Using External Knowledge for Financial Event Prediction Based on Graph Neural Networks

Yiying Yang[1,3], Zhongyu Wei[1,*], Qin Chen[1], Libo Wu[2]
[1]School of Data Science, Fudan University, Shanghai, China
[2]School of Economics, Fudan University, Shanghai, China
[3]AI Department, Pingan Life Insurance, Shenzhen, China
{yangyy17,zywei,qin_chen,wulibo}@fudan.edu.cn

ABSTRACT

This paper focuses on a novel financial event prediction task that takes a historical event chain as input and predicts what event will happen next. We introduce financial news as supplementary information to solve problems of multiple interpretations of same financial event. Besides, a gated graph neural network based approach is utilized to capture complicated relationships between event graphs for better event prediction. For the evaluation, we build a new dataset consisting of financial events for thousands of Chinese listed companies from 2013 to 2017. Experimental results show the effectiveness of our proposed model.

KEYWORDS

Event Representation; Graph Neural Network; Event Prediction

ACM Reference Format:
Yiying Yang[1,3], Zhongyu Wei[1,*], Qin Chen[1], Libo Wu[2]. 2019. Using External Knowledge for Financial Event Prediction Based on Graph Neural Networks. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM'19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3357384.3358156

1 INTRODUCTION

In recent years, machine learning and artificial intelligence have had fruitful applications in the financial domain, such as risk management, marketing, investment prediction and fraud prevention [2, 4]. By using multiple resources of information, including news, company announcements, social media data, etc., intelligent systems could provide valuable advice to investors and find anomalies to support government for market supervision.

Financial events are one of the most commonly used information, that allow us to predict the market trends and provide us with justification of the predictions. Despite its importance, financial event prediction has not alerted enough attention from the research community. In this paper, we propose a novel task of predicting the next financial event for a target company based on the historical event chain. A running example can be seen in Figure 1. Suppose the market has witnessed a series of events for a target company, *published a financial report (A), found revenue growth (B), and then held a shareholders meeting (C)*, what would happen next?

In general, the occurrence of financial events is not isolated but interdependent and traceable. Because event *B* indicates good operating condition of the company, it is more likely a positive event would follow, in other words, the happening probabilities of *dividends (D)* and *acquisition (E)* are higher than *bankruptcy (F), layoffs* and *suspension*. However, the task of financial event prediction is challenging in two perspectives.

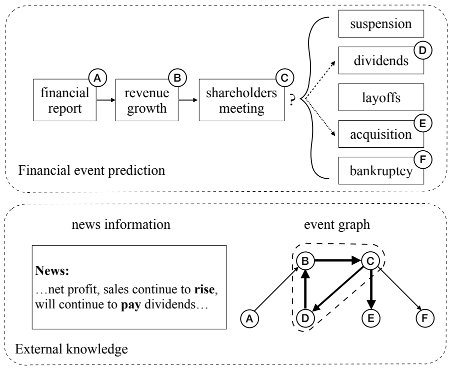

Figure 1: An example of financial event chain with external knowledge including news and event graph.

First, events themselves contain limited information, and they can be interpreted in different ways. External knowledge is useful to alleviate the difficulty. We catch the news report corresponding to the event of *shareholders meeting (C)*, which is *net profit, sales continue to rise, will continue to pay dividends….* With this backup, the probability of the next event being *D (dividends)* will increase. Attention mechanism is introduced to further incorporate the related news information.

Another challenge is the complicated relationships among financial events. To simulate these relationships, we construct an event graph (refer to section3.2, the node represents event, and the edge represents the relationship between two events) based on the historical interactions of financial events. With the information of this event graph, events B, C and D constitute a strongly interrelated component. This indicates that D has higher probability compared to E. Graph neural network [5] is utilized for event graph modeling.

We propose a novel task for financial event prediction, utilize external knowledge including news and event graph for better prediction and the experimental results show the superiority of our proposed graph-based model.

Table 1: The number of occurrences and their proportion of the top 10 event types in the dataset.

Event type	Number	Percent
Restructuring suspension	93,455	6.90%
Disclosure suspension	83,999	6.20%
Trading irregularity	53,723	3.97%
Notice of shareholders meeting	46,804	3.46%
Block trading	42,094	3.11%
Dividend announcement	27,573	2.04%
Equity pledge announcement	25,296	1.87%
Releasing shares announcement	16,819	1.24%
Disclosure of annually report	14,460	1.07%
Shareholders reduce shareholding	14,295	1.06%

2 DATASET

2.1 Event

We build a new dataset using information of Chinese listed companies. Specifically, the information is all events that occurred in a listed company which are exposed to the public via announcement. The dataset includes **1,271,130** events of **3,556** companies happened during 2013 and 2017. Every company has 357 events on average and average interval for two consecutive events is 6.52 days. A single event consists of three parts, that is, *subject (the company code), time (year-month-day) and event type.*

The dataset has 360 event types. The classification criteria come from Wind Information. The top 10 of 360 event types and their data distribution are listed in Table 1. Events related to suspensions, shareholder meetings and announcement occur more frequently.

2.2 News

We also collected financial news from 2013 to 2017 from Wind Information, totaling **1,696,468**. A single news consists of three parts, that is, *subject (one or more company codes related to the news), time (year-month-day) and news (headlines).*

From each news, we get (company, verb) pairs, in which the company is the news subject, and the verbs are obtained from the news headlines through part-of-speech tagging. Each piece of news contains 12.54 words on average, and 1.98 verbs are detected with 3.39 companies involved.

Due to the large number of companies and verbs, the relationships between verbs and companies are sparse. In order to alleviate the sparsity, we cluster companies into the industry according to SWS Industry Classification Benchmark .With **28** industries and **30,042** verbs, we form 200,864 (industry, verb) pairs in total. We build a news graph based on them. Each pair of (industry, verb) represents a node and if two pairs occurred in a single news report, we link an edge between them. Events and news are matched by the same date of the same company, which then mapped to industry. On average, a financial event is related to **1.78** news nodes.

3 PROPOSED APPROACH

For a certain subject (company), given a chain of events e_1, e_2, ..., e_n, and candidate event list c_1,...,c_k, our task is to choose the most likely next event from the candidate list. The overall structure of our model is shown in Figure 2. It consists of three components, namely *event representation*, *graph neural network* and *event prediction.*

3.1 Event Representation

This module obtains a representation for an event by combining initial event embedding and related news representation.

Event Embedding Initialization The event embedding of e_i is represented as $h(e_i) \in \mathbb{R}^d$ (d is the dimension of event embeddings) in the model, which is randomly initialized. It should be noted that candidate events are also selected from these event types, so the embedding method is the same.

Related News Embedding Learning A news node is a pair of (industry, verb). For industry part, since it is a categorical variable of 28 values, we turn it into an one-hot vector. Then, we use the pre-trained BERT [3] model to get verb word embeddings. By concatenating industry vector and verb vector, we get the node representation $v_i \in \mathbb{R}^m$ (m is the dimension of news embeddings).

Since an event may obtain multiple corresponding news nodes, we need to combine the representations of related nodes. We utilize graph attention [10] here. In practice, the final representation h_v of related news is the weighted sum of all related news nodes, which is computed as below:

$$h_v = \sum_{i=1}^{N} \alpha_{R_i} v_i \tag{1}$$

$$\alpha_{R_i} = \frac{exp(tanh(W_v v_i))}{\sum_{j=1}^{N} exp(tanh(W_v v_j))} \tag{2}$$

where $v_i \in \mathbb{R}^m$ is the news embeddings, R_i is the specific event, W_v is the model parameter, and α_{R_i} can be viewed as the weight of node v_i in all concerned nodes.

Event Representation Updating Given the initial event embedding h_e and the representation of its related news h_v, there are several ways to get the joint representation of the target event representation $h_e^{(0)}$ by a mapping function $h_e^{(0)} = f(h_e, h_v)$. We experiment three widely used methods: 1) Average: Use the mean value of vectors. 2) *Nonlinear Transformation*:
$h_e^{(0)} = tanh(W_e \cdot h_e + W_p \cdot h_p + b)$, where W_e,W_p are model parameters. 3) Concatenation: $h_e^{(0)} = h_e \oplus h_p$, concatenate the two vectors as the representation of the whole event, where $h_e^{(0)} \in \mathbb{R}^{d+m}$.

3.2 Gated Graph Neural Network

This module updates the event representation based on events graph using gated Graph Neural Network (GGNN)[6].

Event Graph Construction The event graph is constructed based on the event chains. It is formally denoted as $G = \{E, L\}$, where $E = e_1, e_2, ..., e_n$ is the event node set, and $L = l_1, l_2, ..., l_n$ is the edge set. We count all the adjacent event pairs in the training event chains, and regard each pair as an edge l_i in L. Each l_i is a directed edge $e_i \rightarrow e_j$ along with a weight w, which can be computed by:

$$w(e_j|e_i) = \frac{count(e_i, e_j)}{\sum_k count(e_i, e_k)} \tag{3}$$

where $count(e_i, e_j)$ means the frequency of the (e_i, e_j) event pair in the training event chains. The constructed event graph G has 360 nodes, and 12,079 directed and weighted edge.

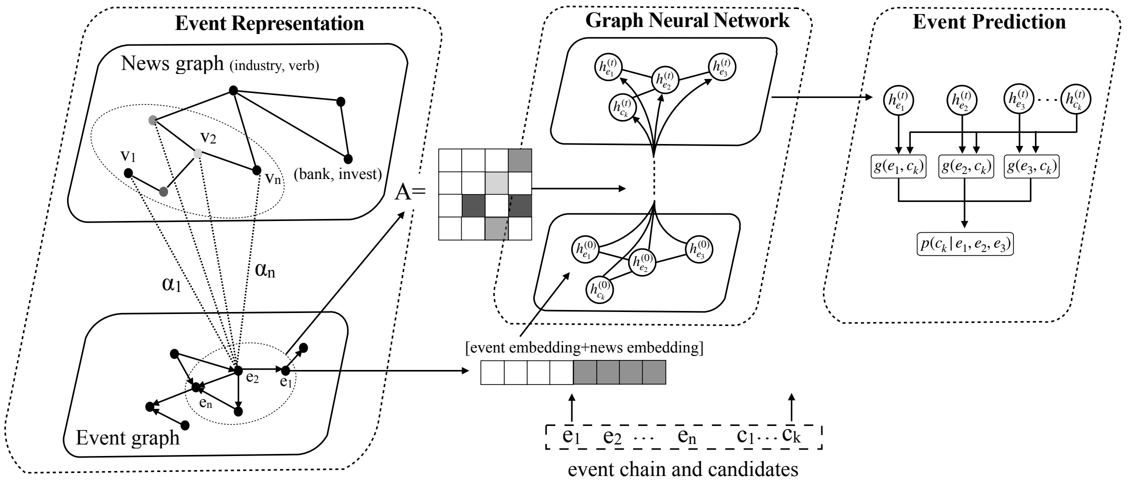

Figure 2: The overall structure of our model.

GGNN Unit By concatenating all context events and candidates, we obtain the event matrix $h^{(0)} = [h^{(0)}_{e_1}, ..., h^{(0)}_{e_n}, h^{(0)}_{c_1}, ..., h^{(0)}_{c_k}]$. The two inputs to GGNN are the event matrix $h^{(0)}$ and the adjacency matrix **A**. $\mathbf{A} \in \mathbb{R}^{(n+k)\times(n+k)}$ is the subgraph adjacency matrix extracted from the event graph, where n is the length of event chain and k is the number of candidates:

$$\mathbf{A}[i, j] = \begin{cases} w(e_j|e_i), & if\ e_i \rightarrow e_j \in L \\ 0, & others \end{cases} \tag{4}$$

The adjacency matrix **A** determines how nodes in the subgraph interact with each other. The basic recurrence of GGNN is:

$$a^{(t)} = \mathbf{A}^\top h^{(t-1)} + b \tag{5}$$

$$z^{(t)} = \sigma(W_z a^{(t)} + U_z h^{(t-1)}) \tag{6}$$

$$r^{(t)} = \sigma(W_r a^{(t)} + U_r h^{(t-1)}) \tag{7}$$

$$c^{(t)} = tanh(W_c a^{(t)} + U_c(r^{(t)} \odot h^{(t-1)}) \tag{8}$$

$$h^{(t)} = (1 - z^{(t)}) \odot h^{(t-1)} + z^{(t)} \odot c^{(t)} \tag{9}$$

Equation 5 is a step of transferring information between different nodes of the graph by the directed adjacency matrix **A**. The rest is a GRU-like operation that updates each hidden state from the previous time step. $z^{(t)}$ and $r^{(t)}$ are update gate and reset gate, σ is the logical sigmoid function, and $\odot$ is element-wise multiplication. We unroll the above loop propagation to a fixed number of steps, t means the t-th step of GGNN. The output $h^{(t)} \in \mathbb{R}^{(n+k)\times(d+m)}$ can be viewed as the updated event representation.

3.3 Event Prediction

This module calculates scores for each candidate event and output the next event c_j.

Score Calculation The output $h^{(t)}$ of GGNN are the updated representation of context events $h^{(t)}_{e_1}$,$h^{(t)}_{e_2}$,...$h^{(t)}_{e_n}$ and candidate events $h^{(t)}_{c_1}$,$h^{(t)}_{c_2}$,...,$h^{(t)}_{c_k}$.

Given a pair of events $h^{(t)}_{e_i}$ and $h^{(t)}_{c_j}$, the relation score is calculated as the euclidean distance of two vectors:

$$s_{ij} = -euclid(h^{(t)}_{e_i}, h^{(t)}_{c_j}) = -\|h^{(t)}_{e_i} - h^{(t)}_{c_j}\| \tag{10}$$

The relevance of the candidate event to the entire event chain can be understood as the sum of the relation score with each event. We choose the subsequent event c_j by choosing $j^* = \arg\max_j s_j$, where $s_j = p(c_j|e_1, e_2, e_n) = \frac{1}{n}\sum_{i=1}^{n} s_{ij}$.

Loss Function Given a set of event chains, each with a number of candidate events, one of which is a real subsequent event, our training objective is to maximize the score of real event while minimize the score of other candidate events. We use margin loss as the loss function:

$$L(\Theta) = \sum_{i=1}^{n}\sum_{j=1}^{k}(max(0, m - s_{iy} + s_{ik})) + \frac{\lambda}{2}\|\Theta\|^2 \tag{11}$$

where s_{ij} is the relation score between context event chain e_i and candidate event c_j, y is the index of real subsequent event, m is the loss function parameter, λ is the L2 regularization parameter and model parameters Θ are optimized by using RMSprop.

4 EXPERIMENTS

4.1 Experimental Setup

The dataset is split into three parts. Events from 2013 to 2016 are training set, the first half of 2017 are development set, and the second half of 2017 are testing set. For each company, we get a time-ordered event list of hundreds of events. By selecting certain length of consecutive events chain from the list as input events, the subsequent one event and the other four selected events are used as candidate events. It is designed as a multi-choice cloze task, because in the real world, we usually also construct a candidate pool of some most controversial events based on prior knowledge, and then choose from them. The amounts of instances for training, development and testing are 1,070,934 : 124,977 : 75,219 respectively, and the ratio is around 8:1:1.

There are several important hyper-parameters in our models, and we tune their values using the development dataset. We set the initial learning rate to 0.0001. The size of node embedding is set to 128, the size of hidden units in hidden layer to 128. GGNN recurrent times is 2, training batch size is 1000, m is 0.015 and λ is 0.00001. In order to avoid over-fitting, dropout is used in the training process with a ratio of 0.2.

Table 2: Overall performance for financial event prediction of different models. Four indicators are Accuracy, Precision, Recall, and F1-score.

Methods	Acc.(%)	Pre.(%)	Re.(%)	F1(%)
PMI[1]	59.79	88.16	57.43	66.44
DeepWalk [8]	78.93	86.26	79.99	81.24
Word2Vec [7]	80.35	86.11	81.42	82.12
LSTM [9]	86.75	90.97	87.93	88.47
GGNN	88.12	90.65	88.73	89.15
GGNN+News (mean)	89.99	92.25	90.73	91.08
GGNN+News (attention)	**90.55**	92.57	91.13	**91.53**

4.2 Compared Models

We evaluate the effectiveness of our proposed model with several state-of-the-art baseline models:
PMI[1]: is the co-occurrence-based model that calculates event pairs relations based on Pairwise Mutual Information.
Word2Vec[7]: is the method for learning event embeddings by taking event chains as sentences.
DeepWalk[8]: is the unsupervised model that uses random walks to learn event graph embeddings.
LSTM [9]: is the model that integrates event order information and pairwise event relations together to generate event representations.
GGNN: is our GGNN model mentioned in Section 3.
GGNN+News (mean): is the GGNN model with external news information. News node embeddings leveraged with mean pooling.
GGNN+News (attention): is the GGNN model with external news information. News node embeddings leveraged with graph attention pooling.

4.3 Experimental Results

The overall experimental results are shown in Table 2. In general, our *GGNN* model is better than all the baseline models. This indicates the effectiveness of modeling the interaction of financial events through event graph.

The performance of *PMI* is significantly worse than other models that indicates it is not enough to merely consider the statistical pair-wise relationships for financial event prediction.

The performance of *DeepWalk* and *Word2Vec* are significantly improved compared to *PMI* since they learn dense representation for financial event.

The performance of models with news information are better than the model without it. This indicates that news as external information can indeed make contribution for event prediction. Besides, graph attention pooling is better than mean pooling.

4.4 Further Analysis

We conduct two additional experiments to further validate the effectiveness of our proposed methods.
Effectiveness of different methods for joint event representation learning Given the event embedding h_e and related news representation h_v, there are several ways to get the joint representation $h_e^{(0)}$, as introduced in Section 3.1. The experimental results are shown in Table 3. Compared to *Average* and *Nonlinear Transformation* function, *Concatenation* has the best experimental results.

Table 3: Three methods to get the joint event representation.

Methods	Accuracy (%)
Average	87.75
Nonlinear	89.43
Concatenation	**90.55**

Table 4: Influence of the number of news nodes used.

Number of Nodes (Accuracy%)	2	4	6
GGNN+News (mean)	89.40	89.53	89.99
GGNN+News (attention)	90.36	90.45	**90.55**

Influence of the number of news nodes for representation learning In the above experiments, we considered 2 related news nodes (zero padding if less than 2) for each event. After data analysis, we found that 75.00% of the events correspond to the number of news nodes less than or equal to 2. 4 nodes can cover 96.62% of data, and 6 nodes can cover 99.80% of data. We explore to see how the number of news nodes used influences the performance. Results can be seen in Table 4. In general, the more news nodes are used, the more external information is introduced, and better experimental results can be obtained.

5 CONCLUSIONS

In this paper, we propose a novel task of financial event prediction and build a dataset to support the research. We propose to use external knowledge for better prediction based on a gated Graph Neural Network. Experimental results show the superiority of our proposed model compared to other baselines.

6 ACKNOWLEDGMENTS

This work is partially supported by National Natural Science Foundation of China (No. 61751201, No. 61702106) and Shanghai Science and Technology Commission (No. 17JC1420200, No. 17YF1427600 and No. 16JC1420401).

REFERENCES

[1] Nathanael Chambers and Daniel Jurafsky. 2008. Unsupervised Learning of Narrative Event Chains. In *ACL*. 789–797.

[2] Yingmei Chen, Zhongyu Wei, and Xuanjing Huang. 2018. Incorporating Corporation Relationship via Graph Convolutional Neural Networks for Stock Price Prediction. In *CIKM*. ACM, 1655–1658.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2019), 4171–4186.

[4] Fahimeh Ghobadi and Mohsen Rohani. 2016. Cost sensitive modeling of credit card fraud using neural network strategy. In *ICSPIS*. IEEE, 1–5.

[5] Marco Gori, Gabriele Monfardini, and Franco Scarselli. 2005. A new model for learning in graph domains. In *IJCNN*. IEEE, 729–734.

[6] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2015. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493* (2015).

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

[8] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *KDD*. ACM, 701–710.

[9] Zhongqing Wang, Yue Zhang, and Ching-Yun Chang. 2017. Integrating Order Information and Event Relation for Script Event Prediction. In *EMNLP*. 57–67.

[10] Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense Knowledge Aware Conversation Generation with Graph Attention. In *IJCAI*. 4623–4629.