Knowledge Graph-based Event Embedding Framework for Financial Quantitative Investments

Dawei Cheng[1], Fangzhou Yang[2], Xiaoyang Wang[3], Ying Zhang[4] and Liqing Zhang[1*]
[1] MoE Key Lab of Artificial Intelligence, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
[2] Seek Data Inc., Shanghai, China [3]Zhejiang Gongshang University, China
[4] University of Technology Sydney, Australia
dawei.cheng@sjtu.edu.cn, fangzhou.yang@seek-data.com, xiaoyangw@zjgsu.edu.cn, ying.zhang@uts.edu.au, zhang-lq@cs.sjtu.edu.cn

ABSTRACT

Event representative learning aims to embed news events into continuous space vectors for capturing syntactic and semantic information from text corpus, which is benefit to event-driven quantitative investments. However, the financial market reaction of events is also influenced by the lead-lag effect, which is driven by internal relationships. Therefore, in this paper, we present a knowledge graph-based event embedding framework for quantitative investments. In particular, we first extract structured events from raw texts, and construct the knowledge graph with the mentioned entities and relations simultaneously. Then, we leverage a joint model to merge the knowledge graph information into the objective function of an event embedding learning model. The learned representations are fed as inputs of downstream quantitative trading methods. Extensive experiments on real-world dataset demonstrate the effectiveness of the event embeddings learned from financial news and knowledge graphs. We also deploy the framework for quantitative algorithm trading. The accumulated portfolio return contributed by our method significantly outperforms other baselines.

CCS CONCEPTS

• **Information systems** → **Information extraction**; • **Applied computing** → **Electronic commerce**.

KEYWORDS

financial knowledge graph, event embedding, deep learning

ACM Reference Format:
Dawei Cheng, Fangzhou Yang, Xiaoyang Wang, Ying Zhang, and Liqing Zhang. 2020. Knowledge Graph-based Event Embedding Framework for Financial Quantitative Investments. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401427

*Corresponding author.

SIGIR '20, July 25–30, 2020, Virtual Event, China

ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401427

1 INTRODUCTION

As an informational efficient market, the price of equities can be seemed as the response to the financial news or events [3]. Thus, retrieving meaningful event representations from massive raw news is crucial to make informed investment decisions. Some recent studies have applied nature language processing (NLP) techniques to learn latent embeddings of news events for predicting market volatility [10] and constructing event-driven trading strategies [23]. Here, an event is defined as a tuple (A, P, O), where A denotes the agent, P represents the predicate and O is the object. For example, "Microsoft buys LinkedIn" can be represented as the event, where A ="Microsoft", P ="buys" and O ="LinkedIn".

Existing works mainly use sample features in event tuples, such as bags-of-words [20], named entities [43], etc. But these representations do not capture their internal relations, which limits their potentials for describing events. For example, the event "Microsoft buys LinkedIn" is represented as term-level features {"Microsoft", "buys", "LinkedIn"} alone. Since the unlinked terms cannot differentiate the acquirer (i.e., "Microsoft") and acquiree (i.e., "LinkedIn"), it is difficult to infer the price movements of the two entities (Microsoft and LinkedIn). To overcome this limitation, researchers employ open information extraction (Open IE) to obtain structured events representations [13, 36], in which the agent and object of events can be better captured. For example, the event above can be represented as (A ="Microsoft", P ="buys" and O ="LinkedIn"). But this structured representation increases the sparsity of event vectors, which limits the predictive power.

As the advance of representative learning and NLP technology, researchers tend to infer structured events by using event embeddings [12, 46]. The main idea is to learn distributed representations of structured events. Then, similar events will have similar features, even if they do not share common words, such as (A ="Microsoft", P ="buys" and O ="LinkedIn") and (A ="Google", P ="acquires" and O ="DeepMind"). The conventional event embeddings methods is based on word representations of the (A, P, O) for an event [10]. Theoretically, embeddings are appropriate for gaining good performance with a density estimator, which may misbehave in high dimensions. While, two events with similar embeddings may be quite unrelated, such as "Bill Gates quits Microsoft board" and "John Smith leaves Hilton". To overcome this, researchers [7, 11] incorporate external information from knowledge graphs (KG) [2] into the learning process of event embeddings. Then, the above two events can have very different embeddings according to their semantic

differences in KG, because "Bill Gates" is the header of "Microsoft board" and "John Smith" is more likely to be a customer at "Hilton".

However, the price movement of an individual equity not only depends on the events about itself but also is highly related to the events of other entities, and may change in a non-synchronous manner [15, 29]. When new events hit the financial market, some equities react faster than others. This correlated yet asynchronous price movement is referred as lead-lag relationship [18], which is arisen due to the different event diffusion speed over entities. For example, the event (A ="Microsoft", P ="buys" and O ="LinkedIn") will also influence the stock movements of upstream and downstream firms/competitors, such as Salesforce and Meetup. Effectively learning event embeddings from lead-lag relationship is challenging, due to the lacking of expert level knowledge of financial systems and the inaccessibility of annotated training data.

Therefore, in this paper, we propose to incorporate the knowledge of lead-lag relationship into a novel knowledge graph-based event embedding framework (KGEEF) for quantitative investment. We extract the structural relationships and event tuples from raw news texts. We store the complex relational and attributed knowledge in a financial knowledge graph (FinKG), in which the categorical vertices represent attributed entities and the edges correspond to the relations among entities. We leverage the relational and attributed knowledge to understand lead-lag relationships among entities. We propose a coherent model to jointly embed the knowledge graph information into the event embedding learning framework. Large-scale experiments on FinKG and stock market data show that incorporating lead-lag relationships from knowledge graph brings promising improvements for event embeddings. In addition, we achieve better performance in quantitative investment with the developed techniques.

In summary, the main contributions of this paper includes:

- We propose a novel and practical knowledge graph-based event embedding framework for quantitative investment, which provides investors with a new event-driven option and enables them to make informed investment decisions.
- We jointly embed the FinKG and events into a unified learning framework and propose a corresponding loss function to incorporate both relational and attributed knowledge.
- We deploy our system in Emoney.cn[1], a leading financial service provider in China, and validate the effectiveness of the developed framework in real-world scenarios.
- We construct a large scale knowledge graph in financial domain named FinKG, including attributes and relationships over 3,000 companies (China A-Shares[2]) listed in the Shanghai and Shenzhen stock exchanges. The current full version of FinKG consists of about 5.26 millions of attributed entities and 6.93 millions of edges. We prove that FinKG is helpful for quantitative investment through extensive experiments.

In the remainder of this paper, we first introduce the financial background of event embedding and knowledge graph in Section 2. In Section 3, we present the overview of the proposed event embedding framework and details of each submodule in turn. We report the experimental results in Section 4 and system implementation in Section 5. Sections 6 and 7 present the related works and conclusion.

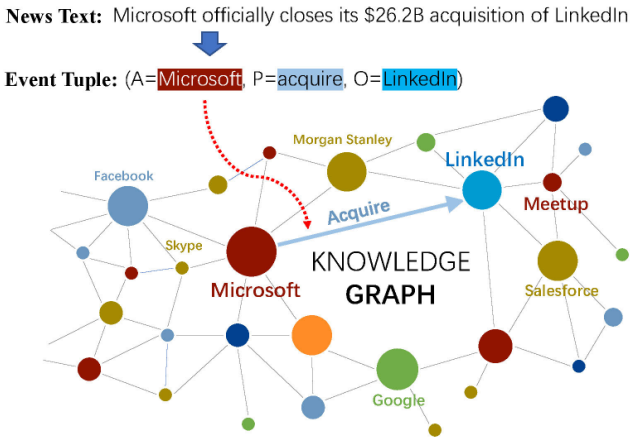

Figure 1: The illustration of the lead-lag effect in financial knowledge graph when a news event hits the market.

2 BACKGROUND

In this section, we describe the background of the lead-lag effects of news event in financial knowledge graphs and the preliminary procedure of our proposed framework.

Figure 1 presents an example that, when a news event hits the market, the corresponding lead-lag effect diffuses across the financial knowledge graph. The event impact differently on different entities in the graph. In the process, we construct the FinKG from news texts firstly. For example, when there is a fact that "In enterprise CRM market, Salesforce is the competitor of LinkedIn", we establish a relationship between "LinkedIn" and "Salesforce" as "competitor". Based on the relation extracted from massive news reports, we then map them with the mentioned entities in the knowledge graph. The relationship of linked entities includes: invest, produce, competitor, shareholder, provider, distributor, etc. They can be categorized into three kind of types: upstream and downstream relation of industry Chain, investment/employment, and cooperative/competitive relationships.

For an event tuple, it not only influences the mentioned equity, but also may have different impact on the other related companies in a non-synchronous manner. For example, an event "Microsoft buys LinkedIn" reported on Jun 13, 2016, the price of LinkedIn raised 46.81% and Microsoft declined 3.2% on that day. Afterwards, the prices of Salesforce, which is the main competitor of LinkedIn, decreased over 6% in the following two weeks. Besides, there are also other related companies may be influenced by this event, whose stock prices are reflected due to the lead-lag relationship. In addition, this lead-lag relationship is more important among manufacturing enterprises. An event on a raw material will also impact various downstream products in different propagation speed over industrial chains. Therefore, in this paper, we propose the FinKG to represent the internal relations and a joint model to learn the event embedding based on the lead-lag effect among entities.

[1] www.emony.cn
[2] https://www.investopedia.com/terms/a/a-shares.asp

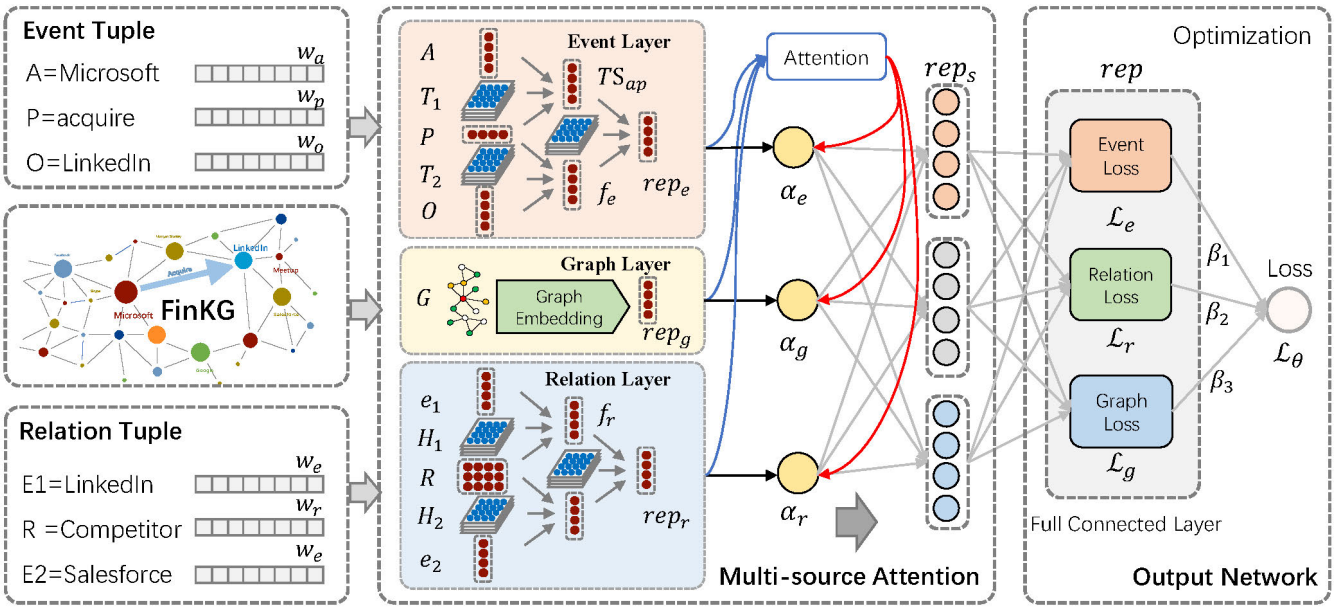

Figure 2: The general architecture of the proposed knowledge graph-based event embedding framework. It mainly includes (a) input layer, which consists of event tuples, FinKG and relation tuples; (b) the event representative learning layer. It learns the embeddings of each module and feed to a multi-source attention network for high-order representations; (c) the output network optimizes the model jointly by preserving the event, relation and graph information.

3 PROPOSED APPROACH

In this section, we first introduce the architecture of the proposed approach, and then present the procedure of constructing the KG and event tuples from raw texts. Lastly, we report each component of the model and the optimization of our proposed KG-based event embedding method.

3.1 Framework Overview

Figure 2 shows the general architecture of the proposed event embedding framework for quantitative investment. The model includes three parts: 1) Multi-source input layer, which transfers the raw texts into event tuples, relation tuples and the subsequent knowledge graph. In particular, we employ the widely-used OpenIE v5.1[3] for event tuple extraction [35]. For the relation detection, we propose a sequential learning model to detect the entity relations in news texts. The relation tuple is formulated as (e_1, R, e_2), where e_1 denotes the head entity, R is the relation type and e_2 represents the tail entity. We then leverage entity linking to store the relation tuple in the knowledge graph. 2) Event representative learning network takes the pretrained embeddings of event tuples, relation tuples and node embeddings in KG, as the inputs. Then it utilizes multi-source attention network to learn the latent representations of each source as outputs. 3) Detection and optimization module, takes the entity, event and graph embedding as inputs to infer the probability that it is a ground-truth event or relation. We utilize joint optimization on the event and relation loss to train to model. Recall that different from existing approaches, the embeddings learned by our proposed framework preserve both the event direct impact and the lead-lag effect impact on equities.

[3]https://github.com/dair-iitd/OpenIE-standalone

3.2 Relation Detection and KG Construction

Given a set of sentences $\{x_1, x_2, \cdots, x_n\}$ and corresponding entities, the relation extraction model infers the likelihood of each relation $r \in R$. Similar to Yankai et al., [28], we first represent the words as low dimensional features. In particular, given a sentence x with m words, i.e., $x = \{w_1, w_2, \cdots, w_m\}$, each word w_i is represented by a vector pretrained by BERT (Bidirectional Encoder Representations from Transformers) [8].

Then, we employ bidirectional long-short term memory networks (Bi-LSTM) to learn the information from both directions for words. The bidirectional LSTM contains a forward $\overrightarrow{LSTM}$ and a backward $\overleftarrow{LSTM}$. We obtain the hidden vector h_i from a given w_i by concatenating the forward hidden state $\overrightarrow{h}_i$ and backward hidden state $\overleftarrow{h}_i$ as follows.

$$\begin{aligned} \overrightarrow{h}_i &= \overrightarrow{LSTM}(x_i), i \in \{1, 2, \cdots, m\} \\ \overleftarrow{h}_i &= \overleftarrow{LSTM}(x_i), i \in \{1, 2, \cdots, m\} \\ h_i &= \overrightarrow{h}_i \parallel \overleftarrow{h}_i \end{aligned} \tag{1}$$

Note that, not all words contribute equally to the representation of relation extraction. Therefore, we introduce multi-head attention to learn the different importance of words for the meaning of relation and aggregate the representation of those informative words as:

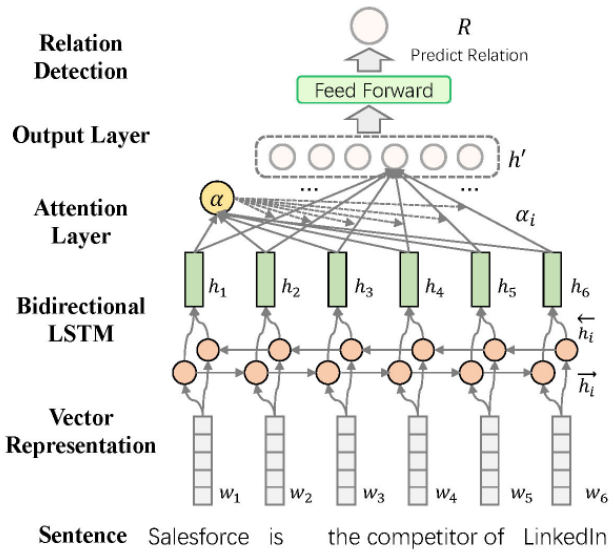

Figure 3: The architecture of relation extraction model.

$$\alpha_i = \frac{\exp(\sigma(W_{rh} h_i + b_{rh}))}{\sum_{j=1}^{m} \exp(\sigma(W_{rh} h_j + b_{rh}))}, \tag{2}$$

where W_{rh} and b_{rh} represent the weight and bias. α_i denotes the attentional weight for w_i. Then, we get the relation representation h' as:

$$h' = \sum_{i=1}^{m} \alpha_i h_i. \tag{3}$$

At last, we compute the conditional probability $p(r|S, \theta_r)$ of relation prediction through a softmax layer.

$$p(R|S, \theta_R) = \frac{\exp(W_{ro} h'_R + b_{ro})}{\sum_{k=1}^{n_R} \exp(W_{ro} h'_k + b_{ro})}, \tag{4}$$

where S indicates the sentence sets, θ_R is the parameter in relation extraction network. n_R represents the total number of relations and W_{ro}, b_{ro} are the weight and bias of output layer, respectively.

Figure 3 shows the general architecture of relation detection network. We learn and optimize the model by introducing a cross-entropy objective function at the set level as:

$$J(\theta_R) = \sum_{i=1}^{s} \log p(R_i|S_i, \theta_R), \tag{5}$$

where s denotes the number of sentence sets. In the implementation, we employ Adam algorithm [19] as the optimizer and leverage dropout [40] on the output layer to prevent overfitting.

3.3 Event Representative Learning

Based on the event tuples and constructed entity relations, we then introduce the event representative learning module of our proposed framework. As the historical information is stored as nodes and edges in the knowledge graph, which represent the level of linkage among entities, our proposed event representative learning preserves the three components in a unified framework.

3.3.1 Event Embedding. Given an event tuple $E = (A, P, O)$, where A is the actor (subject), P is the predicate (action) and O is the object that the action is preformed. We employ the neural tensor network (NTN) model as the event layer. The inputs of NTN are the pretrained d-dimension ($d = 128$) embeddings [8] of A, P and O, and the outputs are the latent event embeddings. The bilinear tensors are utilized to model the connection between the actor and the predicate, as well as the object and the predicate.

The neural tensor network is illustrated as the event learning layer in Figure 2. The middle layer TS of TNT is formulated as:

$$TS_{ap} = \mathrm{NN_e}\left(A^T T_1^{[1:k]} P + W_{ap} \begin{bmatrix} A \\ P \end{bmatrix} + b_{ap}\right), \tag{6}$$

where NN_e is a shallow neural network with Rectified Linear Unit (ReLU) as the activation function. $W_{ap} \in \mathbb{R}^{k \times 2d}$ and $b_{ap} \in \mathbb{R}^k$ are the learned weights and bias vector, respectively. $T_1^{[1:k]} \in \mathbb{R}^{d \times d \times k}$ is a tensor of k matrices, which is in shape of $d \times d$ dimensions.

For event learning, we assume that the ground truth events in the training corpus should have a higher score than the corrupted events. We constructed the corrupted events $E_c = (A_c, P, O)$ (or $E_c = (A, P, O_c)$) by replacing the A (or O) with random arguments A_c (or O_c) in our dictionary $\mathcal{D}$. $\mathcal{D}$ is consisted of all the arguments in the training corpus. We then compute the loss as:

$$\mathcal{L}_e = \sum_{i=1}^{N_e} \sum_{j=1}^{N_{ce}} \max\left(0, 1 - f_e(E^{(i)}) + f_e(E_c^{(j)})\right) + \lambda_e \|\theta_e\|_2^2, \tag{7}$$

where the θ_e is the parameter that should be trained in the event neural network (ENN) and λ_e is the penalty parameter on the L_2 normalization of θ_e, which are determined by cross validation. N_e and N_{ce} are the number of training events and the corrupted ones, respectively. $f_e(\cdot)$ denotes the transformation of event network.

3.3.2 Relation Embedding. Given a relation (e_1, R, e_2) detected by the model in Section 3.2, relation embedding is employed to learn whether two entities (e_1, e_2) are in a certain relation R. Different from the TNT of event embedding, the relation type is a tenor instead of a vector. In our implementation, the relation type is limited so that the usage of tensor could improve the expressive power of relation network. The model infer the likelihood that two entities have a certain relation by the below formulation:

$$f_r = \mathrm{NN_r}\left(u_r^T \tanh\left(e_1^T H_r^{[1:k]} e_2 + W_r \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} + b_r\right)\right), \tag{8}$$

where f_r denotes the transformation of relation embedding neural network. $\mathrm{NN_r}$ is a feed-forward neural network with ReLU as the activation function. tanh is applied in element-wise. $H_r^{[1:k]} \in R\mathbb{R}^{d \times d \times k}$ is a tensor that includes k matrices with dimension of $d \times d$. The output of tensor product $e_1^T H_r^{[1:k]} e_2$ is a vector in $\mathbb{R}^k$.

Then, we construct the relations of corrupted tuples, denoted as $R_c = (e_{c1}, R_c, e_{c2})$, where we randomly replace the e_1, R or e_2 with fake arguments in our dictionary. Finally, the loss function of relation embedding network is defined as:

$$\mathcal{L}_r = \sum_{i=1}^{N_r} \sum_{j=1}^{N_{cr}} \max\left(0, 1 - f_r(R^{(i)}) + f_r(R_c^{(j)})\right) + \lambda_r \|\theta_r\|_2^2, \tag{9}$$

where the θ_r is the parameters should be trained in the relation embedding network $f_r(\cdot)$ and λ_r is the penalty parameter on the

L_2 normalization of θ_r. N_r and N_{cr} are the number of the training tuples and the corrupted relations.

3.3.3 Graph Embedding. Given the financial knowledge graph $G = (V, E)$, the graph embedding network learns the representations of each nodes in the KG so that the entities in the same community and with similar structures should be closer in the embedding space. Note that, the nodes $v = \{v_1, v_2, \cdots, v_{N_v}\}$ in our FinKG have both attributes and labels, where N_v denotes the number of nodes. Hence, following the work of Pan et al. [34], we introduce a deep representative learning model to incorporate the attributes, labels and structure information into node features.

In the graph, each node associates with a set of words $\{w_i\}$, and some of the nodes have L labels $\{c_i\}_{i=1,\cdots,L}$. To preserve the network structures, node attributes and node labels in the embedding, we aim to maximize the loss function as:

$$\begin{aligned}\mathcal{L}_g = (1-\lambda_g)\sum_{i=1}^{N_v}\sum_{s\in\mathcal{S}}\sum_{-b\leqslant j\leqslant b, j\neq 0}\log P(v_{i+j}|v_i)\\ +\lambda_g\sum_{i=1}^{L}\sum_{-b\leqslant j\leqslant b}\log P(\omega_j|c_i)+\lambda_g\sum_{i=1}^{N_v}\sum_{-b\leqslant j\leqslant b}\log P(\omega_j|v_i),\end{aligned} \tag{10}$$

where the λ_g is the parameter that balances the network structure, attribute and label information. b denotes the windows size of sequence $\mathcal{S}$, which is generated by random walks on the network. w_j indicates the j-th word in the contextual window. As presented in Eq. 10, similar to node2vec [16], the first term is motivated by Skip-Gram [22], which models the network structure. The second term preserves the node label information and the third term models the node-content correlations. Therefore, the learned node representation is enhanced by its labels, attributes and the network structure.

3.4 Multi-source Attention and Optimization

Given the even tuple set E, relation set R and the knowledge graph G for training, our proposed approach generates the hidden representations rep_e, rep_r and rep_g from each module described in Section 3.3. Since different sources may contribute differently in the event embedding for quantitative investment task, we propose a novel multi-source attention network (MSAN) to capture the inner relations among them as follows.

$$\begin{aligned}\alpha_e &= \frac{\exp(rep_e^T w_e)}{\exp(rep_e^T w_e)+\exp(rep_r^T w_e)+\exp(rep_g^T w_e)},\\ \alpha_r &= \frac{\exp(rep_r^T w_e)}{\exp(rep_e^T w_e)+\exp(rep_r^T w_e)+\exp(rep_g^T w_e)},\\ \alpha_g &= \frac{\exp(rep_g^T w_e)}{\exp(rep_e^T w_e)+\exp(rep_r^T w_e)+\exp(rep_g^T w_e)},\\ rep_s &= \left[\alpha_e rep_e \parallel \alpha_r rep_r \parallel \alpha_g rep_g\right],\end{aligned} \tag{11}$$

where α_e, α_r and α_g are the attentional coefficients for computing the multi-source vector rep_s. w_e is the embedding of entities.

The high level representation vector rep_s learned from different sources can be employed for event embedding. We utilize the sigmoid function to output the prediction and the loss function is defined as:

$$\begin{aligned}rep &= \sigma(W_s rep_s + b_s),\\ \mathcal{L}(\theta) &= \beta_1\mathcal{L}_e + \beta_2\mathcal{L}_r + \beta_3\mathcal{L}_g,\end{aligned} \tag{12}$$

where $\theta = \{\theta_e, \theta_r, \theta_g, \theta_\alpha, W_s, b_s\}$ are all learnable parameters in the event learning, relation embedding, graph embedding and multi-source attention network. $\beta_{1,2,3}$ are the hyper parameters. W_s and b_s are the weights and bias of the output layer.

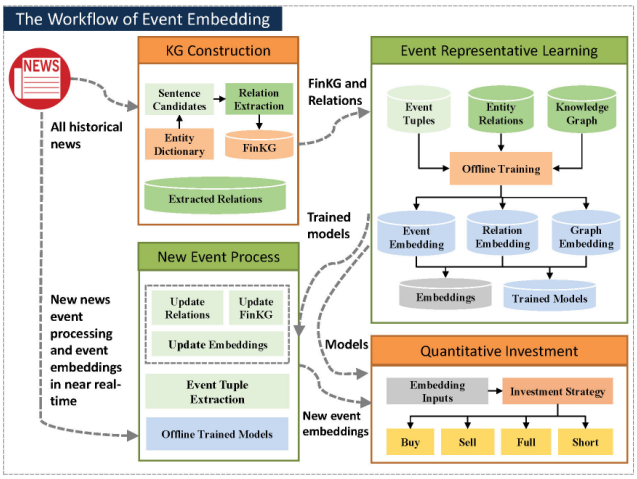

Figure 4: The workflow of event embeddings for quantitative investment, which includes FinKG construction, relation extraction, event learning and quantitative investment.

3.5 Workflow in Quantitative Investments

Figure 4 shows the workflow of event embeddings for quantitative investment. It contain four modules: historical corpus KG construction, event representative learning, new news event embedding and quantitative investment strategy.

- **Historical corpus KG construction** module produces entity relations from raw news corpus by the proposed relation detection model presented in Section 3.2. Then we create an edge of mentioned entities in FinKG if the relation is detected.
- **Event representative learning** module trains the above data by the jointly optimization of the event, relation and graph embedding networks. It produces trained models and the corresponding embeddings. The model is employed to embed new news for quantitative investments and retrained by the results from the online investment module.
- **New news event process** module firstly extracts event tuples from the news data by Open IE. Then, the constructed FinKG and learned embedding model are leveraged respectively to get the latest embeddings of events and entities. In this process, the FinKG could also be updated by new data.
- **Quantitative module** dynamically constructs the investment strategy from new embedding inputs for online decision, which is processed in near real-time. Online strategy processes the new embeddings by normalization and neutralization in turn for appropriate level allocation, which contains four levels: buy, sell, full and short.

4 EXPERIMENTS

In this section, we conduct extensive experiments to evaluate the effectiveness of our developed techniques. We first describe the experimental settings. Then, we present the results of event similarity evaluation and the effectiveness of event embeddings in quantitative investment task, which is the main task of this paper. At last, we visualize typical embeddings in case studies.

4.1 Experimental Settings

4.1.1 Datasets. We crawl the financial news from 87 major websites in Chinese financial market, including Tencent financial[4], East money[5], Shanghai[6] and Shenzhen[7] Stock Exchange, etc, from Jan 01, 2018 to Dec 31 2019. There are 5,134,936 documents in total, from which we extracted 5,262,423 entities with 6,936,818 relations to the knowledge graph (FinKG), involving 3,268 public companies listed in A-shares. We employ the first year data (from Jan 01 to Dec 31, 2018) as the training set and the next year data (from Jan 01 to Dec 31, 2019) as test set to observe the performance of learned embeddings in quantitative investments. We utilize THULAC [26] as Chinese Word Segmentation (CWS) tool to split sentences into terms. An entity may consists of one or multiple terms.

The structural event tuples are extracted from news text by Open IE [35] and the timestamps are also extracted for the alignment with stock price information. We collect the A-shares' stock market data from the distribution of Shanghai and Shenzhen Stock Exchange, China, also from Jan 01, 2018 to Dec 31 2019. There are 325,786 event tuples in total. Detailed statistics of the training and test sets are reported in Table 1. During the test windows, we train the event embedding based on previous days and generate the embedding of listed equities on each day. If an equity is influenced by multiple events, we take the average of the embeddings as the representation of that stock. After that, we predict whether the stock prices will increase in the following day. Note that, there are many options for developing a trading strategy, which is beyond the scope of this paper.

4.1.2 Algorithms. We use the following widely used approaches as baselines to highlight the effectiveness of our proposed methods. All of them are retrained with our corpus and parameters are set as their default recommendation.

- *BOW [30]:* The Bag-of-word features are used for the representation of news events and employed for stock market prediction.
- *WB-BERT [8]:* The word embedding (WB) consists of the average of each word vector in the argument of an event or equity. We employ pretrained BERT tuned by our corpus.
- *KDEB [11]:* An event embedding model that incorporates extra information from knowledge graph, named knowledge-driven event embedding, for the stock prediction.
- *EREC [9]:* The model leverages external commonsense knowledge about the intent and sentiment of the event into the representation of equities. In the implementation, we replace the commonsense knowledge ATOMIC [37] with FinKG.

[4]https://finance.qq.com/
[5]https://www.eastmoney.com/
[6]http://www.sse.com.cn/
[7]http://www.szse.cn/

Table 1: Statistics of the datasets.

Items	Training	Testing	Total
# Documents	2,466,281	2,668,655	5,134,936
# Terms	441,587,477	480,565,780	922,153,257
# Entities	2,361,965	2,900,458	5,262,423
# Relations	3,166,537	3,770,281	6,936,818
# Event Tuples	155,326	170,460	325,786
Interval	Jan 01,2018∼ Dec 31,2018	Jan 01,2019∼ Dec 31,2019	Jan 01,2018∼ Dec 31,2019

To further evaluate the power of each component in our proposed knowledge graph-based event embedding framework (KGEEF), we also include the follows variations: *KGEEF-noGL*, in which knowledge graph embedding layer is not employed; *KGEEF-noRL*, in which we remove the relation representative learning layer; *KGEEF-noATT*, in which we do not employ multi-source attention module. *KGEEF-all* denotes the full model, i.e., the one including all the techniques proposed in this paper. In our implementation, we apply the Adam algorithm [19] as the optimizer to update the parameters. The initial learning rate is set to 0.0001, and the batch size to 512 by default. Hyper parameters β_1, β_2 and β_3 are set to $\{0.46, 0.31, 0.23\}$, which are determined by cross validation.

4.1.3 Evaluation Metrics. We evaluate the performance of our proposed KGEEF with various baselines in the event similarity test (multi-class classification task) by Micro-F1, Macro-F1 and Weighted-F1 score. In detail, we count the number of correct identification of positive labels in True Positives TP, incorrect identification of positive labels in False Positives FP and incorrect identification of negative labels in False Negative FN. Then, we calculate the following evaluation metrics of Micro-F1 and Macro-F1 score as:

$$\begin{aligned} \text{Micro} - \text{F1} &= \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l}(TP_i + FP_i)} \\ \text{Macro} - \text{F1} &= \frac{1}{l}\sum_{i=1}^{l}\frac{TP_i}{TP_i + FP_i} \end{aligned} \tag{13}$$

where l denotes the number of levels in event similarity test. In our experiment, l is set to 5. Similar to Macro-F1, the metrics of weighted-F1 score is formulated as:

$$\text{Weighed} - \text{F1} = \sum_{i=1}^{l}\frac{N_i}{\sum_j N_j}\cdot\frac{TP_i}{TP_i + FP_i} \tag{14}$$

where the N_i denotes the number of label i in ground truth samples.

We employ Information Coefficient (IC) to evaluate the effectiveness of the predicted probability by event embeddings. In the domain of quantitative investment, one of the most popular type of IC is Rank IC, which is formulated as:

$$\text{Rank IC}\,(\text{Pred}, \text{Ret}, t) = corr\left(\text{Order}_{t-1}^{\text{Pred}}, \text{Order}_{t}^{\text{Ret}}\right) \tag{15}$$

where Pred means the predicted value. *corr* means the Pearson product-moment coefficient [1]. Ret means the profit return, $\text{Order}_t^{\text{Ret}}$ denotes the rank of profit return at time t.

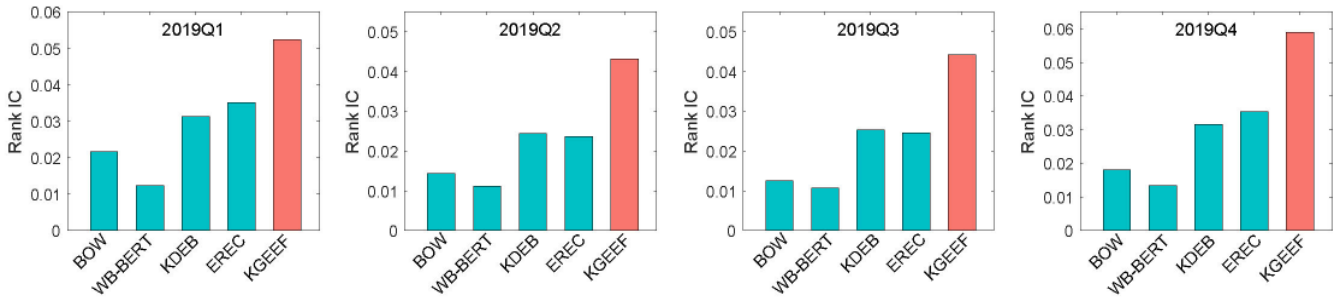

Figure 5: The averaged Rank IC of our proposed KGEEF and four compared baselines in different quarter of the test set time interval (2019). We mark the KGEEF as red and the rest baselines as blue.

Table 2: The results of event embedding similarity task.

Methods	Macro-F1	Micro-F1	Weighted-F1
BOW	0.6414	0.6051	0.6204
WB-BERT	0.6587	0.6222	0.6385
KDEB	0.7273	0.7028	0.7091
EREC	0.7287	0.7039	0.7104
KGEEF-noGL	0.7280	0.7029	0.7093
KGEEF-noRL	0.7408	0.7200	0.7221
KGEEF-noATT	0.7493	0.7299	0.7321
KGEEF-all	**0.7686****	**0.7515****	**0.7527****

4.2 Event Similarity

In this section, we evaluate whether the similarity of the learned event representations is consistent with human-labeled event similarity. We manually annotate 12,316 event pairs, which are evaluated by the financial advisers from the call centre of emoney.cn. Each event pair is associated with five independent human judgments and scored from 1 to 5. The score of 0 indicates that the event pair is completely dissimilar and 5 means that the two events are very similar. We utilize the mode score as the ground-truth label, i.e., selecting the most annotated score for each pair. In the experiment, the similarity score is computed by the cosine similarity of the embedding vector for each event pair. Then, we rank the cosine similarity scores and map them to each category (1 to 5) by a simple classifier, where the random forest is employed. Thus, the event embedding similarity measurement is evaluated as a multi-class classification task.

Table 2 shows the Macro-F1, Micro-F1 and Weighted-F1 of different baselines. The first four rows of Table 2 report the result of baseline methods. As we can see, EREC is close to KDEB. Both of them are better than BOW (Bag-of-words) and word embedding based methods, which demonstrates the importance of incorporating the structural event information and background knowledge. Rows 5 to 7 present the results of different variations of our proposed model. As we can see, KGEEF-noGL is close to EREC, both of them are learned from relation information of raw news taxt. KGEEF-noRL and KGEEF-noATT is considerably better than the rest baselines, including KGEEF-noGL. The result proves that its essential to preserve the structure embedding of knowledge graphs in an event embedding solution, which is also one of the contribution of this paper. The proposed KGEEF-all significantly outperforms all baselines in the Macro-F1, Micro-F1 and weigthed-F1.

4.3 Performance of Embeddings and Portfolio

In the event embedding performance evaluation, we first compute the Rank IC for both our proposed approach and compared methods. Then, we employ a sample random forest to predict whether the stock prices will increase and rank the predicted score in descending. We buy the increase (top score) and sell the decrease (bottom score). If a stock is among the top 100 stocks, we allocate more weights to buy them until full. On the contrary, if it is in the bottom 100, we sell them out (short). We employ the same strategy for all the methods. Note that, there are also many options for developing a trading strategy, which is beyond the scope of this paper.

Figure 5 shows the results of aggregated Rank IC. We can observe that in all quarters of 2019, our proposed method outperforms the baselines significantly, especially since 2019Q2. In particular, BOW and WB-BERT achieve lower performance in RankIC, which reflect poor correlations with the returns. A slightly improvements occurs in the KDEB and EREC that by incorporating the external knowledge and commonsense relations. KGEEF considerably better than all compared baselines, which demonstrates the effectiveness of our proposed techniques.

To further investigate the value of our proposed event embedding framework, we conduct experiments to examine the accumulate return of the constructed portfolio. Figure 6 shows the accumulated returns across the test time interval. As we can see, BOW and WB-BERT is close to the market performance of CSI 300 index[8], which is accordance with their Rank IC performance. Since 2019Q2, KGEEF leads the returns and enlarges the gap with compared baselines. The reason might be that when the market goes down, the lead-lag relationship learned by our proposed method could warn the signal in advance and proceed to sell the related stocks. Thus, KGEEF achieves the bust performance since 2019Q2 and gains over 20% improvements with the best baseline at the end of 2019Q4.

4.4 Case Studies

In case studies, we choose three popular indicators in the quantitative investment to examine the effectiveness of top and bottom stocks predicted by our proposed method. Specifically, we select

[8]http://www.csindex.com.cn/en/indices/index-detail/000300

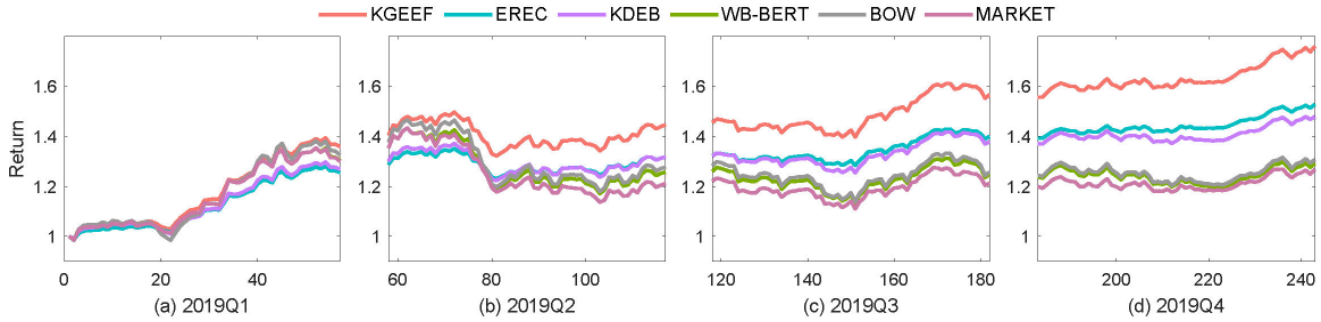

Figure 6: The accumulated returns gained in the test set by the proposed method and compared baselines. For better illustration, we divide it into four quarters view.

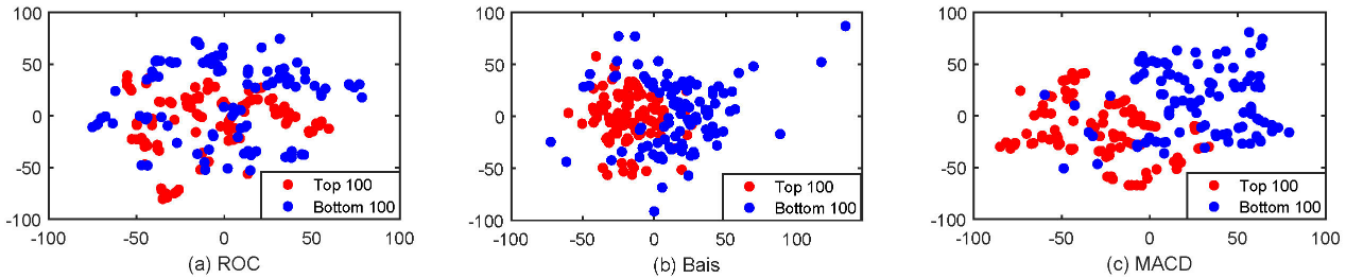

Figure 7: The t-SNE visualization of the learned embeddings for selected stocks (Top 100 and Bottom 100).

ROC, Bias and MACD as example indicators. ROC measures the amplitude of variation compared with previous sequences and Bias implies the deviation level of prices compared to sample moving average. The MACD is computed by the difference between two exponential moving average lines, where one is from a short period (12 days) and the rest is from a long period (26 days). Then, we employ t-SNE [31] to visualize the top 100 weakened and top 100 strengthened equities into two dimension space and color with blue and red respectively.

As we can see from Figure 7, the layout of top 100 and bottom 100 stocks are quite meaningful, in which nodes with same colors are distributed closer. Besides, the stocks in two categories distribute separately in the low-dimension space, which is more significant in the layouts of MACD. The results show that our proposed method is able to assign similar signals for the strengthened and weakened equities. Please note that we ignore the transaction costs in all the experiments for the sake of the simplicity.

5 SYSTEM IMPLEMENTATION

In this section, we report the system implementation of our proposed framework. We first present the technical details of the system and then show the application of our model in mobile mini apps. Finally, we report the full version of applications in the web-based desktop platform. During the implementation, we employ distributed Scrapy [33] as the web crawler, in which the data are transformed by Kafka [38] and in-memory database Redis [32] with bloom-filter to remove the duplicate news. The Elastic Search [21] is utilized as the database of raw texts and Neo4j [17] as the graph database. The training model is implemented by PyTorch and written in Python, Java and Scala.

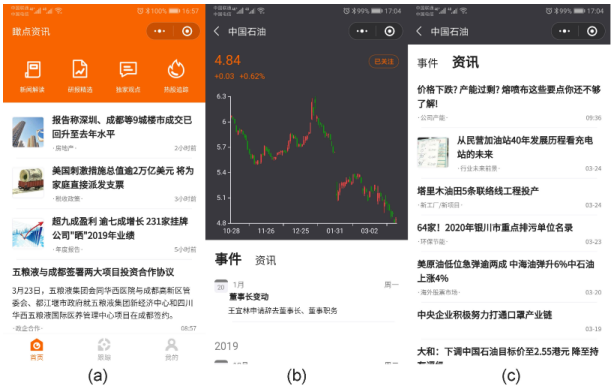

Figure 8: The interface of our proposed method in the mobile mini-apps. It includes event extraction from raw news and visualization of how does the event impact stock prices.

Figure 8 shows the interface of the system deployed in a mobile mini application. As we can see, Figure 8(a) shows the aggregated hot news of the constructed portfolio and the event view is displayed in by each stock. As shown in Figure 8(b), the prices and corresponding events of China National Petroleum Corporation(CBPC: 601857) are reported. Thanks to the knowledge graph based embedding, in the related events view Figure 8(c), our application successfully report the news about Tarim oilfield (the upstream factory), China National Offshore Oil Corporation (Competitor) and the increasing demands of melt-blown fabric (a downstream product of petrochemical industry), because it is a raw material of protective facial mask. Figure 9 shows the interface of web-based desktop view. We provides a new portfolio construction methods

Figure 9: The interface of our proposed KGEEF in a web-based portfolio management product.

based the event embeddings on the top left filters. The system displays the selected stocks by the event-driven strategy. The right part shows that a event predict signal on the stock prices, where S denotes the sell point and B indicates the buy point. We also present the detailed description of why there is a sell or buy point based on the inference by events and knowledge graphs. The result provides informative advise for our customers with a new KG-based event-driven investment option.

6 RELATED WORKS

Efficient market hypothesis [14] states that the price of an equity reflects all of the information available, and every player has a certain degree of accessing to the information. Over the past five decades, the problem of how to extract useful information for stock market has been extensively studied from the fields of finance, mathematics and computer science. In information retrieval (IR), there are two types of information that have been most studied for the quantitative investment.

Time-series market data. Existing studies [6, 47] utilize historical time series market data to predict the upcoming stock prices based on various machine learning technologies, such as factor analysis [5], tensor theory [4], indicator optimization [27], etc. Wang et al. [42] propose an attentional reinforcement learning method to learn from the side of finance, such as the balance of risk and return, the resistance of extreme loss. Researchers believe that predictions can be made through mining the historical market patterns from price and volume movements. However, their works ignore the financial news information, which is one key component of marketing prediction.

Unstructured text data. With the recent advances of NLP techniques, previous studies have found that financial news can influence the market price of an equity [24, 25]. Some researchers employ semantic analysis on the news documents [44]. For example, Tetlock et al. [41] examine how qualitative information is incorporated in aggregate market valuations. Si et al. [39] try to regress the stock price with topic-sentiment time-series. In [10, 11], authors leverage structural event extraction method on the financial news data and learn low-dimensional representations of the events. Zhang et al. [45] improve the event embeddings by incorporating the external knowledge from KG. Ding et al. [9] further incorporate intent and sentiment information into event embeddings.

However, previous work mainly focuses on using the nature information of the events or external knowledge, such as intent and sentiment of event information, to enhance the representations. In this paper, we propose a knowledge graph-based event embedding framework, which not only learning from the relations of the event arguments in the KG, but also preserving lead-lag effects on the other influenced entities.

7 CONCLUSION

High quality representations of the events and the corresponding entities are valuable for event-driven quantitative investments. In this paper, we present a novel knowledge graph-based event embedding framework to learn informative representations from not only the relations of event arguments, but also the lead-lag relations across the financial knowledge graphs, especially the effects on upstream and downstream of the industry chain. In particular, we integrate the graph embedding layer and relation learning layer in the learning process of event embedding framework with multi-source attention mechanism and joint optimization. Extensive experiments show that our method outperforms other state-of-the-art baselines significantly in event similarity, Rank IC and accumulated returns. With the knowledge learned from our constructed large-scale FinKG, the proposed approach outputs meaningful embeddings of the events. The results prove that our approach is practical for real-world quantitative investment applications.

ACKNOWLEDGMENTS

The work is supported by the National Key R&D Program of China (2018AAA0100704) and the China Postdoctoral Science Foundation

(2019M651499). Ying Zhang is supported by ARC DP180103096 and FT170100128. Xiaoyang Wang is supported by NSFC (61802345). The authors want to thank Emoney Inc and Seek Data Inc[9] for providing dataset and supports on the quantitative trading experiments that made this study possible.

REFERENCES

[1] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. In *Noise reduction in speech processing*. Springer, 1–4.

[2] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. 1247–1250.

[3] Wesley S Chan. 2003. Stock price reaction to news and no-news: drift and reversal after headlines. *Journal of Financial Economics* 70, 2 (2003), 223–260.

[4] Dawei Cheng, Ye Liu, Zhibin Niu, and Liqing Zhang. 2018. Modeling similarities among multi-dimensional financial time series. *IEEE Access* 6 (2018), 43404–43413.

[5] Dawei Cheng, Yi Tu, Zhibin Niu, and Liqing Zhang. 2018. Learning Temporal Relationships Between Financial Signals. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2641–2645.

[6] Edouard Delasalles, Ali Ziat, Ludovic Denoyer, and Patrick Gallinari. 2019. Spatio-temporal neural networks for space-time data modeling and relation discovery. *Knowledge and Information Systems* 61, 3 (2019), 1241–1267.

[7] Shumin Deng, Ningyu Zhang, Wen Zhang, Jiaoyan Chen, Jeff Z Pan, and Huajun Chen. 2019. Knowledge-driven stock trend prediction and explanation via temporal convolutional network. In *Companion Proceedings of The 2019 World Wide Web Conference*. 678–685.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 4171–4186.

[9] Xiao Ding, Kuo Liao, Ting Liu, Zhongyang Li, and Junwen Duan. 2019. Event Representation Learning Enhanced with External Commonsense Knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 4896–4905.

[10] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2015. Deep learning for event-driven stock prediction. In *Twenty-fourth international joint conference on artificial intelligence*.

[11] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2016. Knowledge-driven event embedding for stock prediction. In *Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers*. 2133–2142.

[12] Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. 2016. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1555–1564.

[13] Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S Weld. 2008. Open information extraction from the web. *Commun. ACM* 51, 12 (2008), 68–74.

[14] Eugene F Fama. 1965. The behavior of stock-market prices. *The journal of Business* 38, 1 (1965), 34–105.

[15] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. 2019. Temporal relational ranking for stock prediction. *ACM Transactions on Information Systems (TOIS)* 37, 2 (2019), 1–30.

[16] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 855–864.

[17] Florian Holzschuher and René Peinl. 2013. Performance of graph query languages: comparison of cypher, gremlin and native access in Neo4j. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*. 195–204.

[18] Kewei Hou. 2007. Industry information diffusion and the lead-lag effect in stock returns. *The Review of Financial Studies* 20, 4 (2007), 1113–1138.

[19] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[20] Shimon Kogan, Dimitry Levin, Bryan R Routledge, Jacob S Sagi, and Noah A Smith. 2009. Predicting risk from financial reports with regression. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 272–280.

[21] Rafal Kuc and Marek Rogozinski. 2013. *Elasticsearch server*. Packt Publishing Ltd.

[22] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*. 1188–1196.

[23] David Leinweber and Jacob Sisk. 2011. Event-driven trading and the âĂIJnew newsâĂİ. *The Journal of Portfolio Management* 38, 1 (2011), 110–124.

[24] Qing Li, Jinghua Tan, Jun Wang, and HsinChun Chen. 2020. A Multimodal Event-driven LSTM Model for Stock Prediction Using Online News. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[25] Ying Li, Ting Jin, Meng Xi, Shengpeng Liu, and Zhiling Luo. 2018. Massive Text Mining for Abnormal Market Trend Detection. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 4135–4141.

[26] Zhongguo Li and Maosong Sun. 2009. Punctuation as implicit annotations for Chinese word segmentation. *Computational Linguistics* 35, 4 (2009), 505–512.

[27] Zhige Li, Derek Yang, Li Zhao, Jiang Bian, Tao Qin, and Tie-Yan Liu. 2019. Individualized indicator for all: Stock-wise technical indicator optimization with stock embedding. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 894–902.

[28] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2124–2133.

[29] Andrew W Lo and A Craig MacKinlay. 1990. When are contrarian profits due to stock market overreaction? *The review of financial studies* 3, 2 (1990), 175–205.

[30] Ronny Luss and Alexandre dâĂŹAspremont. 2015. Predicting abnormal returns from news using text classification. *Quantitative Finance* 15, 6 (2015), 999–1012.

[31] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[32] Tiago Macedo and Fred Oliveira. 2011. *Redis Cookbook: Practical Techniques for Fast Data Manipulation*. " O'Reilly Media, Inc.".

[33] Daniel Myers and James W McGuffee. 2015. Choosing scrapy. *Journal of Computing Sciences in Colleges* 31, 1 (2015), 83–89.

[34] Shirui Pan, Jia Wu, Xingquan Zhu, Chengqi Zhang, and Yang Wang. 2016. Tri-party deep network representation. *Network* 11, 9 (2016), 12.

[35] Swarnadeep Saha et al. 2018. Open information extraction from conjunctive sentences. In *Proceedings of the 27th International Conference on Computational Linguistics*. 2288–2299.

[36] Swarnadeep Saha, Harinder Pal, et al. 2017. Bootstrapping for numerical open ie. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 317–323.

[37] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3027–3035.

[38] J Shaheen. 2017. Apache Kafka: Real Time Implementation with Kafka Architecture Review. *International Journal Of Advanced Science And Technology* 109 (2017), 35–42.

[39] 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)*. 1139–1145.

[40] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[41] Paul C Tetlock, Maytal Saar-Tsechansky, and Sofus Macskassy. 2008. More than words: Quantifying language to measure firms' fundamentals. *The Journal of Finance* 63, 3 (2008), 1437–1467.

[42] Jingyuan Wang, Yang Zhang, Ke Tang, Junjie Wu, and Zhang Xiong. 2019. AlphaStock: A Buying-Winners-and-Selling-Losers Investment Strategy using Interpretable Deep Reinforcement Attention Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1900–1908.

[43] 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*. 873–883.

[44] Yang Yang, ZHOU Deyu, Yulan He, and Meng Zhang. 2019. Interpretable Relevant Emotion Ranking with Event-Driven Attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 177–187.

[45] Wenbo Zhang, Xiao Ding, and Ting Liu. 2018. Learning target-dependent sentence representations for chinese event detection. In *China Conference on Information Retrieval*. Springer, 251–262.

[46] Sendong Zhao, Quan Wang, Sean Massung, Bing Qin, Ting Liu, Bin Wang, and ChengXiang Zhai. 2017. Constructing and embedding abstract event causality networks from text snippets. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. 335–344.

[47] Ali Ziat, Edouard Delasalles, Ludovic Denoyer, and Patrick Gallinari. 2017. Spatio-temporal neural networks for space-time series forecasting and relations discovery. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 705–714.

[9] www.emoney.cn, www.seek-data.com