with model with structure representations. We explain the phenomenon as that using end-to-end method can avoid complex feature engineering and prevents error propagation from existing NLP tools.

3) Using news titles and news contents gets worse result (-0.6%). We argue that news title is the most informative text to indicate an event occur. And when we take news content into consideration, it might lead in much noise, which might confuse the model and results in worse performance.

From the third conclusion, we get the intuition that adding more data not always brings improvement of performance. To verify this idea, we conduct exper-

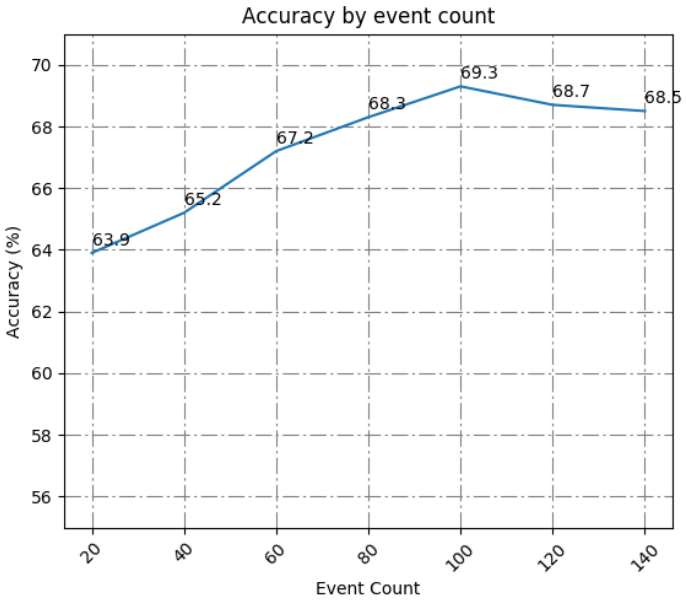

Fig. 2. The performance of our model with respect to event count

iments exploiting the prediction accuracy with respect to the number of events count. We sort all events happen in a day according to their summed TF-IDF values to get the rough order of events by importance, and we take n event from top to down. Experimental results are shown in Fig 2.

As shown in Fig 2, when data is insufficient (less than 100 events), adding more data gets better performance. However, when it hits a threshold, adding more data won't benefit any more, and the performance even starts to drop. The results justify our hypotheses that adding more data will also lead much noise, and then the model will have trouble in finding the right patterns in data, which results in poor performance.

4.4 Short-term, Medium-term, Long-term Influence

We illustrate experiments exploiting the influences of short-term (a day), medium-term (a week), and long-term (a month) on performance. We set period as above and rebuild training and testing data, and then we apply ATT-RENN model to these three settings. The experimental results are plotted in Fig 3.

From the result, we find that medium-term setting yields out the best performance. We explain the phenomenon as that: short-term setting doesn't provide enough time for the market to react, while when setting to long-term, too many other factors will dominate market trending, which also results in poor perfor-

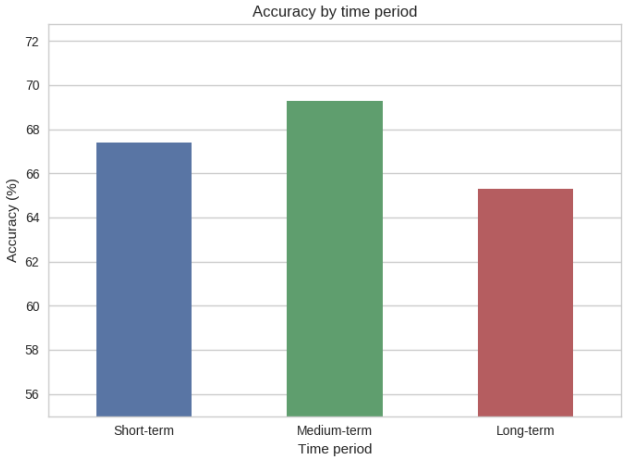

Fig. 3. The performance with respect to time period

mance. Our experiment results show that medium-term, namely one week, is the appropriate period for the market to reflect the effect of events on stock prices.

5 Conclusion and Future Work

In this paper, we propose ATT-ERNN model to exploit the implicit correlations between world events and the movement of stock prices. We conduct extensive experiments on a real-world dataset, and experimental results show the superiority of our model. In the future, we will do fine-grained quantitative analysis and add real market simulation to explore the model's ability further.

Acknowledgement

This work was supported by the Natural Science Foundation of China (No. 61533018) and the National Basic Research Program of China (No. 2014CB340503). And this research work was also supported by Google through focused research awards program.

References

1. Torben G Andersen and Tim Bollerslev. Intraday periodicity and volatility persistence in financial markets. Journal of empirical finance, 4(2):115158, 1997.
2. Batres-Estrada, Gilberto. Deep Learning for Multivariate Financial Time Series. (2015).

3. Bahdanau, Dzmitry et al. Neural Machine Translation by Jointly Learning to Align and Translate. CoRR abs/1409.0473 (2014): n. pag.
4. JBollen, Johan, Huina Mao and Xiao-Jun Zeng. Twitter mood predicts the stock market. J. Comput. Science 2 (2011): 1-8.
5. Das, Sanjiv R. and Mike Y. Chen. Yahoo! for Amazon: Sentiment Extraction from Small Talk on the Web. Management Science 53 (2007): 1375-1388.
6. Ding, Xiao, Yue Zhang, Ting Liu and Junwen Duan. Using Structured Events to Predict Stock Price Movement: An Empirical Investigation. EMNLP (2014).
7. Ding, Xiao, Yue Zhang, Ting Liu and Junwen Duan. Deep Learning for Event-Driven Stock Prediction. IJCAI (2015).
8. Feuerriegel, Stefan and Ralph Fehrer. Improving Decision Analytics with Deep Learning: the Case of Financial Disclosures. ECIS (2016).
9. Fama, Eugene F. "The behavior of stock-market prices." The journal of Business 38.1 (1965): 34-105.
10. Gers, Felix A., Nicol N. Schraudolph and Jrgen Schmidhuber. Learning Precise Timing with LSTM Recurrent Networks. Journal of Machine Learning Research 3 (2002): 115-143.
11. Hochreiter, Sepp, and Jrgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.
12. Kim, Yoon. Convolutional Neural Networks for Sentence Classification. EMNLP (2014).
13. Krizhevsky, Alex, Ilya Sutskever and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. NIPS (2012).
14. Mikolov, Tomas et al. Distributed Representations of Words and Phrases and their Compositionality. NIPS (2013).
15. Malkiel BG. The efficient market hypothesis and its critics. The Journal of Economic Perspectives. 2003 Mar 1;17(1):59-82.
16. Malkiel, Burton G. "A Random Walk Down Wall Street:: the Time-tested Strategy for Successful Investing. 2003."
17. Rnnqvist, Samuel and Peter Sarlin. Bank distress in the news: Describing events through deep learning. CoRR abs/1603.05670 (2016): n. pag.
18. Rush, Alexander M. et al. A Neural Attention Model for Abstractive Sentence Summarization. EMNLP (2015).
19. Srivastava, Nitish, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15 (2014): 1929-1958.
20. Sutskever, Ilya, Oriol Vinyals and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. NIPS (2014).
21. Taylor, Stephen J. and Xinzhong Xu. The incremental volatility information in one million foreign exchange quotations. (2003).
22. TETLOCK, PAUL C., MAYTAL SAAR-TSECHANSKY, SOFUS MACSKASSY, Brad Barber, John Griffin, Alok Kumar, Terry Murray, David Musto, Terrance Odean, Chris Parsons, Mitchell Petersen, Laura Starks and Jeremy Stein. More Than Words: Quantifying Language to Measure Firms Fundamentals. (1998).
23. Paul C Tetlock. Giving content to investor sentiment: The role of media in the stock market. The Journal of Finance, 62(3):11391168, 2007.
24. Takeuchi, Lawrence. Applying Deep Learning to Enhance Momentum Trading Strategies in Stocks. (2013).
25. Xu, Kelvin, Jimmy Ba, Jamie Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel and Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. ICML (2015).