Document-Level Multi-event Extraction via Event Ontology Guiding

Xingsheng Zhang[1,2], Yue Hu[1,2(✉)], Yajing Sun[1,2], Luxi Xing[1,2], Yuqiang Xie[1,2], Yunpeng Li[1,2], and Wei Peng[1,2]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
{zhangxingsheng,huyue,sunyajing,xingluxi,xieyuqiang, liyunpeng,pengwei}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. Document-level Event Extraction (DEE) aims to extract event information from a whole document, in which extracting multiple events is a fundamental challenge. Previous works struggle to handle the Document-level Multi-Event Extraction (DMEE) due to facing two main issues: (a) the argument in one event can correspond to diverse roles in different events; (b) arguments from multiple events appear in the document in an unorganized way. Event ontology is a schema for describing events that contains types, corresponding roles, and their structural relations, which can provide hints to solve the above issues. In this paper, we propose a document-level Event Ontology Guiding multi-event extraction model (EOG), which utilizes the structural and semantic information of event ontology as role-orientated guidance to distinguish multiple events properties, thus can improve the performance of document-level multi-event extraction. Specifically, EOG constructs Event Ontology Embedding layer to capture the structural and semantic information of event ontology. A transformer-based Guiding Interact Module is then designed to model the structural information cross-events and cross-roles under the guidance of event ontology. Experimental results on the DMEE dataset demonstrate that the proposed EOG can achieve better performance on extracting multiple events from the document over baseline models.

Keywords: Event extraction · Document-level · Multi-event · Event ontology · Transformer

1 Introduction

Document-level Event Extraction (DEE) aims to extract structural event information from a unstructured document according to the predefined event ontology, which is an essential task in Natural Language Processing (NLP). DEE can provide valuable structural information to facilitate various NLP tasks, such as language understanding, knowledge base construction, question answering, etc.

G. Memmi et al. (Eds.): KSEM 2022, LNAI 13369, pp. 312–324, 2022.
https://doi.org/10.1007/978-3-031-10986-7_25

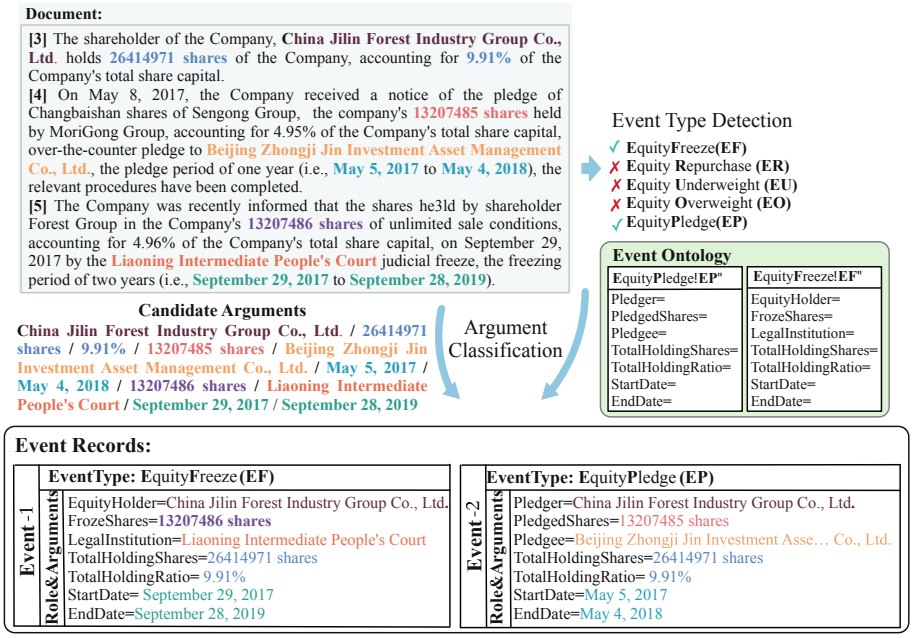

Fig. 1. An example of Document-level Multi-Event Extraction (DMEE). Words in colored are arguments that scatter across multiple sentences.

Existing DEE research [1,2,6,8,10,11] mainly focuses on extracting single event record from a document, but ignores the fact that multiple events appear in one document, which becomes a common issue in the document-level events extraction. As shown in Fig. 1, there are two events *Event-1* and *Event-2* appearing in the document simultaneously. However, existing models have poor ability to extract multiple events from document because the dependency relationships among different events and arguments are hard to capture. Therefore, multi-event is one crucial problem towards completing the DEE task.

In contrast to extracting single event from document, DMEE faces two critical challenges. The first challenge indicates that one event argument corresponds to different roles in different events. Figure 1 illustrates an example that the argument '*China Jilin Forest Industry Group Co., Ltd.*' plays *EquityHolder* role in *Event-1* with type *EquityFreeze*, and plays *Pledger* role in *Event-2* with event type is *EquityPledge* meanwhile. In order to distinguish different event records and different roles, model should have a global understanding of the entire document. Furthermore, it will be more difficult to extract events when a document coupled with the second challenge, in which arguments of multiple events appear disorderly in a document. As shown in Fig. 1, the arguments of *Event-1* appear in sentence `[3]` and `[5]`, and the arguments of *Event-2* appear in sentence `[3]` and `[4]`, indicate these events interlacing each other in the document. This issue requires model to recognize the dependency between these arguments

among multiple events. As a result of these challenges and requirements, the DMEE model calls for a global guidance to integrate effectively multiple events information from the document.

Ontology is the generalization of people's understanding of concrete entities, which reflects the relationship between the concepts beyond the concrete entities. In the event extraction task, event ontology depicts the relationships between the events and roles. Such structural schema provides the global guidance for distinguishing different events information and roles information. Specifically, event ontology contains event type, corresponding roles and the structural relationships between them. Previous works neglect the structural and semantic information of event ontology, and have difficulty in solving the **multi-roles** and **disorderly appearing** challenges, leading to poor performance in document-level multi-event extraction task.

Motivated by this, we attempt to incorporate event ontology as guidance to improve the performance of model in DMEE. In this paper, we propose a document-level **E**vent **O**ntology **G**uiding multi-event extraction model (EOG), which utilizes the structural and semantic information of event ontology as a role-orientated guidance to capture the multiple dependency relationships (i.e., event-event, role-role, event-role) Specifically, EOG constructing event ontology embedding layer to model the structural and semantic information of event ontology. To implement the guiding by event ontology, we design a transformer-based Guiding Interact Module (GIM) to model the interaction between document context and event ontology. GIM could capture the difference between the different events through interaction of sentences embeddings and event types embedding, and the dependency relationship between arguments and different roles through interaction of arguments embeddings and event roles embeddings. In this way, EOG constructs structural information cross-events and cross-roles under the guidance of event ontology. Then, we leverage a event ontology-aware decoder module for generating the event records (i.e., the predefined type with several event arguments corresponding to roles). At the stage of event type detection in the decoder, event type embedding is used to enhance the type-aware document embedding to improve the accuracy of event type detection. At the stage of argument classification in the decoder, event role embedding is used to enhance the role-aware arguments embedding to improve the accuracy of argument classification. Thus, EOG is encouraged to incorporate the inter-dependency and intra-dependency of event records under the guidance of event ontology to improve the ability to extract multiple events.

In summary, our contributions are as follows:

- We propose a **E**vent **O**ntology **G**uiding extraction model (EOG) for document-level multi-event extraction. EOG adopts a event ontology embedding layer, a transformer-based Guiding Interact Module (GIM) and a event ontology-aware decoder extract multiple events from document under the guidance of event ontology.
- We design a transformer-based guiding interact module (GIM) in EOG. GIM capture the dependency of different events and different roles by modeling the interaction between document context and event ontology.

– We conduct extensive experiments on the document-level multi-event extraction dataset, which is split from a widely used DEE dataset. Experiment results illustrate the superiority of EOG over state-of-the-art methods and verify the effectiveness of the guidance of event ontology in DMEE.

2 Related Work

Document-level EE has received widespread attention in recent years. [9] extract key-event from sentence and find other arguments from surrounding sentences to achieve document-level event extraction. [1] try to encode arguments in multi-granularity way for a larger context information. [11] design a two step approach to reduce the number of candidate arguments. [5] proposes an end-to-end neural event argument extraction model by conditional text generation. [2] leverage a structured prediction algorithm with deep value networks (DVN). However, these works mainly focus on solving the problem arguments scattering and ignored the challenge of multi-event.

To address the multi-event issue in document-level extraction, [12] reformulate DMEE as a table filling task, and propose an entity-based directed acyclic graph to fulfill event table. [10] design a multi-granularity decoder to extract events in parallel. [8] construct a heterogeneous graph to model the correlation among sentences and arguments. However, these methods model capture dependency between arguments and sentences only on document itself, and it is insufficient for document-level multi-event extraction.

3 Task Formulation

Before introducing our proposed model, we describe the formalization of the Document-level Multi Event Extraction (DMEE) task. Formally, we first clarify the following key notions: (a) **Entity Mention** is a text span that refers to an entity object, such as '*26414971 shares*' in *Document* in Fig. 1. (b) **Event Argument** stands for the entity's participant property in one event. Generally, an entity object plays a specific event role, such as '*13207486 shares*' plays the role *FrozeShares* in Fig. 1, thus the '*13207486 shares*' is the argument in this event. (c) **Event Role** provides the structural relations and semantic information between event type and its arguments. A type of event contains a set of corresponding roles which the arguments will play, such as *Pledger, Pledgee* corresponding to *EF* in Fig. 1. (d) **Candidate Argument** refers to the entity which can be identified as a specific argument in events, such as *Candidate Arguments* in Fig. 1. (f) **Event Ontology** is the predefined schema for event extraction containing n^T event types and corresponding event roles, such as *Event Ontology* in Fig. 1. It is notated as $\mathcal{O} = \{\mathcal{T}, \mathcal{R}\}$, where $\mathcal{T}$ is a set of event types, and $\mathcal{R}$ stands for the total event roles of all event types. Specifically, $\mathcal{T} = \{t_i\}_{i=1}^{n^T}$, and n^T is the number of event types, each event type t_i corresponding to event roles $\{r_1^{t_i}, r_2^{t_i}, ...\}$, and $\mathcal{R} = \{r_i\}_{i=1}^{n^R}$, and n^R is the number of total event roles.

(e) **Event Record** refers to an entry of a specific event type t_i containing arguments $\{a_1, a_2, ...\}$ corresponding to particular roles$\{r_1^{t_i}, r_2^{t_i}, ...\}$, such as *Event-1* record with type *EF*, *Role* and *Arguments* in Fig. 1;

For the DMEE task, given an input document comprised of n_s sentences $\mathcal{D} = \{s_i\}_{i=1}^{n_s}$, and each sentence s_i is a sequence of tokens $s_i = \left\{w_j^{s_i}\right\}_{j=1}^{n_w}$, where n_w is the number of tokens. The DMEE task aims to extract n_e structured event records $\mathcal{E} = \{e_i\}_{i=1}^{n_e}$.

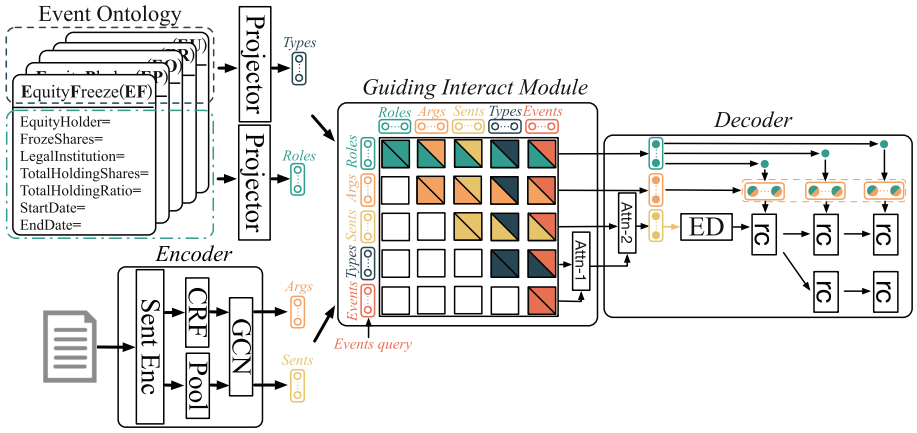

Fig. 2. The overall framework of EOG. Firstly, EOG encodes each sentences of document separately and recognize candidate arguments. Then a multi-layer GCN is used to get the document-level contextual representation. Meanwhile, EOG get representation of event ontology through two projectors. Next, a guiding interact module is designed to capture the dependency among different events and candidate arguments based on interaction between document context and event ontology representations. Finally, EOG generate the event records through event ontology-aware decoder.

4 Methodology

As shown in the Fig. 2, the proposed document-level Event Ontology Guiding multi-event extraction model (EOG) is composed of the following three modules: Encoder, Guiding Interact Module and Decoder. Specifically, (1) Encoder (Sect. 4.1) extracts candidate arguments from document and get contextualized embedding of candidate arguments and sentences. (2) Guiding Interact Module (GIM, Sect. 4.2) models the interaction between document context and event ontology, and enhance the embeddings of candidate arguments and document with the guidance of event ontology. (3) Decoder (Sect. 4.3) generates event records with the type-aware document embedding for detecting event type and the role-aware candidate arguments embedding for role classification.

4.1 Encoder

Given a document $\mathcal{D}$ contains n_s sentences, and each sentence contains n_w words. Encoder aims to get contextualized representation of candidate arguments and sentences. Encoder module is composed of four parts: (1) Sentence-level Encoder layer calculates the semantic representations based on transformer architecture. (2) Conditional Random Field (CRF) layer extracts the candidate arguments from each sentence. (3) Pooling layer gets representation of sentences and entity mentions. (4) Graph Convolution Network (GCN) layer enhances the representation of sentences and entity mentions by capturing the contextual information at document-level.

EOG adopts n_{se} layers Multi-head Self-Attention as the sentence-level encoder to obtain the sentence-level contextualized representation, each sentence s_i is embedded as follow:

$$\left(h_1^i, \ldots, h_{N_w}^i\right) = \text{Sent-Encoder}\left(w_1^i, \ldots, w_{N_w}^i\right) \tag{1}$$

where $h_j^i \in \mathbb{R}^d$ is the representation of j^{th} token in i^{th} sentence. And the representation of i^{th} sentence is obtained by max-pooling on all tokens representation in sentence s_i. Next, EOG employs a CRF layer to extract the entity mentions from each sentence through the BIO(Begin, Inside, Other) labels, and collects entity mentions extracted from all sentences of the input document. The task of extract entity mentions is optimized by the following loss:

$$\mathcal{L}_{er} = -\sum_{i=1}^{n_s} \log p\left(y_i^s \mid s_i\right) \tag{2}$$

where y_i^s is golden label sequence of s_i. An entity mention usually contains multiple tokens, EOG leverage max-pooling operation on multiple tokens representations of each entity mention to get the entity mention representation $S_{em} \in \mathbb{R}^{d \times n_{em}}$, n_{em} is the number of collected entity mentions of the document.

EOG introduces a document-level graph G to capture the dependency relationship between entity mentions at document-level. Specifically, there are two types nodes (sentences nodes and entity mentions nodes) and four types of edges (sentence-sentence edge, sentence-mention edge, Intra-Mention-Mention edge and Inter-Mention-Mention edge) in G. EOG applys Graph Convolution Network (GCN)[4] to model the contextual information at document-level. Specifically, the l-th layer graph convolution representation for node v in graph G is computed as:

$$h_v^{(l+1)} = \sigma\left(\sum_{u \in \mathcal{N}(v) \cup \{v\}} p_{uv}\left(W_{K(u,v)}^{(l)} h_u^{(l)}\right)\right) \tag{3}$$

where σ is the activation function(ReLU); $K(u, v)$ indicates the type of edge between node u and v, $W_{K(u,v)}^{(l)} \in \mathbb{R}^{d \times d}$ is the weight matrix; $p_(uv)$ is the normalization constant of the neighbor node u when updating node v; $\mathcal{N}(v)$ denotes the neighbors for node v.

And then the global information of one entity can be obtained by gathering its multiple entity mentions' feature through computing the average of its mention node representation. We regard these entity objects gathered from entity mentions as candidate arguments. In this way, we obtain the contextual representation of sentences $\mathcal{S} \in \mathbb{R}^{d \times n_s}$ and candidate arguments $\mathcal{A} \in \mathbb{R}^{d \times n_a}$, where n_a is the number of candidate arguments.

Besides, event ontology embedding layer contains two projectors are proposed to project each event type to a learnable embedding and each role to a learnable embedding respectively.

$$\mathbf{E}_{\text{types}} = \left(h_1^T, \ldots, h_{n^T}^T\right) = \text{Projector}_{Type}\left(t_1, \ldots, t_{n^T}\right) \tag{4}$$

$$\mathbf{E}_{\text{roles}} = \left(h_1^R, \ldots, h_{n^R}^R\right) = \text{Projector}_{Role}\left(r_1, \ldots, r_{n^R}\right) \tag{5}$$

Event types representation and event roles representation are denoted as $\mathbf{E}_{\text{types}} \in \mathbb{R}^{d \times n^T}$ and $\mathbf{E}_{\text{roles}} \in \mathbb{R}^{d \times n^R}$. These event ontology representations to represent the structure and semantic information of event ontology.

4.2 Guiding Interact Module

The encoder module only captures the contextual information of document, it is insufficient for model to distinguish the properties of different events. Thus, EOG apply a transformer-based[7] Guiding Interact Module (GIM) to introduce the structural and semantic information of event ontology as guidance to capture the dependency among different events and candidate arguments. In order to construct the relation among different records, EOG use n^Q learnable embeddings to denoted as event records $\mathbf{E}_{\text{events}} \in \mathbb{R}^{d \times n^Q}$. Then, the document-level representations (sentences representation $\mathcal{S}$ and candidate arguments representation $\mathcal{A}$), the event ontology representations (event types representation $\mathbf{E}_{\text{types}}$ and $\mathbf{E}_{\text{roles}}$), and the event records representations $\mathbf{E}_{\text{events}}$ are interacted with each other in GIM to facilitate the information exchange among these representations:

$$\left[\mathcal{A}^{pro}; \mathcal{S}^{pro}; \mathbf{E}_{\text{types}}^{pro}; \mathbf{E}_{\text{roles}}^{pro}; \mathbf{E}_{\text{events}}^{pro}\right] = \text{GIM}\left(\mathcal{A}; \mathcal{S}; \mathbf{E}_{\text{types}}; \mathbf{E}_{\text{roles}}; \mathbf{E}_{\text{events}}\right) \tag{6}$$

With the guidance of event ontology, the candidate arguments $\mathcal{A}^{pro}$ representation aggregate more information related to the various roles of different event records and various dependency relationship between arguments and event roles after interaction. And the sentences representations $\mathcal{S}^{pro}$ aggregate more structural and semantic information related to different events included in the document. Meanwhile, the event type representations $\mathbf{E}_{\text{types}}^{pro}$ and roles representations $\mathbf{E}_{\text{roles}}^{pro}$ are enhanced by infusing contextual information of current document.

To further model the co-existing relationships between event types and events, EOG utilize contextualized event types representation $\mathbf{E}_{\text{types}}^{pro}$ as query and the enhanced event records representation $\mathbf{E}_{\text{events}}^{pro}$ as key and value to enhance the event types representation through the attention weights:

$$\begin{aligned}\mathbf{E}_{\text{types}}^{plus} &= \text{Attn}_1\left(\mathbf{E}_{\text{types}}^{pro}, \mathbf{E}_{\text{events}}^{pro}, \mathbf{E}_{\text{events}}^{pro}\right) \\ &= \text{softmax}\left(\frac{\mathbf{E}_{\text{types}}^{pro}(\mathbf{E}_{\text{events}}^{pro})^T}{\sqrt{d}}\right)\mathbf{E}_{\text{events}}^{pro}\end{aligned} \tag{7}$$

Formally, after this module, we obtain the enhanced candidate argument representations $\mathcal{A}^{pro}$ with the guidance of event ontology, the contextualized event roles representations $\mathbf{E}_{\text{roles}}^{pro}$, and the further enhanced event types representations $\mathbf{E}_{\text{types}}^{plus}$. These aggregated representation serve the next module to generate event records.

4.3 Decoder

The decoder is responsible for generating the event records, which contains event detection and argument classification. Because of the multi-roles and disorderly appearing challenge for multi-event records, it's difficult to extract the accurate and meaningful information from the comprehensive representation for each event record. Therefore, EOG adopts event ontology representation as type-oriented and role-oriented guidance to generate each event record.

At the first stage of decoding, EOG leverage contextualized event types representations $\mathbf{E}_{\text{types}}^{plus}$ as query, and enhanced sentences representations $\mathcal{S}^{pro}$ to obtain the event type aware document representation $\mathbf{E}_{\text{doc}}$:

$$\begin{aligned}\mathbf{E}_{\text{doc}} &= \text{Attn}_2\left(\mathbf{E}_{\text{types}}^{plus}, \mathcal{S}^{pro}, \mathcal{S}^{pro}\right) \\ &= \text{softmax}\left(\frac{\mathbf{E}_{\text{types}}^{plus}(\mathcal{S}^{pro})^T}{\sqrt{d}}\right)\mathcal{S}^{pro}\end{aligned} \tag{8}$$

And then formulate the event detection subtask as a multi-label classification. EOG devises a classifier, which is marked as ED in Fig. 2, to judge whether each event type is triggered based on the type-aware document semantic representation $\mathbf{E}_{\text{doc}}$. The event type detection task adopt the following loss:

$$\mathcal{L}_{td} = -\sum_{i=1}^{n^T} \log p\left(y_i^t \mid \mathbf{E}_{\text{doc}}\right) \tag{9}$$

If the i-th event type is triggered in the document, then $y_i^t = 1$, otherwise $y_i^t = 0$.

Next, in the argument classification stage, EOG classifies the roles of each triggered event type according to the roles' predefined order. For each triggered event type, we formulate the argument classification subtask as task of expanding a tree orderly as previous methods[8,12]. We first define a event role order, and view each event record as a linked list of arguments following this order, where each argument node is either an entity or a special empty node. Classification starts from a root node, and expands by predicting arguments in a sequential order. For each node, we will judge whether candidate arguments playing the current role (role classification), and it will expand several branches by linking

the arguments assigned to the current role. The role classification is formulated as multi-label classification task. In this way, each path from the root node to the leaf node is identified as a unique event record.

EOG concat the candidate arguments embedding with the current role representation to obtain role-aware candidate argument representation $\mathcal{A}^{plus}$:

$$\mathcal{A}^{plus} = \mathbf{e}_i^R + \mathcal{A}^{pro} \tag{10}$$

And then EOG concatenate four representations for role classification: (1) role-aware candidate arguments representation $\mathcal{A}^{plus}$; (2) sentence representation $\mathcal{S}^{pro}$; (3) path memory [12], which is initialized by a memory tensor with the sentence representation at the beginning and updates when expanding the path by appending either the associated argument, to track the arguments already contained by the path; (4) global path memory [8], which encodes the argument representation sequence into to vector with an Long Short Term Memory networks (LSTM), to track the argument information of all paths.

EOG formulate the role classification as a multi-label classification. EOG devises a role classifier for each role, which is marked as *rc* in Fig. 2, to judge whether candidate arguments playing the current role based on the concatenated representation. Therefore, we drive the argument classification loss $\mathcal{L}_{ac}$:

$$\mathcal{L}_{ac} = -\sum_{i=1}^{n_r}\sum_{j=1}^{n_a} \log p\left(y_{ij}^r \mid \mathrm{a}_{ij}^{plus}\right) \tag{11}$$

where n_r is the number of all role classification nodes in event records trees; $\mathrm{a}_{ij}^{plus} \in \mathcal{A}^{plus}$ denotes the j-th candidate argument representation for i-th role classification node. For i-th role classification node, if the j-th candidate argument plays the current role, then $y_{ij}^r = 1$, otherwise $y_{ij}^r = 0$.

During training, we sum the losses coming from three tasks together as the final loss: $\mathcal{L}_{\text{all}} = \lambda_1\mathcal{L}_{\text{er}} + \lambda_2\mathcal{L}_{td} + \lambda_3\mathcal{L}_{ac}$ where λ_1, λ_2, λ_3 are hyper-parameters.

5 Experiments

5.1 Experimental Settings

Dataset. We evaluate our model on a Chinese financial DEE dataset proposed by [12]. It contains five event types: Equity Freeze (EF), Equity Repurchase (ER), Equity Underweight (EU), Equity Overweight (EO) and Equity Pledge (EP), with 35 different kinds of argument roles in total. This dataset contains 32,040 documents in total, but only 29% documents of it express multiple events. In order to evaluate the effectiveness of the model for multi-event extraction, we split the multi-event version dataset (i.e., Train_m, Dev_m, Test_m) from the original version dataset (i.e., Train_o, Dev_o, Test_o) according to the number of events contained in a document. The proposed EOG model is trained on Train_m, evaluated on Test_m and Test_o. The Dev_m and Dev_o are used to select the parameters for the proposed EOG model respectively.

Metrics. We adopt the standard evaluation metrics, i.e., Precision (P), Recall (R), and F1 score (F1), as originally used in Doc2EDAG [12]. The following results are reported with micro-averaged role-level scores as the final event-level metric.

Implementation Details. The event ontology contains 5 event types and 35 roles in total. The dimensions of event ontology embedding and the representation of candidate arguments and sentences are 768. The max sentences length is 128. The max sentences number in one document is 64. The number of GCN layers is 3. The number of Transformer-blocks in the Global Interact Module is 2. The number of learnable embeddings for generated event records is 4. During training, we employ Adam [3] optimizer with 1e-4 learning rate for 100 epochs. The training batch size is 64 and the steps of gradient accumulation is 16. We se t $\lambda_1 = 0.05$, $\lambda_2 = \lambda_3 = 1$ for the loss function. We run all models on the 11G GeForce GTX 1080 Ti GPU.

Table 1. Overall event-level Precision (P.), Recall (R.) and F1 on the multi-event version test set (Test_m).

Models	EF			ER			EU			EO			EP			Overall		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DCFEE-M	42.9	43.9	43.4	38.8	54.4	45.3	39.1	38.3	38.7	44.7	42.6	43.6	51.0	53.5	52.2	48.6	51.3	49.9
Doc2EDAG	72.6	49.1	58.6	72.8	60.7	66.2	72.8	56.1	63.4	73.9	60.8	66.7	78.9	66.5	72.1	77.5	64.0	70.1
GIT	**76.7**	48.5	59.4	73.9	**68.1**	70.9	72.5	**58.7**	**64.9**	72.2	**66.0**	69.0	73.4	**70.5**	71.9	73.4	**67.9**	70.6
Ours EOG	67.3	**61.8**	**64.4**	**76.7**	67.7	**71.9**	**75.7**	56.7	64.8	**77.2**	63.2	**69.5**	**79.6**	67.7	**73.1**	**78.1**	66.4	**71.8**

Table 2. F1 scores of each event type on the original dataset (Test_o) and the average F1 scores (Avg.) on respective single-event split and multi-event split.

Models	EF		ER		EU		EO		EP		Avg.		
	S.	M.	S.	M.	S.	M.	S.	M.	S.	M.	S.	M.	S.&M.
DCFEE-O	49.6	41.7	70.0	50.8	51.3	38.8	45.8	46.8	61.2	53.1	55.6	46.2	52.3
DCFEE-M	42.6	41.4	57.9	44.2	47.3	41.1	41.5	41.6	56.7	52.3	49.2	44.1	47.6
GreedyDec	70.4	37.9	74.1	49.0	57.6	31.3	62.0	29.6	77.0	36.9	68.2	36.9	55.4
Dco2EDAG	67.9	57.5	74.7	66.9	68.5	61.2	**69.5**	66.4	**79.2**	71.7	72.0	64.8	68.9
GIT	72.5	59.7	71.1	67.1	**70.4**	61.6	66.7	67.9	75.3	71.0	71.6	65.5	69.0
Ours EOG	**73.1**	**60.8**	**76.2**	**68.6**	70.1	**65.7**	68.2	**69.1**	77.6	**73.0**	**73.1**	**67.4**	**70.7**

5.2 Baselines

We compare the proposed EOG model with the following baseline methods: (1) **DCFEE** [9] processes a sentence-level extraction model based on a sequence tagging model and expanded the arguments from the surrounding sentences as the result of document-level extraction. The model has two variants which are **DCFEE-O** that only produces one event record and **DCFEE-M** that can produce multiple event records from a document. (2) **Doc2EDAG** [12] uses transformer encoder to obtain sentence and entity embeddings and regards multiple

event records generation as path expanding based entity. (3) **GreedyDec** [12] is a simple varient of Doc2EDAG, which generate one event record greedily. (4) **GIT** [8] proposes a heterogeneous graph to get the interactions between sentences and entity mentions, and utilizes a tracker-based RNN to memory the information of multiple events for multiple event records generation.

5.3 Main Results

Overall Performance. In multi-event version split dataset (i.e., Train_m, Dev_m and Test_m), as shown in Table 1, the proposed EOG achieves significant improvements overall baselines, thanks to the guidance of event ontology. Specifically, EOG improves 1.2 micro F1 compared with the previous state-of-the-art, GIT, especially EOG gains 5.0 improvement in Equity Freeze (EF) event type. Compared to other event types, the number of documents that contain EF events is the least, which leads to the low performance for general models. However, the improved performance indicates that the correlation between different event types is also helpful for current event extraction. With the learnable embedding of event ontology, EOG could improve the extraction performance of EP event records with the knowledge learned from other event types of documents. Furthermore, the vast improvement of EF events extraction also proves the effectiveness of event ontology guidance in event extraction.

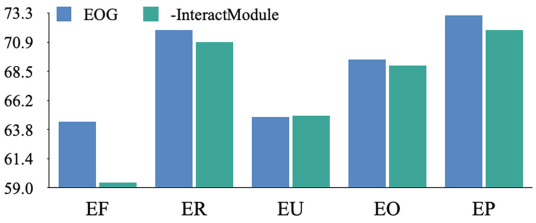

Fig. 3. F1 score of ablation studies on EOG for each event type.

Single-event vs. Multi-event. Multi-Event documents are more complex, as they owns longer text and more entities, than single-event documents. We assume that the model trained on a Multi-Event dataset can also work out the Single-Event extraction. To evaluate the effectiveness of our model in the actual scenario with a mixture of documents comprising Single-Event and Multi-Event, We train EOG model on multi-event train dataset (Train_m and evaluate on original dataset (Dev_o/Test_o). Table 2 shows the F1 score on single-event and multi-event sets for each event type and the averaged (Avg.) under the experiment setting (Train_m/Dev_o/Test_o). We can observe that EOG surpasses the SOTA baseline by 1.1 and 1.9 on single-record and multi-record sets, respectively. The above result demonstrates that event ontology guidance is effective in multi-event extraction and single-event scenario.

5.4 Ablation Studies

To verify the essential designs of EOG, we conduct ablation tests by removing the interact module. The results are shown in Fig. 3, we can observe that: 1)the micro F1 decreases 1.5 on average demonstrates the effectiveness of event ontology guidance for event extraction; 2)significant drop on Equity Freeze(EF) type has been discussed in 5.3. 3)event ontology guidance contributes less on Equity Repurchase(ER) type mainly because ER documents are much longer than other documents, the encoder could not work effectively for arguments scattering.

6 Conclusion

We propose a multi-event extraction model via event ontology guiding (EOG) to tackle multi-roles and disorderly appearing challenges in Document-level Multi-Event Extraction (DMEE). EOG introduces event ontology in the form of learnable embedding to provide structural and semantic information of event. And EOG designs a Guiding Interact Module to model and enhance features of events through the cross-event and cross-roles interaction under the guidance of the event ontology. Then, EOG employs a decoder to generate the event records with the event ontology aware event features. Experimental results show that EOG can significantly outperform previous methods in the multi-event scenario. Further analysis verifies the effectiveness of event ontology guidance for multi-event extraction.

Acknowledgement. We thank all anonymous reviewers for their constructive comments and we have made some modifications. This work is supported by the National Natural Science Foundation of China (No. U21B2009).

References

1. Du, X., Cardie, C.: Document-level event role filler extraction using multi-granularity contextualized encoding. In: Proceedings of the ACL, pp. 8010–8020 (2020)
2. Huang, K.H., Peng, N.: Document-level event extraction with efficient end-to-end learning of cross-event dependencies. In: Proceedings of the Third Workshop on Narrative Understanding, pp. 36–47 (2021)
3. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, 7–9 May 2015, Conference Track Proceedings (2015)
4. Kipf, T., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2017)
5. Li, S., Ji, H., Han, J.: Document-level event argument extraction by conditional generation. In: Proceedings of the NAACL, pp. 894–908 (2021)
6. Lou, D., Liao, Z., Deng, S., Zhang, N., Chen, H.: MLBiNet: a cross-sentence collective event detection network. In: Proceedings of the ACL, pp. 4829–4839 (2021)
7. Vaswani, A., et al.: Attention is all you need. arXiv preprint arXiv:1706.03762 (2017)

8. Xu, R., Liu, T., Li, L., Chang, B.: Document-level event extraction via heterogeneous graph-based interaction model with a tracker. In: Proceedings of the ACL, pp. 3533–3546 (2021)
9. 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 ACL 2018, System Demonstrations, pp. 50–55 (2018)
10. Yang, H., Sui, D., Chen, Y., Liu, K., Zhao, J., Wang, T.: Document-level event extraction via parallel prediction networks. In: Proceedings of the ACL, pp. 6298–6308 (2021)
11. Zhang, Z., Kong, X., Liu, Z., Ma, X., Hovy, E.: A two-step approach for implicit event argument detection. In: Proceedings of the ACL, pp. 7479–7485 (2020)
12. Zheng, S., Cao, W., Xu, W., Bian, J.: Doc2EDAG: an end-to-end document-level framework for Chinese financial event extraction. In: Proceedings of the EMNLP-IJCNLP, pp. 337–346 (2019)