Article

TDJEE: A Document-Level Joint Model for Financial Event Extraction

Peng Wang [1,2,*], Zhenkai Deng [2] and Ruilong Cui [1]

[1] School of Computer Science and Engineering, Southeast University, Nanjing 211189, China; xiaocuikindle@gmail.com
[2] School of Cyber Science and Engineering, Southeast University, Nanjing 211189, China; zhenkaideng@seu.edu.cn
* Correspondence: pwang@seu.edu.cn

Abstract: Extracting financial events from numerous financial announcements is very important for investors to make right decisions. However, it is still challenging that event arguments always scatter in multiple sentences in a financial announcement, while most existing event extraction models only work in sentence-level scenarios. To address this problem, this paper proposes a relation-aware Transformer-based Document-level Joint Event Extraction model (TDJEE), which encodes relations between words into the context and leverages modified Transformer to capture document-level information to fill event arguments. Meanwhile, the absence of labeled data in financial domain could lead models be unstable in extraction results, which is known as the cold start problem. Furthermore, a Fonduer-based knowledge base combined with the distant supervision method is proposed to simplify the event labeling and provide high quality labeled training corpus for model training and evaluating. Experimental results on real-world Chinese financial announcement show that, compared with other models, TDJEE achieves competitive results and can effectively extract event arguments across multiple sentences.

Keywords: event extraction; financial event; event knowledge base; distant supervision

Citation: Wang, P.; Deng, Z.; Cui, R. TDJEE: A Document-Level Joint Model for Financial Event Extraction. *Electronics* **2021**, *10*, 824. https://doi.org/10.3390/electronics10070824

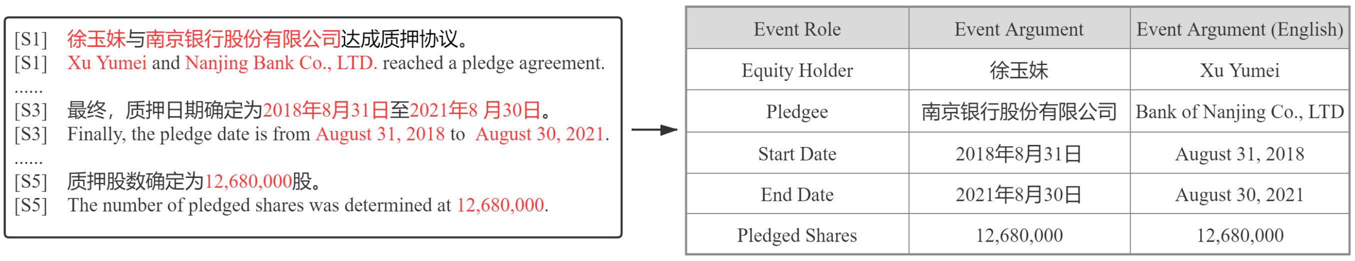
Academic Editor: Paolo Torroni

Received: 22 February 2021
Accepted: 28 March 2021
Published: 31 March 2021

1. Introduction

Event extraction [1], which aims to identify event arguments which are primary roles composing an event and fill them into corresponding pre-defined event types, is a challenging task in NLP (Natural Language Process). Naturally, event extraction can be divided into two sub-tasks: event argument extraction and event type detection. It has a wide range of applications in the fields such as intelligent question answering, information retrieval [2], automatic summarization [3], recommendation [4], etc.

In recent years, with the increase of financial announcements, it is a labor-intensive task to analyze large amounts of financial announcements manually. Therefore, extracting structured events from financial announcements automatically is very critical for investors to make decisions. However, it is still a challenge to solve the problem of event arguments scatter in different sentences in financial domain. To clearly illustrate such challenge, here is an example shown in Figure 1. Event roles Equity Holder and Pledgee are in Sentence 1, but Start Date and End Date are in Sentence 3. Furthermore, Pledged Shares is in Sentence 5. However, most event extraction models [5–7] extract arguments within a sentence, such as the ACE 2005 dataset (https://www.ldc.upenn.edu/collaborations/past-projects/ace, 15 January 2021), a popular event extraction dataset. It is obvious that such models lack the ability to extract event arguments across multiple sentences. Researchers have paid attention at the document-level to challenge the arguments-scattering problem. Huang et al. [8] point out that the pipeline architecture with three-stage task can extract document-level context information, but error information propagating will be an obstacle to generate correct results. Zheng et al. [9] use an end-to-end model to identify event arguments at the sentence level first. Then, binary classifiers are used to determine the event type, and event arguments are transformed into a directed acyclic graph. However, it is still hard to solve the problem that event arguments are far apart.



[S1] 徐玉妹与南京银行股份有限公司达成质押协议。
[S1] Xu Yumei and Nanjing Bank Co., LTD. reached a pledge agreement.
......
[S3] 最终，质押日期确定为2018年8月31日至2021年8 月30日。
[S3] Finally, the pledge date is from August 31, 2018 to August 30, 2021.
......
[S5] 质押股数确定为12,680,000股。
[S5] The number of pledged shares was determined at 12,680,000.

Event Role	Event Argument	Event Argument (English)
Equity Holder	徐玉妹	Xu Yumei
Pledgee	南京银行股份有限公司	Bank of Nanjing Co., LTD
Start Date	2018年8月31日	August 31, 2018
End Date	2021年8月30日	August 30, 2021
Pledged Shares	12,680,000	12,680,000

Figure 1. An example of event arguments-scattering.

In this paper, we propose a relation-aware Transformer-based [10] Document-level Joint Event Extraction model (TDJEE) to address the challenge that event arguments across multiple sentences. TDJEE includes two sub-models: event argument extraction and event type detection. In event argument extraction, word representations are obtained from the BERT [11], and each representation of word contains context information from other sentences. Then, the conditional random fields (CRF) method [12] is used to identify the event arguments. Event type detection is considered as a classification task, which aims to identify the core event sentence from documents and detect the corresponding event type. In the event type detection, we utilize attention mechanism [13] to integrate word representations to get sentence representations. A relation-aware Transformer is used to further capture document-level information.

Moreover, the absence of adequate labeled data in the financial domain could lead to unstable models of low quality which is known as the cold start problem. To train and evaluate our model, we first employ Fonduer [14] to automatically build a domain-specific knowledge base, which stores structured Chinese financial events, then generate training data by distant supervision labeling. Moreover, a matcher and filter are designed to filter noise information. Then, we utilize weak supervisor method to obtain a financial specific Chinese dataset, which is about 60 times larger than ACE 2005.

Experimental results on real-world datasets demonstrate that TDJEE achieves competitive results, and can effectively extract event arguments across different sentences in financial announcements. The implementation codes and datasets used in this paper are available at https://github.com/q5s2c1/TDJEE/tree/master (20 March 2021).

The rest of the paper is organized as follows. Section 2 discusses the related work. In Section 3, we first outline the TDJEE model and give some preliminaries, then details the construction of Fonduer-based knowledge base, distant supervision based event labeling, document encoding, event argument extraction, event type detection, and the optimization techniques, respectively. The experimental results and discussion are in Section 4. Finally, Section 5 draws conclusions and further work.

2. Related Work

There are two types of event extraction methods: pipeline methods and joint models. Pipeline methods divide the event extraction task into multiple sub-tasks and process them in sequence. Its disadvantage is that the errors between different sub-tasks could be propagated. In order to solve such problems, joint models are proposed for event extraction, and it usually contains joint inference and joint modeling. Joint inference uses the ensemble learning to optimize the models through an overall objective function. Joint modeling regards the event structure as a dependency tree, and then converts the extraction task into a dependency tree structure prediction. It can recognize the trigger words and extract elements at the same time. The inference and modeling share hidden layer parameters that could avoid the performance decreasing caused by error propagation.

From the technique perspective, event extraction methods can also be divided into template- and rule-based methods, deep learning-based methods, and weak supervision-based methods. Early event extraction methods are based on template matching or regular expressions [15]. Researchers used syntactic analysis and semantic constraints to identify events in sentences. PALKA [16] uses semantic frames and phrase patterns to represent the extraction schema of events. By incorporating the semantic information of WordNet [17], PALKA can achieve results close to human beings in specific domains. However, the performance of template method highly depends on languages and has poor portability.

In recent years, most event extraction methods are based on deep learning. Compared with traditional template-based and rule-based methods, deep learning methods are more general. Chen et al. [6] propose a Dynamic Multi-Pooling Convolution Neural Network (DMCNN) to extract events, which regards event extraction as a two-stage multi-classification task: trigger classification and arguments classification. However, this method suffers from error propagation and ignores the dependency between the trigger word and the arguments. The DEEB-RNN model proposed by Zhao et al. [18] makes full use of document information in event extraction, in which text is encoded by RNN-based fusion hierarchy and attention mechanism, then the representation of text is used to judge event trigger words and types in sentences. However, as the sentence length becomes longer, the RNN-based method cannot work well. Han et al. [19] propose a network model based on the dilate gated convolutional neural network, in which the word representations and depth of the network are expanded to improve the performance. Peng et al. [20] combine CNN with gate linear mechanism to accelerate the encoding of text.

Specially, deep learning models heavily rely on a large-scale training corpus. Weak supervision method is used to reduce event labeling. Chen et al. [21] use a high-quality labeled corpus to train the classifier. Iteratively, the trained classifier is used to label the unlabeled data, and samples with high-confidence are selected to train the classifier. Finally, a high-quality labeled dataset is obtained for event extraction. Liao et al. [22] first use the self-training [23] and semi-supervised learning method to extend the labeled corpus. Then, classifiers in word and sentence granularity are trained simultaneously, and co-training [24] is utilized to extend the labeled data for event extraction.

In financial event extraction, Li et al. [25] propose a method for extracting Chinese financial events by automatically constructing the extraction rules, but this method ignores the background information about entities and relationships. Ding et al. [26] propose a joint event embedding model KGEB, which embeds the knowledge graph into the event vector representation. KGEB uses knowledge bases (such as Freebase and YAGO) to provide two types of background knowledge for event embedding: classification knowledge and relational knowledge, and has achieved the promising results in the task of stock price fluctuation prediction. Dor et al. [27] focus on digging out company-related events from text, which includes articles of describing companies on Wikipedia. Its labeling training corpus is generated through weak supervision. It utilizes a classification model to detect the company-related event sentences. For reports in the financial domain, Ding et al. [28] first define the basic event framework based on expert knowledge, then combine with traditional natural language processing tools such as rules, part-of-speech tagging, and named entity recognition for trigger word recognition and event element recognition, and finally transform the events in the financial reports into a structured form. DCFEE proposed by Yang et al. [29] realizes the extraction of equity freezing, pledge, repurchase, and increase or decrease of holdings in financial announcements. The DCFEE model expands the training corpus by using distant supervision, and divides the event extraction into two-stage sub-tasks. In the first stage, the sentence-level event extraction task is performed. The Bi-LSTM (Bi-direction Long Short-Term Memory) [30] and CRF (Conditional Random Fields) [12] are combined to label the event elements in a sentence, and the event trigger word is detected through the dictionary. In the second stage, the identified event element and event trigger word are concatenated with the current sentence as input, and CNN is used to determine whether the current sentence is an event sentence. Doc2EDAG et al. [9] transform the event into an entity-based directed acyclic graph (EDAG), which can transform the hard slot-filling task into several sequential path-expanding sub-tasks. Moreover, Doc2EDAG designs a memory mechanism for path expanding to support the EDAG generation efficiently.

3. Model

In this section, we first provide an overview of our model, TDJEE. Then, the construction of Fonduer-based knowledge base is presented. Moreover, we introduce the improved event labeling method based on distant supervision in detailed. Subsequently, more details of TDJEE including document encoding, event argument extraction, and event type detection will be presented. Finally, a loss function of TDJEE is discussed.

3.1. TDJEE Overview

The main idea of TDJEE is to incorporate richer context information into sentence representation. Figure 2 shows an overview of the TDJEE model.

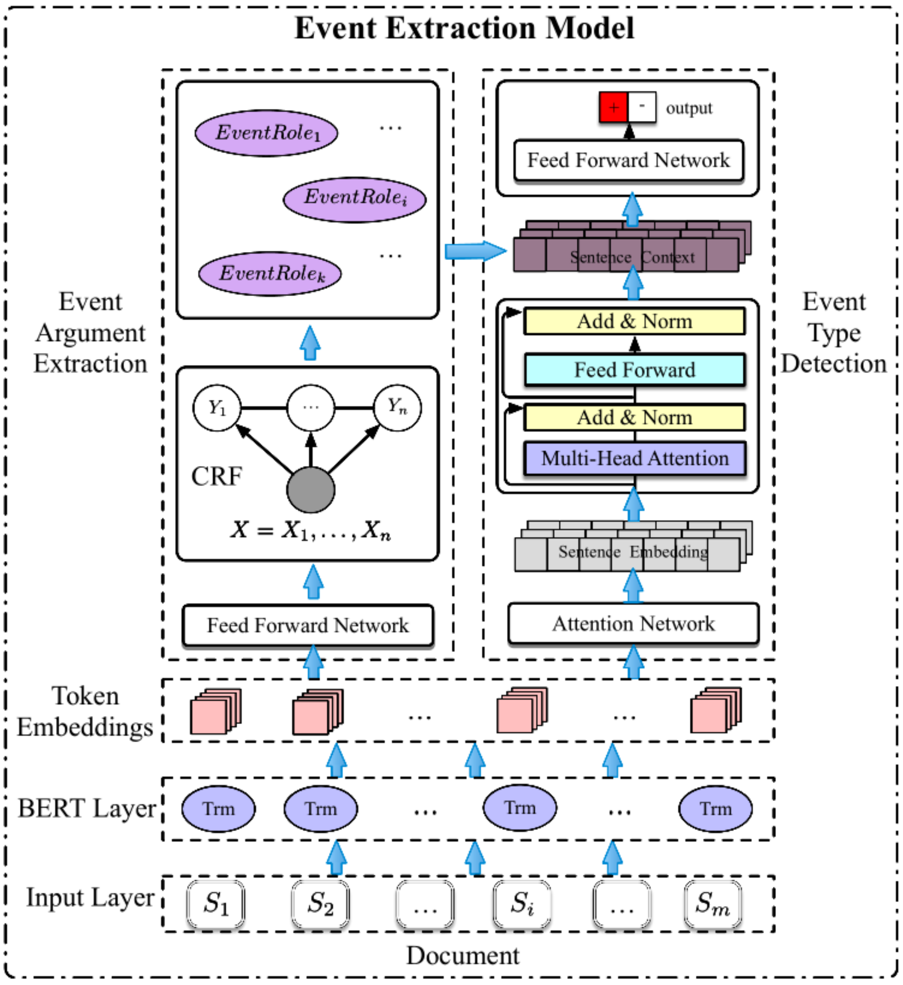


Figure 2. The overview of relation-aware Transformer-based Document-level Joint Event Extraction model (TDJEE model.

TDJEE first implements the three-layer document encoding, namely, using BERT to generate the token embeddings for financial documents. Then, the event extraction mainly includes two parts: event argument extraction and event type detection. The event argument extraction is a sequence labeling task. In the event argument extraction model, the document is first encoded by BERT, providing a semantic vector representation of the document. Then, the document representation of high dimensions is mapped to low dimensions through a feedforward neural network, and finally the event argument entities are identified by the CRF layer. Event type detection can be considered as a typical classification task. In the event type detection, a relation-aware Transformer-based encoder is used for capturing context information in multiple sentences. Furthermore, the results of event argument extraction are integrated into sentence representations, which are used in final classification.

3.2. Fonduer-Based Knowledge Base Construction

Fonduer is a machine learning-based knowledge base construction (KBC) system for richly formatted data, which uses the deep learning model to automatically capture the representation (i.e., features) [14]. Figure 3 presents the framework of construction of event knowledge base. The construction of the event knowledge base is mainly divided into three phases: data preprocessing, matching and filtering event candidate set by leveraging pre-defined rule-based matcher, and generating multimodal features for the candidate event set and training the weakly supervised classification model.

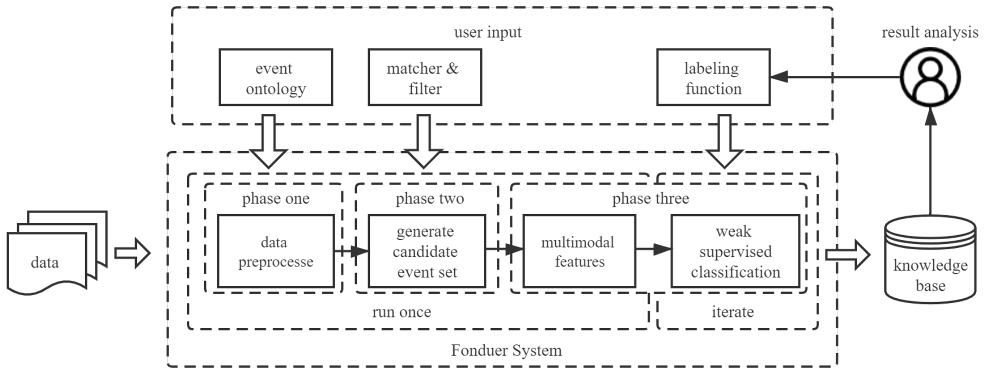


Figure 3. The process of knowledge base construction.

3.2.1. Data Preprocessing

This paper mainly studies five types of financial announcement events. Different event types use different database to store extracted structured events. The structure of the table in database is corresponding to the type of event ontology.

During data preprocessing, financial announcements of PDF format are converted into HTML fomat by using pdfminer (https://github.com/pdfminer/pdfminer.six, 15 January 2021), and all announcements are input into Fonduer. Data in PDF format are responsible for providing visual information, and data in HTML format are responsible for providing text and structured information. The Fonduer system first parses the HTML format data to identify paragraphs, sentences, tables, etc., and then converts the parsed content into the data model defined in Fonduer. The data model in Fonduer is a directed acyclic graph (DAG), which can clearly describe the hierarchical structure between contexts in the document. Each node in the graph represents a context. The root node of a DAG is the corresponding document. A document can be divided into chapters. The chapters include text, tables, and pictures. The text is divided according to paragraphs, and each paragraph is composed of different sentences. For tabular data, row, column, and header attributes can be segmented. Notice that the announcement data does not contain image contents. The advantage of using Fonduer data model is that it can extract event arguments from different contexts in the document and form a candidate event.

3.2.2. Candidate Event Set Generation

To generate the candidate event set, some matchers and filters are designed first. PersonMatcher, DateMatcher, OrganizationMatcher, and NumberMatcher are used to match named entities such as person's names, dates, organization names, and numbers appearing in sentences, respectively. These named entity identifiers are provided by the spaCy (https://github.com/howl-anderson/Chinese_models_for_SpaCy, 26 March 2021). DictionaryMatcher is used to match event arguments from a pre-defined dictionary. RegexMatchSpan is to match qualified event arguments from sentences based on regular expressions. LambdaFunctionMatcher contains the functions that defined by user to filter the input N-gram characters. Eventually multiple event arguments can be obtained.

As the combination of different entities would form a new candidate event, which could lead to an exponential increase in the number of candidate events. Therefore, there are a large number of negative examples in the generated candidate events. The filter is essentially a set of rule which mining by user from large amounts of announcements. For any type of event, following rules can be used to filter the candidate event:

- Rule 1: If multiple event arguments have same value, the remaining can be filter out.
- Rule 2: If the value of any key event arguments is empty, it can be filtered out.
- Rule 3: If an event argument is in the table structure, then other event arguments should also be in the same row of the same table, otherwise it can be filtered out.

3.2.3. Multimodal Feature and Weak Supervised Classification

In order to effectively improve the performance of the classification model, Fonduer generates a multimodal feature for each candidate event, including text features, structural features, table features, and vision features. The text feature is the concatenation of the part of speech and the named entity tag corresponding to the event argument. The structural feature represents the location of the event argument in the original HTML document, the corresponding tags, attributes, and other information. Table features represent information such as the position and attribute distance of event arguments in the table of financial announcements. Vision features are mainly used to indicate whether different event arguments have similar visual features. The multimodal features of the event are concatenated from the above features and used for the weakly supervised classification task.

As shown in Figure 4, in weakly supervised classification, the user first provides weakly supervised labels for the data by labeling functions and then uses the classification model to learn potential relationships from the manually labeled data, uses test data to verify the performance of model, and finally the user adjusts the marking function by observing the performance of model. Such process is iterative. A classification model with better performance will eventually be trained. After using this model to label all candidate events, a structured event knowledge base can be obtained. Here, the marking function is a data programming paradigm, which is to label candidate events in a programmatic way, namely, mark the candidate event as a positive/negative example or skip labeling.

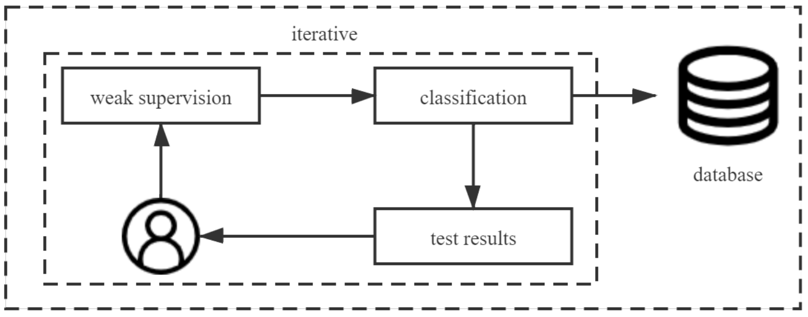


Figure 4. The process of event labeling.

3.2.4. Event Ontology

Table 1 shows the event ontology for five types of announcements: Equity Repurchase (ER), Equity Freeze (EF), Equity Underweight (EU), Equity Overweight (EO), and Equity Pledge (EP). The event ontology defines the event role information that needed to form a structured event, including two types of key roles and non-key roles. The key role is the necessary information to form an event, such as the Company Name and Repruchased Shares in ER. Without any key role, a complete structured event can not be formed.

Table 1. Event ontology of ER, EF, EU, EO, and EP.

Event Type	Event Role	Description	Key Role
EquityRepurchase (ER)	CompanyName	the name of company that repurchasing equity	True
	RepurchasedShares	number of repurchased shares	True
	HighestTradingPrice	the highest trading price	False
	LowestTradingPrice	the lowest trading price	False
	ClosingDate	the end date of repurchasing	False
	RepurchaseAmount	total number of repurchased shares	False
EquityFreeze (EF)	EquityHolder	the name of holder whose equity is frozen	True
	FrozeShares	number of frozen shares	True
	LegalInstitution	legal institution	True
	StartDate	start date of equity frozen	False
	EndDate	end date of equity frozen	False
	UnfrozeDate	date of equity unfrozen	False
	TotalHoldingShares	total holding shares	False
	TotalHoldingRatio	total holding ratio	False
EquityUnderweight (EU)	EquityHolder	the name of holder that underweighting equity	True
	TradedShares	number of traded shares	True
	StartDate	start date of equity underweight	False
	EndDate	end date of equity underweight	False
	AveragePrice	average price of trading	False
	LaterHoldingShares	number of shares after underweighting	False
EquityOverweight (EO)	EquityHolder	the name of holder that overweighting equity	True
	TradedShares	number of traded shares	True
	StartDate	start date of equity overweight	False
	EndDate	end date of equity overweight	False
	AveragePrice	average price of trading	False
	LaterHoldingShares	number of shares after overweighting	False
EquityPledge (EP)	EquityHolder	the name of holder that pledging equity	True
	PledgedShares	number of shares that pledged	True
	Pledgee	target organization that pledged	True
	StartDate	start date of equity pledged	False
	EndDate	end date of equity pledged	False
	ReleasedDate	released date of shares	False
	TotalPledgedShares	total pledged shares	False
	TotalHoldingShares	total holding shares	False
	TotalHoldingRatio	total holding ratio	False

3.3. Distant Supervision Based Event Labeling

Motivated by distant supervision data labeling method based on the knowledge base [31], we first construct Fonduer-based financial event knowledge base, and then align the knowledge base to the documents (i.e., announcements) with distant supervision method to obtain a large amount of labeled data.

Using the distant supervision method to align the knowledge base with unstructured text can automatically build a large amount of training data, reducing the dependence on manual annotation data. The disadvantage of this approach is that the assumption is too positive, and a lot of noise will be introduced. In event extraction, an event contains multiple event arguments, and the event arguments may scatter in multiple sentences. Therefore, the traditional distant supervision method is not suitable for the labeling of event arguments. In this paper, we improve the alignment of the knowledge base and unstructured text in the distant supervision method, and use the directional link method to match the knowledge in the knowledge base with the text, which is shown in Figure 5.

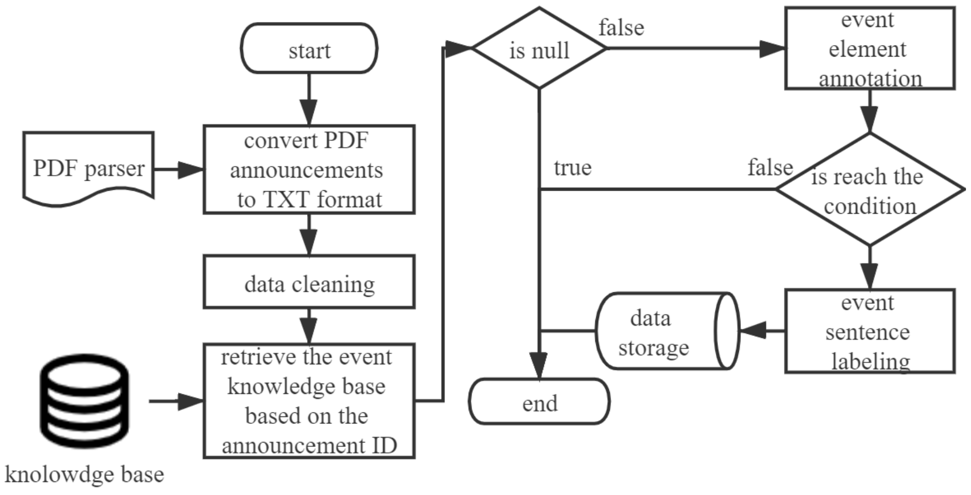


Figure 5. The process of distant supervision event labeling.

Specifically, we use a PDF parsing tool to convert the announcement document in PDF format into text format. Then, we clean the text data. Data cleaning is done to standardize the converted text, which mainly includes filtering garbled characters and segmenting sentences according to special symbols. Then, we retrieve structured events in the document from the knowledge base according to the document ID, and tag the event arguments with BIO (Beginning, Inside, Outside) format. The annotation results are filtered by two conditions: (1) an event must contain a key role (e.g., person or company) defined by event ontology and (2) the document and the event ontology have same numbers of event roles. Finally, we obtain the corpus with labeled data for five types of event: Equity Freeze (EF), Equity Repurchase (ER), Equity Underweight (EU), Equity Overweight (EO), and Equity Pledge (EP).

3.4. Document Encoding

Recently, pre-training language models have achieved good results on many natural language processing tasks. In this paper, we utilize BERT to embed the document for capturing context information beyond the sentence boundary. Before encoding the document with BERT, we first segment the document D into several sequences $\{S_1, S_2, S_3, \ldots, S_q\}$, where q is the number of sentences. The length of each sequence is less than 128. However, directly truncating sentences with a length greater than 128 causes the incompleteness of event arguments in predicting. We select the punctuation mark with the largest index value less than 128 as the cutting position of the sentence. As the number of sentences in different documents is different, in order to be able to use the BERT model to batch calculate the vector representation corresponding to each character in the sentence, we limit the maximum number of sentences allowed in the document and fix the input of each document to two-dimensional matrix of 64×128. That is, the number of sentences does not exceed 64, and the length of each sentence does not exceed 128. If the number of sentences exceeds the limit value, it will be truncated directly. Otherwise, if the number of sentences is less than the limit value, the blank rows will be filled with <PAD> characters, and combined with a mask to indicate the actual sentence length and number. Finally, the document can be represented as Equation (1):

$$\{E_{W_1}, E_{W_2}, E_{W_3}, \dots, E_{W_l}\} = BERT(S_1, S_2, S_3, \dots, S_q) \tag{1}$$

where $E_{W_i} \in \mathbb{R}^{768}$ is the representation of i-th word and l is the number of words in the document.

3.5. Event Argument Extraction

The event argument extraction can be treated as a sequence labeling task. Generally, B-LABEL is used to mark the beginning part of entity with type of LABEL, and I-LABEL to mark the middle and tail parts. However, the labels with the maximum score may be wrong when simply using BERT for sequence labeling task. It could cause that I-LABEL1 followed directly by I-LABEL2, or the label begins with I-X. To address such issues, we introduce CRF into our model to improve the performance of sequence labeling task (i.e., event argument extraction).

In this stage, BERT first provides a representation for each word. Then, the model learns the state characteristics of the sequence through a fully connected neural network to obtain a state score matrix $C \in \mathbb{R}^{m \times n}$ with the following equation:

$$C = FC(E_{W_1}, E_{W_2}, E_{W_3}, \dots, E_{W_n}) \tag{2}$$

where n is the number of words in a sentence (i.e., 128). Next, the score is input into the CRF layer. Finally, the CRF layer learns a transition score matrix $T \in \mathbb{R}^{m \times m}$ and computing all path (i.e., possible tag sequences) scores as Equations (3) and (4):

$$T = CRF(C) \tag{3}$$

$$score = \sum_{i=1}^{m} C_{i,y_i} + \sum_{i=1}^{n-1} T_{y_i,y_{i+1}} \tag{4}$$

where m is the number of label types, n is the length of the sentence (in this case, 128), C_{i,y_i} is the score of the tag y_i of the $i-th$ token in the sequence, and $T_{y_i,y_{i+1}}$ represents the score of a transition from the tag y_i to tag y_{i+1}. A softmax function is applied over scores for all path to get the probabilities $\{P_1, P_2, P_3, \dots, P_N\}$, where N is the number of paths.

3.6. Event Type Detection

Event type detection is processed in sentences. As different event arguments in the same event type may scatter in multiple sentences in event extraction, we detect the event type of sentences by a utilizing Transformer-based encoder for further capturing context information between different sentences. However, a documents may contain lots of sentences. Transformer may be hard to capture so difficult inner dependencies. Inspired by [32], we first modify the equation of self-attention in the Transformer as Equations (5)–(7):

$$e_{ij}^{(h)} = \frac{x_i W_Q^{(h)} (x_j W_K^{(h)} + r_{ij})^T}{\sqrt{d_z/H}} \tag{5}$$

$$\alpha_{ij}^{(h)} = \frac{\exp e_{ij}^{(h)}}{\sum_{k=1}^{n} \exp e_{ik}^{(h)}} \tag{6}$$

$$z_i^{(h)} = \sum_{j=1}^{n} \alpha_{ij}^{(h)} (x_j W_V^{(h)} + r_{ij}) \tag{7}$$

where the r_{ij} terms encode the known relation between two input elements: x_i and x_j. Other parameters are the same as self-attention [10]. Specifically, $W_Q^{(h)}$, $W_K^{(h)}$, and $W_V^{(h)}$ are parameter matrices that represent Query, Key, and Value in self-attention. d_z is the dimension of $z_i^{(h)}$, and H means self-attention layer employ H attention heads. Furthermore, $\alpha_{ij}^{(h)}$ is weight coefficient. $z_i^{(h)}$ is the output of self-attention. Then, we defined two undirected edges (relations): r_{same} and $r_{different}$. If event roles in S_i and S_j are able to appear in an event, S_i and S_j will be connected by edge r_{same} as shown in Figure 6. Otherwise, we add an edge $r_{different}$ to connect them. r_{same} and $r_{different}$ can be learned during training. In this way, some inner relations are exposed and encoded into context information. Besides, we call the modified Transformer above as relation-aware Transformer in this paper.

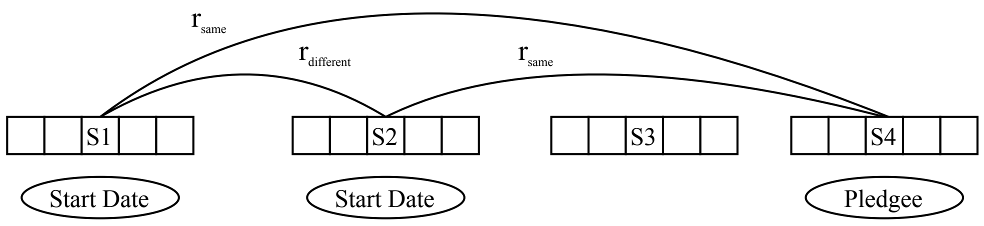


Figure 6. An example of relation-aware. Start Date and Pledgee can appear in an event like EO, but Start Date in S1 and Start Date in S2 can not appear in an event.

In event type detection, we first use the attention mechanism to integrate word representations to get sentence representations. The document can be represented by a set of sentence representations $\{E_{S_1}^{att}, E_{S_2}^{att}, E_{S_3}^{att}, \ldots, E_{S_q}^{att}\}$, where q is the number of sentences. Then, the representations refined by relation-aware Transformer are denoted as $\{\hat{E}_{S_1}, \hat{E}_{S_2}, \hat{E}_{S_3}, \ldots, \hat{E}_{S_q}\}$. In the attention network, *Query Vector* is hidden state of a word, and key_i along with $value_i$ are corresponding to word representation.

Intuitively, event arguments are primary elements of events, and may provide important information for event type detection. Therefore, we incorporate the results of event argument extraction $\{P_{S_{i-1}}, P_{S_{i-2}}, \ldots, P_{S_{i-n}}\}$ with refined representation $\hat{E}_{S_i}$. Moreover, each vector $P_{S_{i-k}}$ corresponds to an event role in the sentence, which can be learned during training. Finally, we consider event type detection as a binary classification task, and defines five discriminant models for five event types (i.e., Equity Freeze (EF), Equity Repurchase (ER), Equity Underweight (EU), Equity Overweight (EO), and Equity Pledge (EP)) for classification. Each discriminant model is composed of a simple feedforward neural network. The probability of a sentence for an event type can be calculated by Equation (8):

$$P(type = x|S_i, P_{S_{i-1}}, P_{S_{i-2}}, \ldots) = \sigma(W(\hat{E}_{S_i} + P_{S_{i-1}} + P_{S_{i-2}} + \ldots)) \tag{8}$$

where W is trainable parameter and σ is sigmoid function. The event type of a document is determined by type with the highest votes.

3.7. Optimization

Both the event type detection and the event argument extraction share the same document encoding layer. The event argument extraction calculates the loss value of the CRF layer in units of sentence, as shown in Equation (9):

$$loss_c = -\log\left(\frac{P_{\text{RealPath}}}{P_1 + \cdots + P_N}\right) \tag{9}$$

where P_{real} represents the score of ground truth. The event argument extraction loss of the document is summation of the loss of all sentences. The event type detection model uses cross entropy denoted as $loss_e$ to calculate the classification loss for each event type. Equation (10) shows the overall loss function of the event extraction model:

$$Loss = \lambda \sum_{i=1}^{q} loss_c^i + (1-\lambda) \sum_{i=1}^{k} loss_e^i \tag{10}$$

where q represents the number of valid sentences in the document, k represents the number of event types, and λ is hyperparameter used to adjust the loss ratio of event argument extraction and event type detection tasks.

4. Experiments

In this section, we present the experimental results of TDJEE. We first describe implementation details, including the quality of knowledge base and hyperparameter setting. Then, we show experimental results including the baseline, knowledge base construction, main results of our method, and ablation performance.

4.1. Datasets and Settings

4.1.1. Datasets

The experimental data are company announcements from 2008 to 2018 trawled from the Chinese financial portal East Money (http://data.eastmoney.com/notices/hsa.html, 10 January 2020). Table 2 shows the number of announcement documents crawled for constructing knowledge base and the statistics of the dataset used in this paper. We obtain 31,748 documents of five event types in total, of which 5900 are used to manually labeled to verify the quality of dataset and 25,848 are used for distant supervision labeling. Then, the dataset is randomly shuffled and divided into training set, testing set, and validation set at a ratio of 8:1:1 for evaluating the TDJEE model.

Table 2. Statistics of dataset for evaluating construction of knowledge base and TDJEE. Num. of Doc represents the number of documents. Train, Validation, and Test means the numbers of documents for training, validating, and testing, respectively.

Event Type	**Num. of Doc**	**KB Experiments**		**TDJEE Experiments**		
		Train	**Test**	**Train**	**Validation**	**Test**
ER	3667	2967	700	2942	360	365
EF	1168	968	200	938	120	110
EU	5593	4593	1000	4453	560	580
EO	5787	4787	1000	4622	580	585
EP	15,533	12,533	3000	12,443	1555	1535
Total	31,748	25,848	5900	25,398	3175	3175

4.1.2. Hyperparameter Settings

In our implementation, we set the maximum number of sentences and the maximum sentence length to 64 and 128, respectively. Furthermore, the dimensions of hidden layer and output layer of feedforward in sub-task event argument extraction are set to 1024 and 43, respectively. During training, we set $\lambda = 0.2$. We employ the Adam [33] optimizer with the learning rate 10^{-4}, and train for at most 30 epochs. The batch size is set to 30 and the dropout rate is set to 0.1. More details settings of hyperparameters can be found in Table 3.

Table 3. Hyperparameters in the TDJEE model.

	Parameter	Value
Input Layer	Maximum Sentence Length	128
	Maximum Number of Sentences	64
BERT Layer	Model Type	BERT-Base Chinese
	Dimension	768
Feed Forward Layer	Hidden Layer Dimension	1024
	Output Dimension	43
Encoder Layer	Number of Layer	4
	Number of Heads	12
	Dimension	768
Event Argument Extraction	Number of Event Role	21
	Number of Label Type	43
Event Type Detection	Event Type	5
	Number of Classification Models	5
Training Parameters	Optimizer	Adam
	Epoch	30
	Learning Rate	10^{-4}
	Step	[2,4,**8**]
	Batch	[8,16,**32**]
	λ	[0.1,**0.2**,0.3,0.4]
	Dropout Rate	0.1

4.2. Experimental Results

4.2.1. Baseline

We use TIER [8], DCFEE [29], and Doc2EDAG [9] models as baseline models. TIER employs a pipeline architecture with three-stage task to get document-level context information: classifying narrative document, recognizing event sentence and noun phrase analysis. DCFEE divides the event extraction into two-stage tasks for processing. The first stage is sentence-level event extraction task, using bidirection LSTM [30] and CRF to mark the event arguments in the sentence. Event trigger words are detected through a dictionary. In the second stage, a convolution neural network (CNN) is used to identify event sentence and completeness of arguments. Doc2EDAG uses an end-to-end method to identify event arguments at the sentence level first. Then binary classifiers are used to determine the event type, and event arguments are transformed into a directed acyclic graph.

4.2.2. Performance of Knowledge Base Construction

We use distant supervision to label the structured event data to obtain the training corpus. Therefore, the quality of the knowledge base could influence the quality of labeled data. As Fonduer uses weak supervision to label data, it cannot directly verify the quality of the labeling function. Therefore, the quality of the dataset can be indirectly verified by testing the performance of the classification model in Fonduer.

First, an experiment of threshold in the classification model is studied. Figure 7 shows the influence of classification threshold. It can be seen that (1) with same threshold 0.65, three types of events—ER, EF, and EO—have achieved the highest F1 score, and (2) EP has the worst performance under this threshold. Therefore, we use different thresholds for different types of events to maximize the performance of all models. After obtaining the best classification model, we further perform an experiment on five types of event to verify the quality of event labeling. As shown in Table 4, it can be observed that event labeling on ER have performance with 83.3% F1 score. The F1 scores on other three event types are a little lower than that on ER. This may owe to more noise brought by labeling function. In addition, simply increasing the number of labeling function cannot improve the quality of event labeling.

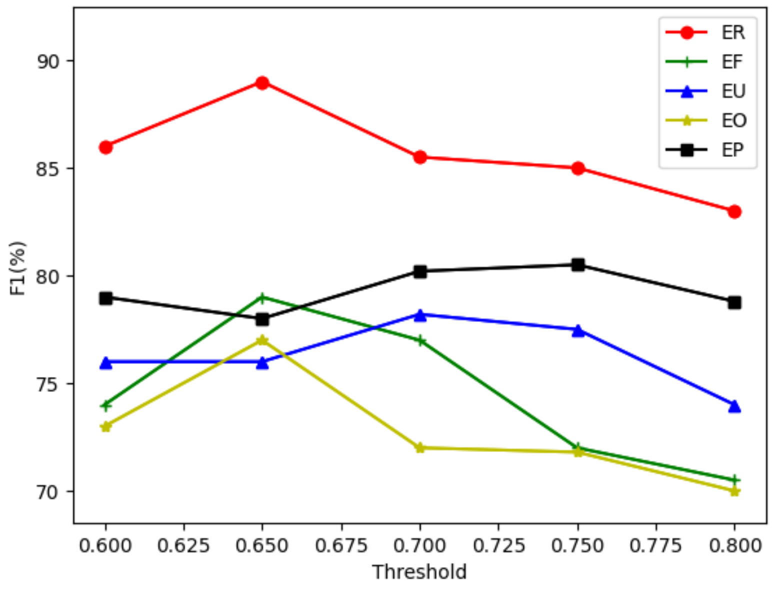


Figure 7. Threshold's influence on the F1 value of the classification model.

Table 4. Performance of event labeling on five types of event, where Role Num. represents the number of role of an event type, Function Num. represents the number of function for labeling an event type, and Threshold is the classification threshold in event labeling.

Event Type	Role Num.	Function Num.	Threshold	P (%)	R (%)	F1 (%)
ER	6	12	0.65	84.5	82.1	83.3
EF	8	16	0.65	88.2	68.8	77.3
EU	6	18	0.7	78.0	71.8	74.8
EO	6	17	0.65	78.6	75.3	76.9
EP	9	15	0.75	80.1	73.6	76.7

4.2.3. Main Results

Besides the pretrained model BERT, in our experiments, we also replace BERT with Bi-LSTM as embedding layer in TDJEE verify the advantage of our method. As Table 5 shows, TDJEE performs better than baseline models for almost all types of event. Specifically, compared with DCFEE, TDJEE improves 9.53%, 18.79%, 13.24%, 18.17%, and 22.49% F1 scores on ER, EF, EP, EO, and EU, respectively.

Table 5. Overall event-level precision (P.), recall (R.), and F1 scores.

Model	**ER**			**EF**			**EP**			**EO**			**EU**		
	P (%)	**R** (%)	**F1** (%)	**P** (%)	**R** (%)	**F1** (%)	**P** (%)	**R** (%)	**F1** (%)	**P** (%)	**R** (%)	**F1** (%)	**P** (%)	**R** (%)	**F1** (%)
TIER	76.28	77.85	77.06	52.08	32.64	40.13	55.85	49.17	52.30	43.13	34.38	38.26	54.61	25.84	35.08
DCFEE	81.91	76.94	79.35	54.14	35.04	42.54	58.63	53.75	56.08	43.07	37.51	40.10	54.71	30.66	39.30
Doc2EDAG	87.95	87.85	87.90	63.52	53.24	57.93	70.51	**68.85**	**69.67**	57.05	51.02	53.87	**77.07**	51.92	**62.04**
TDJEE-BiLSTM	79.31	74.53	76.85	64.06	48.53	55.22	57.90	55.00	56.41	50.54	42.04	45.90	64.53	46.60	54.12
TDJEE	**88.73**	**89.03**	**88.88**	**65.24**	**57.87**	**61.33**	**73.02**	65.97	69.32	**62.20**	**54.80**	**58.27**	75.19	**52.44**	61.79

In our dataset, event arguments scatter in multiple sentences. Meanwhile, an event role may appear in multiple events. Although DCFEE uses contextual information to predict event triggers, its context-agnostic arguments completion strategy makes it unable to effectively solve the event arguments scattering problem. Moreover, the direct document-level supervision are more robust than the extra sentence-level supervision used in DCFEE, which assumes that the sentences having most event arguments would contain the key event. This assumption does not work well on some event types, such as EF, EU, and EO. When event arguments appear in different sentences, original Transformer is hard to capture inner dependencies. Therefore, Doc2EDAG may fail to extract such event. In TIER, event extraction is divided into multi-stage tasks, but error propagation between multiple models is ignored. TDJEE uses joint learning to alleviate the influence of error propagation. Furthermore, even replacing BERT with Bi-LSTM, our method is still better than most baseline models. It mainly dues to that TDJEE can effectively capture contextual information, and integrate this information to form document-level contextual information.

4.2.4. Ablation Performance

To evaluate different parts of TDJEE, we perform an ablation experiment to compare three different variants: (1) CRF: removing the CRF layer from event argument extraction model. (2) Relation Encoding: removing relation encoding from Transformer, that is, replacing relation-aware Transformer with original Transformer. (3) Context Embedding: removing the context information from event type detection model, that is, removing relation-aware Transformer layer. The experimental results are shown in Table 6.

Table 6. Performance (F1 score) of TDJEE and its variants on five types of event.

Model	ER	EF	EP	EO	EU
TDJEE	88.88	61.33	69.32	58.27	61.79
-CRF	83.71	57.92	64.69	51.84	58.56
-Relation Encoding	82.38	51.24	59.21	49.45	51.23
-Context Embedding	73.47	43.00	55.54	41.12	41.15

It can be seen that, without the CRF layer, the F1 scores drop about 4 percentage points on each event type. At the same time, the F1 scores drop by 6.50%, 10.09%, 10.11%, 8.82%, and 10.56% without Relation Encoding. After removing Context Embedding, the performance of our model dropped rapidly. Specifically, F1 scores drop by 15.41%, 18.33%, 13.78%, 17.15%, and 20.64% on five event types, respectively. In event argument extraction model, CRF can correct outputs from BERT and reduce the occurrence of error propagation. Therefore, the effect of the TDJEE model can be improved by adding CRF layer. In event type detection model, explicitly encoding relations can not only help information flow in related sentences, but also enhance the weight of such relations in inner dependencies. This is important for downstream tasks. Furthermore, the influence of error propagation at event argument extraction stage can be alleviated. In addition, contextual information is crucial for the detection of event type. For an event role may appear in different events, ignoring the context information could lead to misclassification, reducing the performance of the model.

4.2.5. Evaluation of Knowledge Base and Hyperparameters

We compare the performance of the event extraction model under corresponding event knowledge base. As shown in Figure 8, the F1 score of knowledge base in ER is the highest, and the corresponding event extraction model has the best performance. Although EP is 4.2 times the size of ER in terms of dataset, the performances of knowledge base construction and event extraction model are lower than that of ER. Therefore, simply increasing the size of the corpus cannot effectively improve the performance of the model, and it is also necessary to improve the quality of knowledge base at the same time. EF and EP have very close performance of knowledge base, and their difference in the performance of the event extraction model between the two is less than 1%. EO and EF also have close performance of the knowledge base construction, but the performance of the event extraction model has a difference of 3%. Therefore, although different event types are close to each other in the performance of knowledge base construction, there may still be a large gap between the performance in event extraction model.

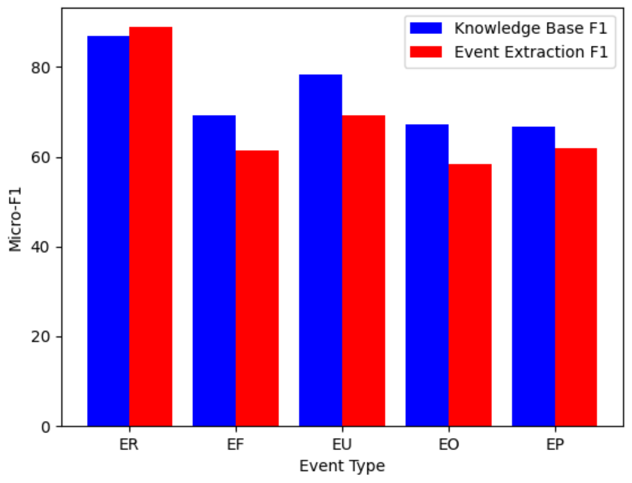


Figure 8. Comparison of knowledge base construction and event extraction model.

At the same time, in order to study the influence of hyperparameters in TDJEE model, we compared the P, R, and F1 scores with different hyperparameter λ and batch size. As shown in Figure 9a, when the value of λ is 0.2, the performance of the model is the highest, and when the values of λ are 0.3 and 0.4, the effect of the model shows a downward trend. As the input of the event extraction model is based on the document, the number of event arguments in the document is more than the event sentence. Therefore, the loss generated by the accumulation of event arguments is greater than that generated by event type detection, and a smaller value of λ can effectively balance the losses, thereby maximizing the performance.

Figure 9b shows the P, R, and F1 scores corresponding to the TDJEE model with different batch sizes. The training data set in this paper is processed on GTX 2080 GPU. As the batch size increases, the model converges faster under the same number of epoch. When the batch size is 8, it will take at least 30 epochs to converge. When the batch size is 32, the model has tended to converge after 20 epochs. At the same time, the F1 score of the model also increases as the batch size increases.

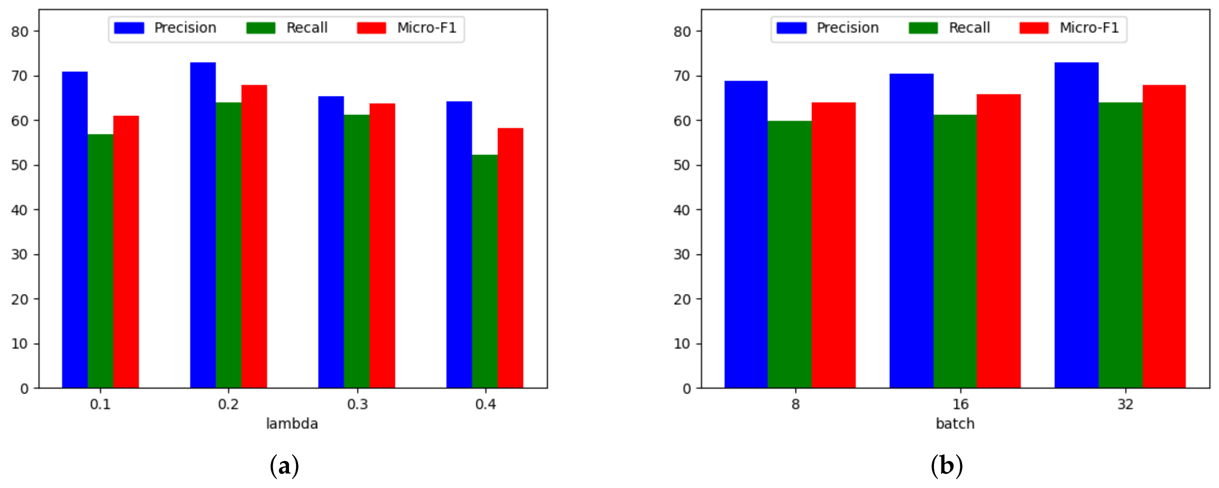


Figure 9. The impact of hyperparameter on the overall performance of the model. (**a**) The impact of λ. (**b**) The impact of batch.

5. Conclusions

In this paper, we first propose a new document-level joint event extraction model, TDJEE, to address the challenge of event arguments scattered in financial field. TDJEE includes two sub-models: event argument extraction and event type detection. Event argument extraction is regarded as a sequence labeling task. BERT and CRF are used for event argument extraction and role labeling. Event type detection is regarded as a binary classification task. Five discriminant models are defined for corresponding five event types. In addition, a relation-aware Transformer and attention network are used to further capture the semantic information of the document-level context. We build a Fonduer knowledge base and utilize a distant supervision method to label a large amount of high-quality corpora for training and evaluating.

Experimental results on real-world Chinese financial announcement dataset show that our model performs better than baseline models and achieves competitive results. It is worth mentioning that TDJEE is the language-independent model, and it can be used for financial event extraction in other languages.

Furthermore, our experimental results show that the quality of knowledge base could significantly effect the performance of event extraction in two sides. First, the errors in event knowledge base could cause wrong labels during distant supervision labeling, then wrong labels could affect the performance of TDJEE model. Especially, the quality of our current knowledge base is not perfect and has wrong event knowledge. Therefore, in the future, manual verification or reinforcement learning will be introduced to improve the quality of knowledge base. Second, as a joint model, even TDJEE overcomes the error propagation problem in the pipeline models, but it still suffers from the problem of argument extraction bottleneck, which is also worth exploring in the future work.

Author Contributions: Conceptualization, P.W., Z.D., and R.C.; methodology, P.W. and R.C.; validation, Z.D.; writing—original draft preparation, Z.D. and R.C.; writing—review and editing, P.W. All authors have read and agreed to the published version of the manuscript.

Funding: The work is supported by National Key R&D Program of China (2018YFD1100302), National Natural Science Foundation of China (No.61972082), and All-Army Common Information System Equipment Pre-Research Project (No.31511110310, No.31514020501, No.31514020503).

Data Availability Statement: The financial event extraction dataset used in this paper is available at https://github.com/q5s2c1/TDJEE/tree/master (20 March 2021).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hogenboom, F.; Frasincar, F.; Kaymak, U.; Jong, F.d.; Caron, E. A survey of event extraction methods from text for decision support systems. *Decis. Support Syst.* **2016**, *85*, 12–22. [CrossRef]
2. Qian, Q.; Pang, L.; Gao, S. An automatic question answering system for restricted fields based on word co-occurrence diagram. *Appl. Res. Comput.* **2013**, *30*, 841–843.
3. Hu, M.; Liu, B. Mining and summarizing customer reviews. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, 24–27 August 2004.
4. Huang, X.; Liao, G.; Xiong, N.; Vasilakos, A.V.; Lan, T. A Survey of Context-Aware Recommendation Schemes in Event-Based Social Networks. *Electronics* **2020**, *9*, 1583. [CrossRef]
5. Hong, Y.; Zhang, J.; Ma, B.; Yao, J.; Zhou, G.; Zhu, Q. Using cross-entity inference to improve event extraction. In Proceedings of the 49th annual meeting of the Association for Computational Linguistics, Portland, OR, USA, 19–24 June 2011.
6. Chen, Y.; Xu, L.; Liu, K.; Zeng D.; Zhao J. Event extraction via dynamic multi-pooling convolutional neural networks. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics, Beijing, China, 26–31 July 2015.
7. Wang, X.; Han, X.; Liu, Z.; Sun, M.; Li, P. Adversarial training for weakly supervised event detection. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics, Minneapolis, MN, USA, 3–5 June 2019.
8. Huang, R.; Riloff, E. Peeling back the layers: detecting event role fillers in secondary contexts. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, Portland, OR, USA, 19–24 June 2011.
9. Zheng, S.; Cao, W.; Xu, W.; Bian, J. Doc2EDAG: An end-to-end document-level framework for chinese financial event extraction. *arXiv* **2019**, arXiv:1904.07535.

10. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.
11. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
12. Lafferty, J.; McCallum, A.; Pereira, F.C. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the 2001 Eighteenth International Conference on Machine Learning (ICML), Williamstown, MA, USA, 28 June–1 July 2001.
13. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2015**, arXiv:1409.0473.
14. Wu, S.; Hsiao, L.; Cheng, X.; Rekatsinas, T.; Levis, P.; Ré, C. Fonduer: Knowledge base construction from richly formatted data. In Proceedings of the 2018 International Conference on Management of Data, Houston, TX, USA, 10–15 June 2018.
15. Freitag, D. Toward general-purpose learning for information extraction. In Proceedings of the 17th international conference on Computational linguistics, Montreal, QC, Canada, 10–14 August 1998.
16. Kim, J.T.; Moldovan, D.I. Acquisition of linguistic patterns for knowledge-based information extraction. *IEEE Trans. Knowl. Data Eng.* **1995**, *7*, 713–724.
17. Miller, G.A. Wordnet: a lexical database for english. *Commun. ACM* **1995**, *38*, 39–41. [CrossRef]
18. Zhao, Y.; Jin, X.; Wang, Y.; Cheng, X. Document embedding enhanced event detection with hierarchical and supervised attention. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018.
19. Han, Z.; Jiang, J.; Qiao, L.; Dou, Y.; Xu, J.; Kan, Z. Accelerating Event Detection with DGCNN and FPGAs. *Electronics* **2020**, *9*, 1666. [CrossRef]
20. Peng, G.; Chen, X. Entity–Relation Extraction—A Novel and Lightweight Method Based on a Gate Linear Mechanism. *Electronics* **2020**, *9*, 1637. [CrossRef]
21. Chen, Z.; Ji, H. Language specific issue and feature exploration in chinese event extraction. In Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics, Suntec, Singapore, 2–7 August 2009.
22. Liao, S.; Grishman, R. Acquiring topic features to improve event extraction: in pre-selected and balanced collections. In Proceedings of the International Conference Recent Advances in Natural Language Processing, Varna, Bulgaria, 2–4 September 2019.
23. Rosenberg, C.; Hebert, M.; Schneiderman, H. Semi-supervised self-training of object detection models. In Proceedings of the 7th IEEE Workshops on Application of Computer Vision, Breckenridge, CO, USA, 5–7 January 2005.
24. Wang, W.; Zhou, Z.H. Analyzing co-training style algorithms. In Proceedings of the 2007 European conference on machine learning, Warsaw, Poland, 17–21 September 2007.
25. Li, W.; Wong, K.; Yuan, C. A design of temporal event extraction from Chinese financial news. *Int. J. Comput. Process. Lang.* **2003**, *16*, 21–39. [CrossRef]
26. Ding, X.; Zhang, Y.; Liu, T. Knowledge-driven event embedding for stock prediction. In Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers, Osaka, Japan, 11–16 December 2016.
27. Dor, L.; Gera, A.; Toledo-Ronen, O. Financial Event Extraction Using Wikipedia-Based Weak Supervision. In Proceedings of the Second Workshop on Economics and Natural Language Processing, Hong Kong, China, 3–4 November 2019.
28. Pan, D.; Liang, Z.; Deng, Y. Textual Information Extraction Model of Financial Reports. In Proceedings of the 2019 7th International Conference on Information Technology: IoT and Smart City, Shanghai, China, 20–23 December 2019.
29. Yang, H.; Chen, Y.; Liu, K.; Xiao, Y.; Zhao, J. Dcfee: A document-level chinese financial event extraction system based on automatically labeled training data. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018.
30. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
31. Chen, Y.; Liu, S.; Zhang, X.; Liu, K.; Zhao J. Automatically labeled data generation for large scale event extraction. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017.
32. Shaw, P.; Uszkoreit, J.; Vaswani, A. Self-attention with relative position representations. *arXiv* **2018**, arXiv:1803.02155.
33. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.