applied sciences

MDPI

Article

DEEDP: Document-Level Event Extraction Model Incorporating Dependency Paths

Hui Li [1,*], Xin Zhao [2], Lin Yu [1], Yixin Zhao [1] and Jie Zhang [1]

[1] School of Automation, Nanjing University of Science and Technology, Nanjing 210094, China
[2] Science and Technology on Information Systems Engineering Laboratory, Nanjing 210023, China
* Correspondence: mcclane@njust.edu.cn

Abstract: Document-level event extraction (DEE) aims at extracting event records from given documents. Existing DEE methods handle troublesome challenges by using multiple encoders and casting the task into a multi-step paradigm. However, most of the previous approaches ignore a missing feature by using mean pooling or max pooling operations in different encoding stages and have not explicitly modeled the interdependency features between input tokens, and thus the long-distance problem cannot be solved effectively. In this study, we propose Document-level Event Extraction Model Incorporating Dependency Paths (DEEDP), which introduces a novel multi-granularity encoder framework to tackle the aforementioned problems. Specifically, we first designed a Transformer-based encoder, *Transformer-M*, by adding a Syntactic Feature Attention mechanism to the Transformer, which can capture more interdependency information between input tokens and help enhance the semantics for sentence-level representations of entities. We then stacked Transformer-M and Transformer to integrate sentence-level and document-level features; we thus obtained semantic enhanced document-aware representations for each entity and model long-distance dependencies between arguments. Experimental results on the benchmarks MUC-4 and ChFinAnn demonstrate that DEEDP achieves superior performance over the baselines, proving the effectiveness of our proposed methods.

Keywords: document-level event extraction; dependency path; multi-granularity encoder; Transformer; syntactic feature attention

check for updates

Citation: Li, H.; Zhao, X.; Yu, L.; Zhao, Y.; Zhang, J. DEEDP: Document-Level Event Extraction Model Incorporating Dependency Paths. *Appl. Sci.* **2023**, *13*, 2846. https://doi.org/10.3390/app13052846

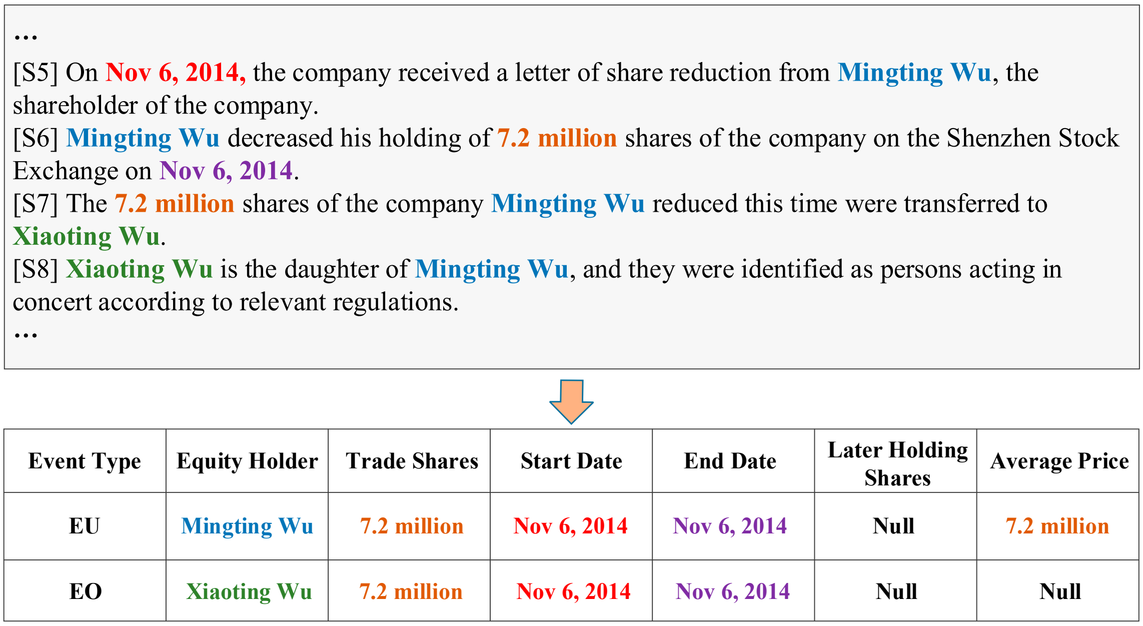
Academic Editors: Shengzong Zhou and Jingsha He

Received: 26 December 2022
Revised: 18 February 2023
Accepted: 21 February 2023
Published: 22 February 2023

1. Introduction

Event extraction (EE) is a crucial and challenging task for Information Extraction (IE), which aims to identify events of pre-defined types and extract arguments to fill the corresponding roles from plain texts. In recent years, EE has received growing attention in many domains, such as Finance, Public Safety, Intelligent Operations, and Maintenance, because it can produce valuable structured event knowledge to facilitate critical incident handling in these domains. Most existing approaches [1–6] mainly explore sentence-level EE (SEE), which detects and extracts events from a single sentence within the given document. Moreover, the evaluation work of these approaches is mainly based on a manually annotated benchmark, ACE-2005 [7], which labels only event arguments within a sentence scope. However, there may be more than one event described in a real-world text, and the arguments for an event record may distribute in multiple sentences. Therefore, the SEE performs poorly when extracting events across sentences. To this end, document-level event extraction (DEE) is proposed to address the aforementioned problems.

Compared with SEE, DEE focuses on two vital challenging tasks: arguments-scattering and multi-events. Arguments-scattering refers to the arguments of an event that may be distributed in different sentences in a document. As illustrated in Figure 1, the arguments of Equity Overweight (EO) scatter across four sentences ([S5] to [S8]), and the extraction cannot be completed within a single sentence. In this scenario, the DEE model needs to understand the whole document comprehensively and capture long-distance interaction information among arguments distributed in different sentences. Multi-events indicates that a document may describe several correlated events. The two events, Equity Underweight (EU) and EO, are interdependent and there is no obvious textual boundary between them. To tackle this challenge, DEE models need to integrate local and global information, fill the event roles of a pre-defined type with relevant arguments that scatter across sentences, and identify multiple events simultaneously. However, in real-world applications, these challenges are often coupled, which makes the DEE task more difficult and complicated.



…

[S5] On **Nov 6, 2014,** the company received a letter of share reduction from **Mingting Wu**, the shareholder of the company.

[S6] **Mingting Wu** decreased his holding of **7.2 million** shares of the company on the Shenzhen Stock Exchange on **Nov 6, 2014**.

[S7] The **7.2 million** shares of the company **Mingting Wu** reduced this time were transferred to **Xiaoting Wu**.

[S8] **Xiaoting Wu** is the daughter of **Mingting Wu**, and they were identified as persons acting in concert according to relevant regulations.

…

Event Type	Equity Holder	Trade Shares	Start Date	End Date	Later Holding Shares	Average Price
EU	Mingting Wu	7.2 million	Nov 6, 2014	Nov 6, 2014	Null	7.2 million
EO	Xiaoting Wu	7.2 million	Nov 6, 2014	Nov 6, 2014	Null	Null

Figure 1. An example document from ChFinAnn proposed by Zheng et al. [8]. In the document, only four sentences (S5–S8) are shown, the lists Equity Underweight (EU) and Equity Overweight (EO) are extracted events, and the bold-faced and coloured words are event arguments with specific roles.

To address the two aforementioned challenges, previous studies [8–13] employed a multi-Transformer [14] or Long Short-Term Memory (LSTM) [15] encoder to obtain sentence-level and document-level representations for entities and sentences, ignoring the feature loss problem while performing mean pooling and max pooling operations, and without the ability to model long-distance dependencies between arguments. Yang et al. [16] noted the latter issue, and proposed the application of a multi-turn and multi-granularity Transformer encoder to model local and global interaction, which can capture dependencies between arguments and help to understand the entire document comprehensively. Although this approach is promising, it overlooks the feature loss problem, and the input token features they used are limited.

In this work, to overcome the aforementioned shortcomings of previous methods and address the troublesome challenges in DEE, we propose a novel end-to-end model, the Document-level Event Extraction Model Incorporating Dependency Paths (DEEDP). DEEDP introduces a novel multi-granularity encoder framework that can model long-distance dependencies and capture more sentence-level syntactic features to facilitate DEE. In detail, we first designed a Transformer-based encoder, Transformer-M, by adding a Syntactic Feature Attention mechanism to the Transformer, which can capture more interdependency information between input tokens and help enhance the semantics for sentence-level representations of entities. We then stacked Transformer-M and Transformer to integrate sentence-level and document-level features; we thus obtained semantic enhanced document-aware representations for each entity and modeled long-distance dependencies between arguments.

To verify the effectiveness of the proposed DEEDP, we performed experiments on MUC-4 and ChFinAnn, two widely used DEE benchmarks, and reported the experimental results of the standard evaluation metrics. Significantly, our method achieved +1.3 and +2.1 improvements over the current state-of-the-art (SOTA) model on the MUC-4 and the ChFinAnn dataset, respectively.

Our contributions can be summarized as follows:

1. We proposed a novel DEE model, DEEDP, which introduces a novel multi-granularity encoder framework to effectively tackle the unique challenges of DEE.
2. We designed a Transformer-based encoder, Transformer-M, which can explicitly model the dependency between input tokens to avoid feature loss and enhance the semantic representations for arguments.
3. We performed extensive experiments on the widely used DEE benchmarks, and the experimental results demonstrated the superiority of DEEDP.

2. Related Works

2.1. Sentence-Level Event Extraction

Most previous EE approaches focus mainly on the sentence level and evaluate their performance on the expert-annotated benchmark ACE-2005 [7]. In the early years, researchers concentrated mainly on utilizing hand-designed features to conduct trigger and argument extraction. Li et al. [17] developed a joint EE framework based on structured prediction that can incorporate global features to jointly extract triggers and arguments. Li et al. [18] utilized the knowledge encoded in the Abstract Meaning Representation (AMR) to capture the deeper semantic information of trigger words and then used these features to extract the triggers. With the development of neural network theory and an increase in computing power, some studies have applied neural-based methods to automatically capture local and global features for trigger and argument extraction [19]. Recent studies focus mainly on utilizing more knowledge such as document context information [20,21], dependency tree information [13,22], and external knowledge incorporation [23,24]. These methods have achieved great success on SEE; however, these extractions are conducted on the sentence level.

2.2. Document-Level Event Extraction

In many real scenarios, several event records may be described in a document, and arguments for a record may be scattered. Consequently, DEE has attracted increasing attention, and many promising techniques have been developed. To evaluate their performance, two benchmark datasets, MUC-4 [16] and ChFinAnn [8], were constructed. MUC-4 is a document-level template-filling task that aims at identifying event arguments for filling the corresponding pre-defined roles with associated role types from a document. Researchers on EE have proposed a variety of advanced approaches to improve performance on this task. Recent studies have explored the extraction of role fillers utilizing manually designed linguistic features [25] and neural-based contextual representations [26–28].

For arguments-scattering and multi-events, the real challenges for DEE, previous studies focus mainly on the ChFinAnn task. DCFEE [9] proposed a key event detection model for DEE that extracts events from the central sentence and searches for missing arguments from the surrounding ones. Doc2EDAG [8] treated DEE as a table-filling task and utilizes multiple Transformer encoders to capture sentence and document features to obtain document-aware entities and sentence representations, and then performs triggered event records construction with a tracker. GIT [11] proposed a heterogeneous graph interaction framework to model global information interactions among sentences and entities, and adopted an improved tracker to construct detected event records. DE-PPN [12] proposed an encoder-decoder model to obtain document-aware representations and generate events in parallel. MMR [16] transforms DEE into a machine reading comprehension (MRC) task with a multi-granularity reader framework. Although these methods have improved the performance of DEE, they ignore the feature loss problem caused by mean pooling and max pooling operations or neglect the interdependency between local semantic information and global semantic information, which limits performance improvement.

3. Preliminaries

We first clarify a set of key notions: (1) **entity mention**: some consecutive words within a sentence that refers to a specific entity object; (2) **event role**: a pre-defined field of an event type; (3) **event argument**: an entity that can fill a specific event role field for an event; (4) **event record**: an entry corresponding to a specific pre-defined event type and its associated event roles. For example, Figure 1 presents two event records, where the entity "*Xiaoting Wu*" plays an *Equity Holder* role in event EO, i.e., "*Xiaoting Wu*" is an entity mention that refers to a person and this entity is taken as an event argument to fill the *Equity Holder* role for the event EO.

Following Xu et al. [11], we first introduce the formalization of the DEE task, and then describe our proposed approaches in the following sections. Specially, we denote $\mathcal{T}$ and $\mathcal{R}$ as the set of pre-defined event types and role categories, respectively. Given a document with N_s sentences, $\mathcal{D} = \{\mathrm{S_i}\}_{\mathrm{i=1}}^{\mathrm{N_s}}$, where each S_i contains a sequence of tokens, we denote it as $S_i = \{w_{i,j}\}_{\mathrm{j=1}}^{\mathrm{l}}$, where l is the length of the sentence. The DEE task aims to extract possible event records $Y = \{y_i\}_{i=1}^{k}$, where each y_i^t corresponds to a pre-defined event type t, and contains several event roles $\left(r_i^1, r_i^2, \cdots, r_i^n\right)$ filled by corresponding arguments $\left(a_i^1, a_i^2, \cdots, a_i^n\right)$. The variable k represents the number of events described in a document, and n is the number of pre-defined roles for event type t, $t \in \mathcal{T}$, and $r \in \mathcal{R}$.

4. Methodology

In this section, we first present the overall architecture of DEEDP. We then introduce the detailed implementations of every module that consists of DEEDP.

4.1. Model Architecture

Figure 2 depicts the overall architecture of DEEDP, composed of four stacked modules: (1) the sentence-level encoder responsible for capturing basic features, like lexical and syntactic features, from the tokens of the input sentences, and projecting the tokens into a continuous low-dimension space, after which we can obtain their sentence-level representations, which are fed into a conditional random field (CRF) layer to perform named entity recognition (NER); (2) the document-level encoder responsible for encoding the representations of entities and sentences at the sentence-level into a united feature space, so that we can obtain their document-level representations; (3) the sentence-document feature fusion module designed and responsible for capturing more lexical and syntactic features, enhancing the document-level representations of entities and sentences; (4) the event record module responsible for organizing the candidate entities to construct event records included in the input documents. The implementation of each module is introduced in the following subsections.

4.2. Sentence-Level Encoder

Transformer Encoder. Following most pre-training language models, such as BERT [29], MBERT [30], MLRIP [31], SpanBERT [32], and document-level event and event element joint extraction models such as Doc2EDAG [8] and MMR [16], we applied Transformer [14] as our basic encoder, which utilizes a multi-head attention and masking strategy to capture the lexical and syntactic information for each token in the input sequence, and the corresponding contextual representation for each token can be generated.

Input Embeddings. Modelling more features of the input tokens facilitates downstream tasks. Compared with Doc2EDAG, MMR, and DEEB-RNN [33], we modeled a part-of-speech (POS) feature, entity type feature, and document topic feature to represent each token of the input sentence, rather than using only the token and position features. Formally, given a document $\mathcal{D} = \{S_i\}_{i=1}^{N_s}$ containing N_s sentences, sentence S_i can be denoted as a sequence of tokens $[w_{i,1}, w_{i,2}, \cdots, w_{i,l}]$, where l is the length of the sentence, and $w_{i,j} \in R^d$ denotes the representation for the jth token of the S_i. The representations of the token sequence were fed into the Transformer Encoder, *TF*-1, and the final hidden state C_i of the S_i could be used for down-stream tasks, which can be calculated as follows:

$$C_i = TF - 1(w_{i,1}, \cdots, w_{i,l}) \tag{1}$$

where $C_i = [c_{i,1}, c_{i,2}, \cdots, c_{i,l}]$, $c_{i,j} \in R^d$ is the final hidden state for the jth token of S_i and $C_i \in R^{l \times d}$ is the final hidden state of S_i. Then C_i was fed into a CRF layer for sequence labelling. After that, we could obtain the candidate event arguments.

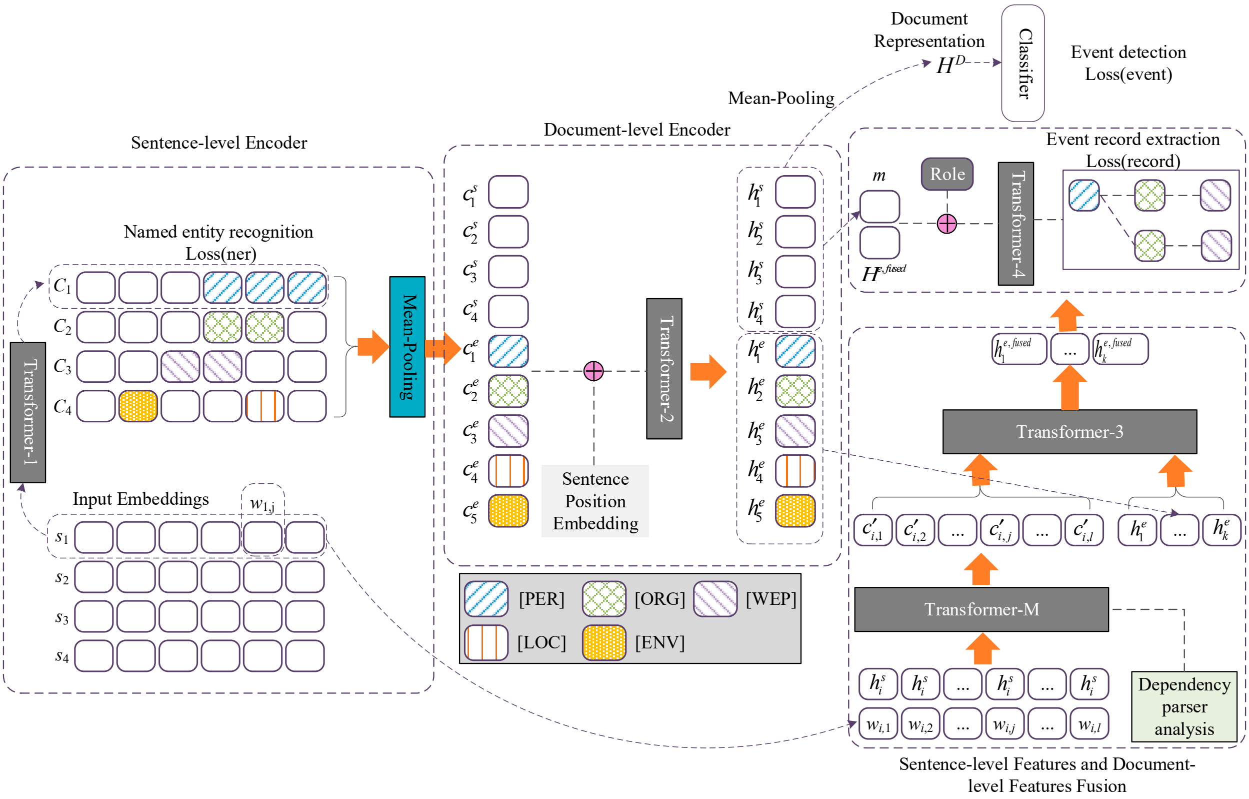

Document Representation
H^D
Classifier
Event detection Loss(event)
Mean-Pooling
Sentence-level Encoder
Document-level Encoder
Event record extraction Loss(record)
m
Role
Transformer-4
$H^{e,fused}$
Named entity recognition Loss(ner)
C_1
C_2
C_3
C_4
Transformer-1
Mean-Pooling
c_1^s c_2^s c_3^s c_4^s
c_1^e c_2^e c_3^e c_4^e c_5^e
Transformer-2
h_1^s h_2^s h_3^s h_4^s
h_1^e h_2^e h_3^e h_4^e h_5^e
Sentence Position Embedding
$h_1^{e,fused}$ ... $h_k^{e,fused}$
Transformer-3
$c'_{i,1}$ $c'_{i,2}$ ... $c'_{i,j}$ ... $c'_{i,l}$ h_1^e ... h_k^e
Input Embeddings
$w_{1,j}$
s_1 s_2 s_3 s_4
[PER] [ORG] [WEP]
[LOC] [ENV]
Transformer-M
h_i^s h_i^s ... h_i^s ... h_i^s
$w_{i,1}$ $w_{i,2}$ ... $w_{i,j}$ ... $w_{i,l}$
Dependency parser analysis
Sentence-level Features and Document-level Features Fusion

Figure 2. DEEDP model overview. For a given document, we first conduct NER with the sentence-level encoder. Then we feed the sentence-aware representations of entities and sentences of the input document to the document-level encoder to acquire their document-aware representations. To aggregate both representations for each entity and enhance their semantic, we designed a Transformer-M and stacked it with a Transformer. Finally, the fused representations of entities and document-aware representations of sentences are fed into the event record extraction module for event record construction.

Given a token, its input representation is the sum of the embeddings for token, segment, position, POS, entity type, and document topic. A visualization of the input embedding is illustrated in Figure 3.

Entity Recognition. Entity recognition aims to extract all candidate elements for the events described in a given document and is always taken as a sequence tagging task. In this proposed work, we performed entity recognition at the sentence level and followed GIT and Doc2EDAG, which added a CRF layer to the final Transformer Encoder to conduct sequence tagging for the input sentences. To be specific, given a sentence s, we applied a Transformer encoder (*TF*-1) to encode it and the final hidden states $C_i = TF - 1(s)$ were fed into the CRF layer to conduct entity recognition with the BIO (Begin, Inside, Other) schema. For training, we minimized the sequence tagging loss as follows:

$$\mathcal{L}(ner) = -\sum_{s \in \mathcal{D}} \log(P(y_s|s)) \tag{2}$$

where y_s is the golden-label tagging sequence for s. With respect to inference, we applied the Veterbi decoding algorithm to obtain the golden label sequence.

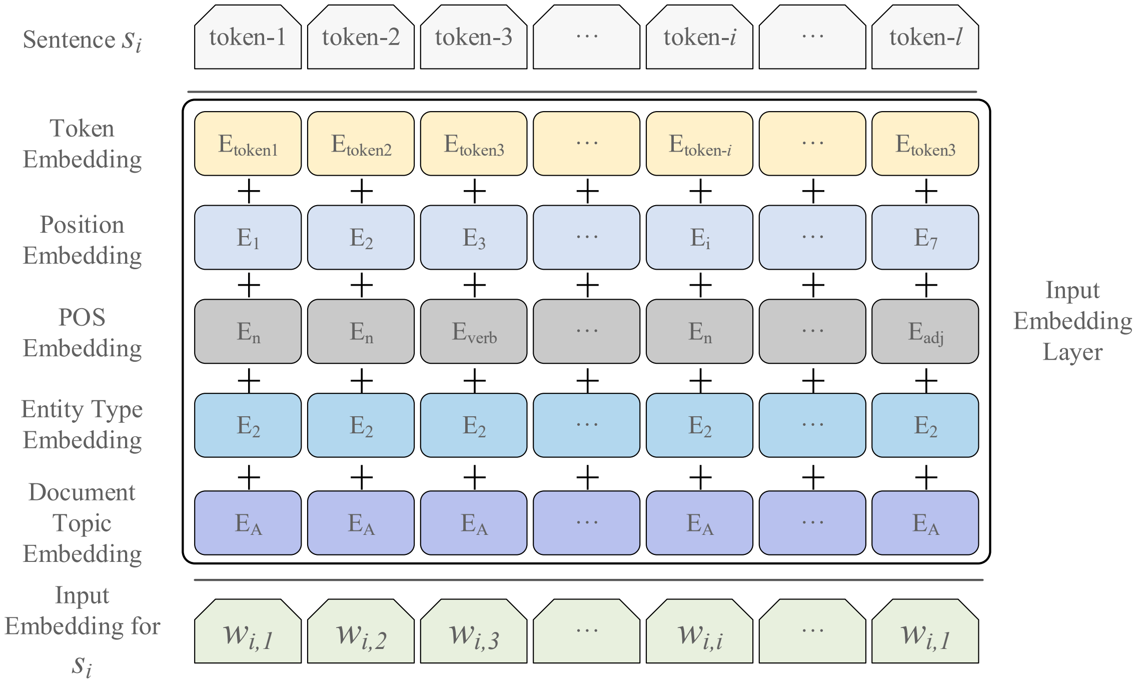


Figure 3. The input embedding layer of sentence-level encoder. The input embeddings for each token are the sum of the token, position, POS, entity type, and the document topic embeddings.

4.3. Document-Level Encoder

Document-level contextualized representations for entities and sentences take advantage of the global semantic of the document and can benefit DEE.

Entity & Sentence Embedding. To acquire the sentence-level representations of entities, we aggregated all word representations by performing a *mean pooling* operation over consecutive tokens that mention e_i. Specifically, given entity e_i and its span in sentence S_i, which starts at the kth token and ends at the tth token, the representation for e_i can be represented as:

$$c_i^e = MeanPool(c_{i,k}, \cdots, c_{i,t}) \tag{3}$$

where $MeanPool(\cdot)$ denotes the *mean pooling* operation and $c_i^e \in R^d$ denotes the representation for e_i. The variables $c_{i,k}, c_{i,t} \in R^d$ are tensors for the start token and end token of entity e_i, which were used to calculate the sentence-level representation for e_i. Similarly, we performed a *mean pooling* operation for all entities contained in document $\mathcal{D}$, then obtained a series of sentence-level entity representations $[c_1^e, c_2^e, \cdots, c_{N_e}^e]$, with N_e being the number of entities contained in document $\mathcal{D}$.

For sentences, we also performed a *mean pooling* operation over the tokens covering a sentence and obtained the sentence-level sentence representations $[c_1^s, c_2^s, \cdots, c_{N_s}^s]$, where $c_i^s \in R^d$ denotes the representation for sentence S_i.

Document-level Encoding. To obtain the document-level contextualized representations for all sentences and entities, we applied a Transformer encoder, *TF*-2, to model the information interaction between them, enabling the awareness of document-level contexts. Following Doc2EDAG [8], we added sentence position embeddings to the sentence representation to inform the sentence order in the given document before feeding them into *TF-2*. Document-aware representations could be obtained through document-level encoding:

$$[H^e, H^s] = TF - 2\left(c_1^e, \cdots, c_{N_e}^e; c_1^s, \cdots, c_{N_s}^s\right) \tag{4}$$

where *TF-2*$(\cdot)$ is used to obtain the document-level representations for entities and sentences contained in the given document, which takes the sentence-level representations of entities and sentences as inputs. The variable $H^e \in R^{N_e \times d}$ denotes document-level entity representation, and $H^s \in R^{N_s \times d}$ denotes document-level sentence representation. As entities may be mentioned by different word spans in a document, we performed a *max pooling* operation over all the mention embeddings referring to the same entity to obtain a fused embedding. Then we acquired the distinct document-aware context representation $H^e \in R^{N_e^{'} \times d}$.

In this study, we performed event type classification tasks at the document-level encoding stage. We first applied a *mean pooling* operation over H^s to obtain the document representation H^d, and then fed H^d into multiple feed-forward networks (FFN) to perform event type predictions. Concretely, the event t can be predicted by:

$$p_t = softmax\left(H^d \cdot W_e^t\right) \tag{5}$$

where p_t indicates the probability for event type t, which is calculated by the *softmax*$(\cdot)$ function. The variable $W_e^t \in R^{d \times 2}$ represents learnable parameters for predicting the event type t. Variables $t \in \mathcal{T}$ and $\mathcal{T}$ denote the pre-defined event type set.

For training *TF*-2, we minimized the following loss function:

$$\mathcal{L}(event) = -\sum_{t=1}^{|\mathcal{T}|}\left(\mathrm{I}\left(\hat{R}_t = 1\right)\log p_t + \mathrm{I}\left(\hat{R}_t = 0\right)(1 - \log p_t)\right) \tag{6}$$

Equation (6) represents the application of multiple binary classification tasks to construct the loss function, where $\hat{R}_t$ represents whether event type t is contained in the document, which is usually seen as *Golden Label*. $\mathrm{I}(\cdot)$ is the indicator function.

4.4. Sentence-Document Feature Fusion

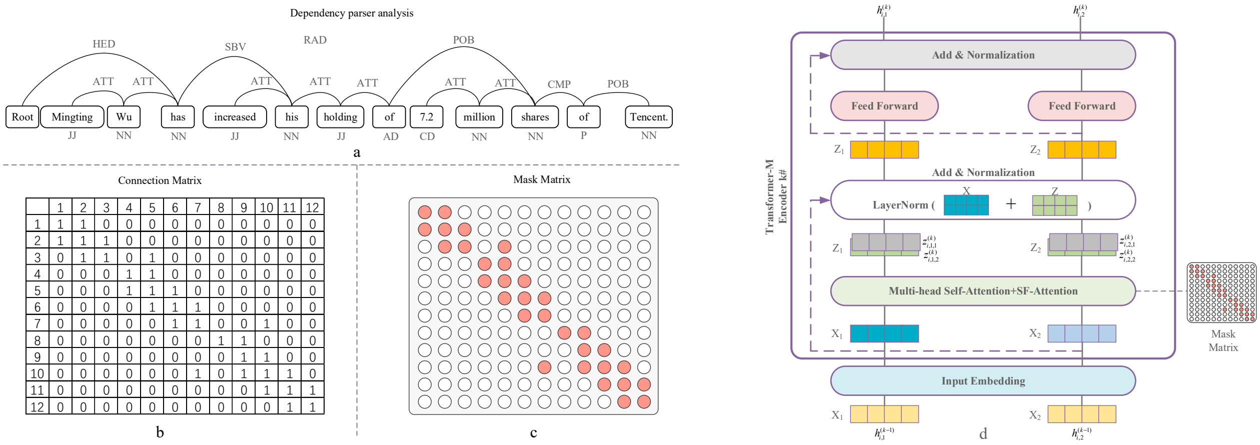
At this stage, we introduced a novel network to enhance document-aware representations for entities by integrating the sentence-document features. We stacked an advanced Transformer encoder, named *Transformer-M*, and Transformer encoder, *TF*-3, together to achieve this goal, where *Transformer-M* helped to extract more sentence-level features and the *TF*-3 was applied to integrate different kind features. Figure 2 depicts the novel network architecture.

Transformer-M. To explicitly model the syntactic features and aggregate sentence-aware and document-aware representations for each entity, we designed a novel encoder based on Transformer, named *Transformer-M* (*TF-M*). Figure 4d depicts the network architecture of *TF-M*. We added a Syntactic Feature Attention mechanism, *SF-ATT*, to Transformer, which was used to capture the syntactic features of the input sentences and enhance the representations for each token contained in the sentences. In addition, we concatenated the embeddings of each token used in *TF*-1 and the document-level representation of S_i as the input of *TF-M*, which helped us to make full use of the information of both the document-level and local sentence to model the long dependency. Specifically, the input embedding of the *j*th token can be calculated as:

$$w_{i,j}^{'} = \left[h_i^s; w_{i,j}\right] \tag{7}$$

where $w_{i,j} \in R^d$ is the j^{th} token input embedding used in *TF*-1; $h_i^s \in R^d$ is the document-level representation for S_i; and $w_{i,j}^{'} \in R^{2d}$ represents the input embedding vector of the j^{th}

token for S_i, which is fed into TF-M to obtain feature enhanced representation. $[\cdot\,;\,\cdot]$ denotes the concatenation operation.



	1	2	3	4	5	6	7	8	9	10	11	12
1	1	1	0	0	0	0	0	0	0	0	0	0
2	1	1	1	0	0	0	0	0	0	0	0	0
3	0	1	1	0	1	0	0	0	0	0	0	0
4	0	0	0	1	1	0	0	0	0	0	0	0
5	0	0	0	1	1	1	0	0	0	0	0	0
6	0	0	0	0	1	1	1	0	0	0	0	0
7	0	0	0	0	0	1	1	0	0	1	0	0
8	0	0	0	0	0	0	0	1	1	0	0	0
9	0	0	0	0	0	0	0	0	1	1	0	0
10	0	0	0	0	0	0	1	0	1	1	1	0
11	0	0	0	0	0	0	0	0	0	1	1	1
12	0	0	0	0	0	0	0	0	0	0	1	1

Figure 4. The overall workflow of Syntactic Feature Mask matrix construction and the architecture of *Transformer-M* encoder. (**a**) We first use the dependency parser tool (Stanford core NLP) to perform dependency analysis and obtain the dependency tree; (**b**) with the dependency tree, we can quickly obtain the connection matrix for the input sentences; and (**c**) we then obtain the mask matrix; (**d**) the overview of the Transformer-M is shown in (**d**), where the key component is the syntactic feature attention, which is designed to model the syntactic features of the input sentences.

As shown in Figure 4a, for the given document and sentences $\mathcal{D} = \{S_i\}_{i=1}^{N_s}$, we first applied the Spacy tool to construct the dependency relation for each token $w_{i,j} \in S_i$ and its parent. We were then able to obtain the dependency paths and the link matrix could be constructed with them. Subsequently, we were able to construct a syntactic feature mask matrix for the input sentence, which was used in *TF-M*. To be specific, in the *i*th layer of *TF-M*, the Syntactic Feature Attention and Multi-head Attention (*MH-ATT*) hidden states were taken together to enhance the representations for each token in the input sentence; the former mainly focuses on the syntactic relations of $w_{i,j} \in S_i$ and its parent, and the latter is responsible for modelling the whole lexical and semantic features for all tokens in the sentence, which can be formalized as follows:

$$z_{i,j,1}^{(k)} = MH - ATT\left(h_{i,1}^{(k-1)}, \cdots, h_{i,l}^{(k-1)}\right) \tag{8}$$

$$z_{i,j,2}^{(k)} = SF - ATT\left(h_{i,1}^{(k-1)}, \cdots, h_{i,l}^{(k-1)}\right) \tag{9}$$

$$z_{i,j}^{(k)} = z_{i,j,1}^{(k)} + z_{i,j,2}^{(k)} \tag{10}$$

where $z_{i,j,1}^{(k)} \in R^d$ and $z_{i,j,2}^{(k)} \in R^d$ denote the hidden state calculated after the *k*th layer of MH-ATT and SF-ATT for the *j*th token of S_i, respectively. The variables $z_{i,j}^{(k)} \in R^d$ denote the hidden state calculated with the *k*th layer. Then $z_{i,j,1}^{(k)}$ and $z_{i,j,2}^{(k)}$. $z_{i,j}^{(k)}$ were integrated with the hidden state calculated by the MH-ATT to enhance the representation for the token $w_{i,j}$.

The inputs were fed into *SF-ATT* to capture their explicit syntactic features, and the calculation can be formalized as follows:

$$z_i = \sum_{j=1}^{l} \alpha_{i,j} \cdot \left(w_j \cdot W_v\right) \tag{11}$$

$$\alpha_{i,j} = \frac{exp\left(a_{i,j}\right)}{\sqrt{d} \cdot \sum_{k=1}^{l} exp\left(a_{i,j}\right)} \tag{12}$$

$$a_{i,j} = Masking\big((w_i \cdot W_q) \cdot (w_j \cdot W_k), M\big) \tag{13}$$

$$M_{i,j} = \begin{cases} 0, w_i \leftrightarrow w_j \\ -\infty, w_i - w_j \end{cases} \tag{14}$$

where $z_i \in R^d$ denotes the encoding vector of *SF-ATT* for token w_i, which corresponds to the $z^{(k)}_{i,j,2}$; l denotes the length of S_{i}; $\alpha_{i,j}$ and W_v are the attention weight of token w_j and the linear transformation of token embedding w_j, which are used to calculate the attention value for the token w_i. The *Masking* function in Equation (13) restrains the dependency relation among the input tokens. The equation indicates that only linked tokens and the current token itself are involved to update the token embedding, and this is controlled by the masking matrix $M_{i,j}$, where $\leftrightarrow$ represents that there is a dependency path between the ith and jth token, and $-$ represents none. Figure 4a illustrates the dependency paths for the input tokens. Similar to W_v, W_q and W_k are independent linear transformations for token embeddings.

Feature integration. As shown in Figure 2, the final hidden states of the *TF-M* and the document-aware representations of entities were fed into *TF*-3. Then we were able to obtain the multi-turn and multi-granularity representations for the entities. Formally, given a sentence S_{i}, *TF-M* encodes it to obtain $C^{'}_i = [c^{'}_{i,1}, c^{'}_{i,2}, \cdots, c^{'}_{i,l}]$, where $c^{'}_{i,j} \in R^d$, then $C^{'}_i$ and H^e are fed into *TF*-3, and then we can obtain the integrated representations $h_i^{e,fused} \in R^d$ for each recognized entity in the input document.

$$h_1^{e,fused}, \cdots, h_{N_e}^{e,fused} = TF - 3\Big(h_1^e, \cdots, h_{N_e}^e; c^{'}_{i,1}, \cdots, c^{'}_{i,l}\Big) \tag{15}$$

where $h_i^{e,fused} \in R^d$ denotes the multi-granularity encoded document-aware and local feature enhanced representation for entity e_i. As the enhanced representation could model more local features and long-distance dependency information, it would facilitate the event expanding, which will be discussed in Section 4.5. We were able to obtain fused representations $H^{e,fused} = \{h_{\mathrm{i}}^{\mathrm{e,fused}}\}_{i=1}^{Ne}$ for each token and entity in the document using Equation (15).

4.5. Event Expanding

In realistic scenarios, the number of event records described in a document is unknown in advance, thus we need to perform event detection as shown in Section 2.2 and then fill arguments for each specific event roles as pre-defined in the event table. In this work, we performed event expanding to fulfil the event roles filling task as previously described [8,16]. For each triggered event, the event expanding subtask can be formalized as a set of binary classification tasks, i.e., predicting to expand (1) or not (0) for all candidate entities. The expanded event path state information can distinctly guide the remaining event roles filling task; thus, an event memory mechanism is designed to memorize the extracted event paths and their arguments. To take advantage of the useful current states, such as the event path state, processed contexts, and event role, the memory tensor m and fused entity tensor $H^{e,fused}$ were concatenated, and then a trainable event role indicator embedding was added to form a new tensor for the next event role prediction. The tensor was fed into the fourth Transformer module, *TF*-4, to facilitate the event path prediction. Finally, the context-aware entity tensor e^r, the final hidden states of *TF*-4, was fed into a linear layer to conduct the event expanding classification. Figure 5 illustrates the event expanding process, which can be formalized as follows:

$$e^r = TF - 4\Big(\Big[m; H^{e,fused}\Big] + role_i\Big) \tag{16}$$

$$Label_{t,r} = softmax(e^r \cdot W_{role}) \tag{17}$$

where m denotes the memory tensor, which is used to integrate the current path and history contexts information. We initialized it with the document-level sentence tensor H^s and updated it when expanding by filling either the associated entity tensor or the zero-padded one for the NA argument. $e^r \in R^{N_e \times d}$ denotes the enriched entity representations, which were fed into a linear layer to conduct the path-expanding classification, and $W_{role} \in R^{d \times 2}$ is a learnable parameter for role prediction.

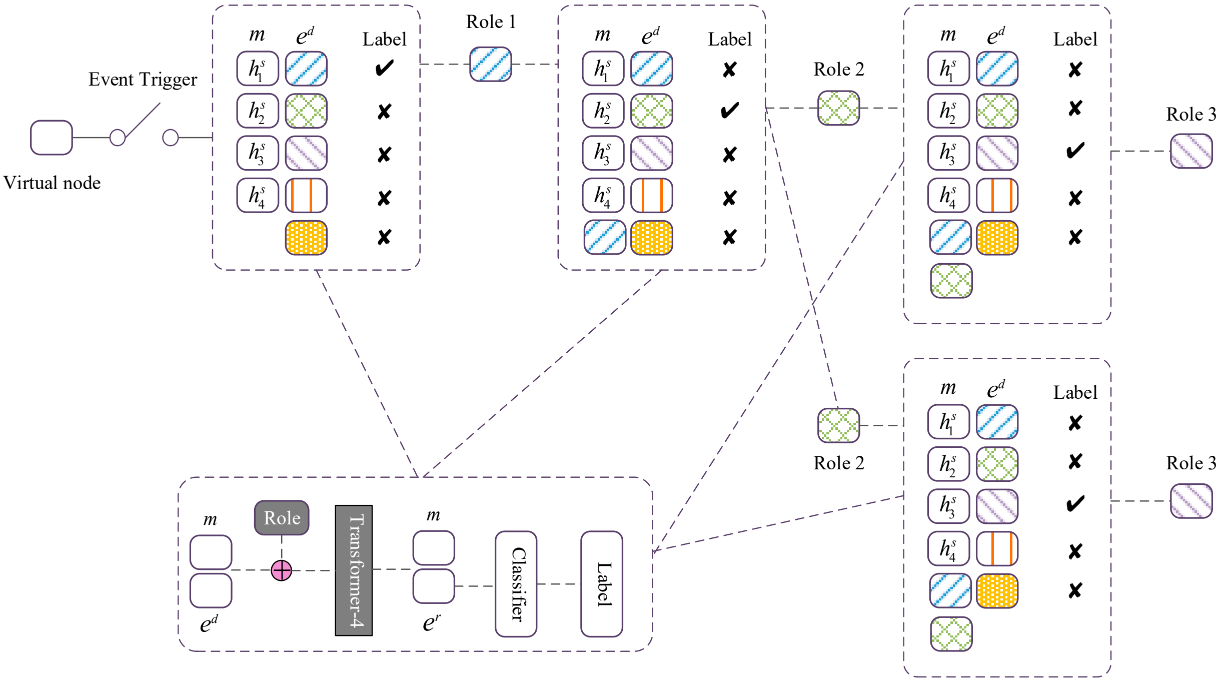


Figure 5. The overall workflow of event path expanding. For a triggered event, to enable the awareness of the current path state, history contexts and the current event role, we first concatenate the memory tensor m and the entity tensor e^d, then add them with a trainable event role indicator embedding, and encode them with the fourth Transformer module, TF-4, to facilitate the context-aware reasoning. Finally, we extract the enriched entity tensor e^r from outputs of TF-4 and stack a linear classifier over e^r to conduct the path-expanding classification.

4.6. Model Training

For event expanding, we calculated a cross-entropy loss for each prediction subtask, then summed these losses as the final event record extraction $\mathcal{L}(record)$, which can be calculated as:

$$\mathcal{L}(record) = \sum_{t=1}^{N_t} \sum_{r=1}^{N_{t,r}} CrossEntropy(Goden\_Lable_{t,r}, Label_{t,r}) \tag{18}$$

where $Goden\_Lable_{t,r}$ denotes the golden label for predicting role type r of event t, N_t is the number of events detected from the input document, and $N_{t,r}$ is the number of role types pre-defined for event t.

During training, we summed the entity recognition loss, event detection loss, and event expanding loss with different weights as the final loss $\mathcal{L}_{all}$:

$$\mathcal{L}_{all} = \lambda_1 \mathcal{L}(ner) + \lambda_2 \mathcal{L}(event) + \lambda_3 \mathcal{L}(record) \tag{19}$$

where λ_1, λ_2, and λ_3 are hyper-parameters. At inference, for a given document, we first recognized all the entities and then detected event types. Finally, we constructed event records with extracted entities and triggered event types.

5. Experiments and Analysis

In this section, we introduce the implementation details of DEEDP, and verify its effectiveness on two DEE benchmarks. Additionally, we perform ablation study on every strategy and novel architecture proposed in this paper.

5.1. Datasets and Evaluation Metrics

We compared the performance of DEEDP with baselines on the following two DEE benchmarks:

MUC-4. The MUC-4 dataset contains 1700 English documents and is mainly concerned with terrorist events. We followed the standard dataset division and set the number of documents for the train/dev/test set to 1300/200/200. Following previous work, we reported results on MUC-4 with Precision (P), Recall (R), and F1-score for the macro average for all event roles.

ChFinAnn. ChFinAnn is developed by [8], and widely used for evaluating DEE models. ChFinAnn is constructed by using 32,040 Chinese financial documents. It mainly concerns the following event types: Equity Freeze (EF), Equity Repurchase (ER), Equity Underweight (EU), Equity Overweight (EO), and Equity Pledge (EP). It is a challenging task because (1) there are 35 different event role types in total, (2) about 29% documents contain multiple events, and (3) the arguments for an event record can be involved in about six sentences. For more information, please refer to Doc2EDAG. To compare with previous SOTA models [8,12,16], we adopted the same evaluation metrics used in those models. Specially, for each given document, we selected, without replacement, one most similar ground-truth event record for each predicted event record to calculate role-level Precision (P), Recall (R), and F1 score for each event type. Because an event type usually consists of multiple roles, we calculated the micro-averaged scores of the role-level and took them as the final DEE metric.

5.2. Implementation Details

For all documents to be processed, we denoted the maximum number of sentences as N_s and the maximum sentence length as l and set them to 64 and 128, respectively. We employed the basic Transformer, hidden size set to 768, and attention heads were set to 12 for each layer, as the encoder architecture for the sentence-level encoder, document-level encoder, sentence-document features fusion encoder and event expanding encoder. During training, we set $\lambda_1 = 0.05$ and $\lambda_2 = \lambda_3 = 0.95$, as suggested by [8], and adopted the AdamW optimizer [34] with the Dropout 0.1 and learning rate 2×10^{-5}. Notably, as the training datasets for both the benchmarks are not very large, we set the training epoch to [80, 100, 150, 200] to prevent the model from overfitting, and we found that the model achieved the best performance when the epoch was set to 100. As a result, we trained the model for at most 100 epochs and saved the parameters that helped to achieve the best validation score on the development set. The detailed hyper-parameters are listed in Table 1.

Table 1. The hyper-parameter setting.

Hyper-Parameter	Value	Hyper-Parameter	Value
Number of generated events	5	Layers of *TF-M*	4
Embedding size	768	Learning rate for *TF*-(1~4)	2×10^{-5}
Hidden size	768	learning rate for TF-M	1×10^{-5}
Tagging scheme	BIO (Begin, Inside, Other)	Optimizer	AdamW
Layers of *TF*-1	4	λ_1	0.05
Layers of *TF*-2	4	λ_2, λ_3	0.95
Layers of *TF*-3	4	Dropout rate	0.1
Layers of *TF*-4	4	Training epoch	100

5.3. Results on MUC-4

Baselines. We conducted a comparison experiment to verify the effectiveness of our method on the MUC-4 with the following baselines: pipeline-based or manual feature engineering based methods. GLACIER [35] applied a sentential event recognizer to decide which sentences should be selected for event extraction and then used multiple plausible role filler recognizers to extract arguments as role fillers for formulating event records. TIER [36] proposed a multi-stage approach for event extraction, which divided the processing procedure into the following: identifying narrative document, detecting event sentence, and noun phrase analysis. Cohesion Extract [37] first performed event arguments recognition in the processing text and took them as candidate role fillers, and then a cohesion sentence classifier was applied to refine the candidate set. MGR [38] formulized the DEE as a neural seq-to-seq learning task and proposed a multi-granularity reader framework to dynamically aggregate semantic information from local sentence to the whole document by applying the advanced pre-trained language model BERT [29]. MMR [16] proposed a MRC based event extraction framework for DEE, which can directly extract event records in a machine reading manner without performing sentence-level event arguments extraction.

Main Results. Table 2 presents the main results on the benchmark MUC-4. From the comparison results, we can observe the following: (1) compared with other baselines, MMR achieved significant improvements, proving that the multi-granularity encoder benefits DEE; and (2) our proposed method, DEEDP, achieved +1.77 and +0.84 F1-scores over MMR for the head noun and extract match, respectively. This can be attributed to the semantic enhancing strategy introduced by using *TF-M* encoder.

Table 2. Overall precision (P), recall (R) and F1 scores evaluated on the MUC-4 test set.

Models	**Head Noun Match**			**Extract Match**		
	P (%)	**R (%)**	**F1 (%)**	**P (%)**	**R (%)**	**F1 (%)**
GLACIER	47.80	57.20	52.08	-	-	-
TIER	50.80	61.40	55.60	-	-	-
Cohesion Extract	57.80	59.40	58.59	-	-	-
MGR	56.44	**62.77**	59.44	52.03	**56.81**	54.32
MMR *w/o contextualized embedding*	63.95	58.71	61.19	60.66	55.34	57.87
DEEDP	**65.42**	61.57	**62.96**	**61.34**	56.29	**58.71**

In short, DEEDP effectively handles the long-distance dependencies problem between arguments by using a multi-granularity encoder, which helps in tackling the challenge of arguments-scattering for DEE. In addition, DEEDP introduces a novel Syntactic Feature Attention mechanism to explicitly model the semantic information of interdependency between tokens, which helps enhance the document-aware representations for each entity and improves the performance on document-level role filling tasks.

5.4. Results on ChFinAnn

Baselines. We examined the effectiveness of DEEDP and the following baselines on the ChFiAnn dataset: DCFEE [9] proposed a key event detection model for discovering event mentions and a strategy for argument completion. Two variants were developed for this model, i.e., DEFEE-O and DCFEE-M. DCFEE-O is a simple version that only produces one event record from a document, while DCFEE-M produces multiple ones. Doc2EDAG is an end-to-end model proposed by Zheng et al. [8]. Doc2EDAG combines different Transformers [14] to perform event and event arguments recognition and transfers the DEE as a table filling task. To verify the necessity of end-to-end, a greedy baseline of Doc2EDAG, named GreedyDec, was proposed, which only applies recognized entity roles to fill one event table entry for a document. DE-PPN is an encoder-decoder model for DEE proposed by Yang et al. [12]. DE-PPN obtains document-aware representations by using a document-level Transformer encoder, and then decodes using multi-granularity non-autoregressive decoders in parallel. Similar to Doc2EDAG, there is also a single event version of DE-PPN, DE-PPN-1. MMR [16] formulated the paradigm of DEE as a MRC task, and proposed a multi-turn and multi-granularity reader for DEE, which can help extract events from given documents directly without conducting SEE. In addition, there is a simple one-record decoding baseline of MMR, MMR-1. For fairness, we also present a simple decoding version of DEEDP, DEEDP-O, which produces only one event record with the recognized entity roles.

Main Results. Table 3 presents the overall performance for DEEDP and the baselines on the benchmark ChiFiAnn. As we can see, the proposed method, DEEDP, consistently outperforms all baselines and improves the F1-score by 0.4, 3.7, 2.2, 1.4, and 0.5 over the SOTA method, DE-PPN, on EF, ER, EU, EO, and EP, respectively. These vast improvements benefit from the multi-granularity encoder and explicitly model syntactic features of the input sentences at the global-to-local feature aggregation stage. Because the multi-granularity encoder applies more encoders to model the interdependency between sentence-level and document-level features, and the strategy of explicitly modeling syntactic features helps to capture more sentence-level features of the input sentences, both of these strategies facilitate the DEE task. Additionally, the simple decoding baseline, DEEDP-O, achieves the best performance compared with other simple baselines, such as DCFEE-O, GreedyDec, DE-PPN-1, and MRR-1, which also demonstrates the advantages of our proposed strategies.

Table 3. Overall event-level precision (P), recall (R), and F1 scores evaluated on the ChFinAnn test set.

Models	**EF**			**ER**			**EU**			**EO**			**EP**		
	P (%)	**R (%)**	**F1 (%)**	**P (%)**	**R (%)**	**F1 (%)**	**P (%)**	**R (%)**	**F1 (%)**	**P (%)**	**R (%)**	**F1 (%)**	**P (%)**	**R (%)**	**F1 (%)**
DCFEE-O	66.0	41.6	51.1	84.5	81.8	83.1	62.7	35.4	45.3	51.4	42.6	46.6	64.3	63.6	63.9
DCFEE-M	51.8	40.7	45.6	83.7	78.0	80.8	49.5	39.9	44.2	42.5	47.5	44.9	59.8	66.4	62.9
GreedyDec	79.5	46.8	58.9	83.3	74.9	78.9	68.7	40.8	51.2	69.7	40.6	51.3	**85.7**	48.7	62.1
Doc2EDAG	77.1	64.5	70.2	91.3	83.6	87.3	80.2	65.0	71.8	82.1	69.0	75.0	80.0	74.8	77.3
DE-PPN-1	77.8	55.8	64.9	75.6	76.4	76.0	76.4	63.7	69.4	77.1	54.3	63.7	85.5	43.0	57.2
DE-PPN	78.2	**69.4**	73.5	89.3	85.6	87.4	69.7	**79.9**	74.4	81.0	71.3	75.8	83.8	73.7	78.4
MMR-1	81.2	48.7	60.9	82.9	73.2	77.8	81.2	45.1	58.1	75.5	45.8	57.0	84.3	50.8	63.4
MMR	78.4	65.5	71.3	89.3	88.1	88.7	79.5	66.4	72.4	83.5	71.4	76.9	82.3	74.1	78.0
DEEDP-O	**83.5**	50.1	62.9	77.3	65.2	70.7	80.5	58.7	67.9	**84.1**	62.6	71.8	84.9	54.3	66.2
DEEDP	79.6	68.9	**73.9**	**92.5**	**89.7**	**91.1**	**82.1**	71.8	**76.6**	81.3	**73.5**	**77.2**	82.6	**75.5**	**78.9**

5.5. Results on Single-Event and Multi-Event

Following previous works [8,11,12], and we performed experiments on a single-record set (S.), in which only one event is described in a document, and a multi-record set (M.), in which multi event records are embedded in a document. In other words, single-record set refers to documents in this set containing only one event record, and multi-record set refers to documents that contain multiple event records. The comparison results for different scenarios are listed in Table 4. We can observe that (1) it is a more challenging task for multiple event record extraction compared with single-record event extraction, and as a result, the extraction performance of all models drops significantly; (2) however, our model, DEEDP, still achieves a 23.2, 2.2, 2.6, 3.6 average Micro-F1 score over DCFEE-M, DE-PPN, MMR, and Doc2EDAG, respectively; (3) DEEDP-O outperforms one event record baselines on the single-record set, demonstrating the effectiveness of the strategies proposed in this work; (4) but DEEDP-O only produces one event record and, as a result, it works poorly in the scenario of a multi-record set, and the performance decreases significantly compared with testing on the single-event set.

In conclusion, we find that a multi-granularity encoder can model more contextualized information for multi-record event extraction than models that apply only sentence-level and document-level encoders, such as Doc2EDAG and DE-PPN. In addition, explicitly modeling the syntactic features of the input sentences can compensate or enhance some features discarded in the mean pooling or max pooling operations, benefiting information aggregation in the sentence-document feature fusion stage. Thus DEEDP achieves a higher F1 score than MMR.

Table 4. F1 scores for each pre-defined event type and the averaged ones (Avg.) on single-event (S.) and multi-event (M.) sets.

Models	**EF**		**ER**		**EU**		**EO**		**EP**		**AVG**		
	S. (%)	**M. (%)**	**S. (%)**	**M. (%)**	**S. (%)**	**M. (%)**	**S. (%)**	**M. (%)**	**S. (%)**	**M. (%)**	**S. (%)**	**M. (%)**	**S.&M. (%)**
DCFEE-O	56.0	46.5	86.7	54.1	48.5	41.2	47.7	45.2	68.4	61.1	61.5	49.6	58.0
DCFEE-M	48.4	43.1	83.8	53.4	48.1	39.6	47.1	42.0	67.0	60.0	58.9	47.7	55.7
GreedyDec	75.9	40.8	81.7	49.8	62.2	34.6	65.7	29.4	88.5	42.3	74.8	39.4	60.5
Doc2EDAG	80.0	61.3	89.4	68.4	77.4	64.6	79.4	69.5	85.5	72.5	82.3	67.3	76.3
DE-PPN-1	82.4	46.3	78.3	53.9	82.2	45.6	78.1	39.3	82.8	38.5	80.7	44.7	66.2
DE-PPN	82.1	63.5	89.1	70.5	79.7	66.7	80.6	69.6	88.0	73.2	83.9	68.7	77.9
MMR-1	79.2	52.1	88.2	49.3	70.3	49.4	74.2	44.7	87.4	45.2	79.8	48.1	63.4
MMR	81.2	61.8	89.8	70.1	77.9	65.4	80.8	71.7	86.2	72.6	83.2	68.3	77.4
DEEDP-O	80.2	52.3	79.4	51.3	78.6	48.6	76.4	45.5	86.9	47.9	80.3	49.1	67.8
DEEDP	**82.8**	**66.4**	**93.6**	**73.7**	**82.4**	**68.4**	**81.7**	**71.9**	**87.5**	**74.1**	**85.6**	**70.9**	**79.5**

5.6. Ablation Study

In this section, we describe the ablation tests performed to investigate the effectiveness of the critical designs of DEEDP. To address this issue, we constructed the following variants of DEEDP: (1) w/o RicFea, the input embedding at the sentence-level encoding stage without using the document topic, POS, and entity type features; (2) w/o DocEnc, the input embedding at the sentence-document feature fusion stage without using document-aware sentence representations; (3) w/o SF-ATT, removing Syntactic Feature Attention from the Transformer; and (4) w/o TF-M, removing *TF-M* encoder. Table 5 presents the result of ablation tests on DEEDP, and we have the following observations: (1) compared with only use token embedding and positional embedding, richer input features inject more useful external features to enhance the semantic information for each token of the input sentences, benefiting the following tasks accordingly, and a +0.7 F1-score on average contribution is obtained by using the information of these rich features; (2) the use of document-aware sentence representations contributes a +1.4 F1-score on average compared to when they are not used; (3) the *SF-ATT* strategy that explicitly models the syntactic information of input sentences facilitates enhancing the semantic information for each document-aware representations of each entity, hence making great contributions, improving by 2.4 F1-score points on average; (4) the *TF-M* is a pivotal design for DEE that helps to achieve a +2.7 F1-score improvement on average, indicating that the *TF-M* can aggregate sentence-aware and document-aware features to improve the performance of DEE.

Table 5. F1 scores of ablation tests on DEEDP variants for all event types and the averaged ones (Avg.).

Model	**EF (%)**	**ER (%)**	**EU (%)**	**EO (%)**	**EP (%)**	**AVG. (%)**
DEEDP	73.9	91.1	76.6	77.2	78.9	79.5
w/o RicFea	−0.9	−0.7	−1.1	−0.4	−0.3	−0.7
w/o DocEnc	−2.3	−1.2	−1.9	−0.6	−0.8	−1.4
w/o SF-ATT	−3.1	−2.7	−3.8	−1.1	−1.4	−2.4
w/o TF-M	−2.9	−3.1	−4.2	−1.5	−1.7	−2.7

5.7. Effect of Different Transformer-M Layers

In this section, we describe experiments performed to verify the importance of the feature integration encoder by setting different layers of *TF-M* encoder. To be specific, we set the layers of the *TF-M* encoder to {0, 1, 2, 4}, where 0 represents that the *TF-M* encoder is to be removed. The effect of different *TF-M* encoder layers is depicted in Figure 6, and we can observe that the best average F1-score is achieved for our method when we set the number of layers to 4. We conjecture that stacking more layers of the *TF-M* encoder facilitates better aggregation of sentence-aware and document-aware information.

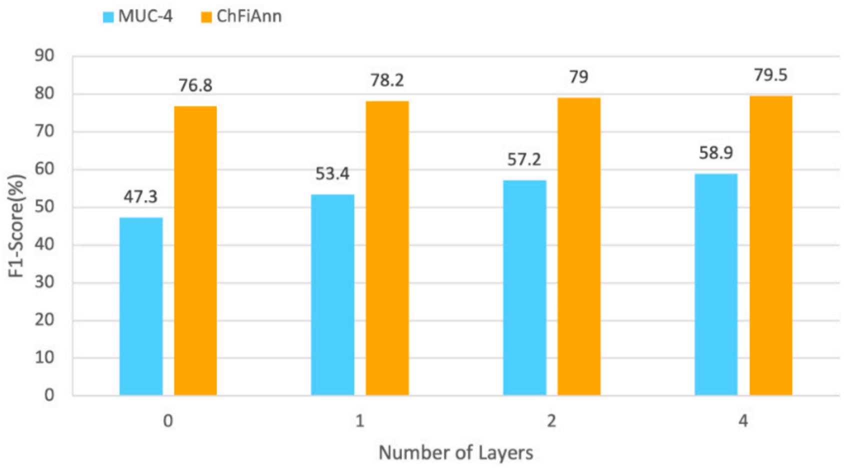


Figure 6. F1-score for performance with different *TF-M* encoder layers.

6. Discussion

Experimental results and ablation studies have demonstrated the superiority of DEEDP, but we wanted to determine the detailed advantages and disadvantages of our model. Table 6 presents the results of this comparison.

Table 6. Model comparison results. T: Token, P: Position, POS: Part-of-Speech, ET: Entity Type, DT: Document Topic, HGI: Heterogeneous Graph Interaction, SVM: Support Vector Machine, TF: Transformer, TF-M: Transformer-M, GNN: Graph Neural Network.

Model	Paradigm	Techniques	Input Features	Trigger Detection	Distant Dependency	Feature Enhanced
GLACIER	pipeline	Bayes, SVM	lexical, syntactic, entity	Yes	-	-
TIER	pipeline	SVM	lexical, semantic	Yes	-	-
Cohesion Extract	pipeline	SVM	lexical, discourse, roll filler distribution	Yes	-	-
MGR	joint	BERT, LSTM, CRF	T, P	No	-	-
DCFEE	pipeline	LSTM, CRF	T, P	Yes	-	-
Doc2EDAG	joint	TF	T, P	No	-	-
DE-PPN	joint	TF	T, P	No	-	-
MMR	joint	BERT	T, P	No	global-to-local	-
GIT	joint	TF, GNN, CRF	T, P	No	HGI	-
DEEDP	joint	TF, TF-M	T, P, POS, ET, DT	No	global-to-local	SF-ATT

We compared DEEDP with the most promising DEE models in terms of six metrics: paradigm, techniques, input features, trigger detection, distant dependency, and feature enhancement. From Table 6, we observe that (1) DEEDP is a joint DEE model that can avoid fault propagation. (2) It models more input features, such as POS, entity type, and document topic, and an ablation study has shown that richer input features can facilitate DEE. (3) MMR, GIT, and DEEDP can model long-distance dependency information, which is helpful for event expansion. However, compared to DEEDP, the dependency information modeled by MMR and GIT is limited because of the feature loss problem caused by the *mean pooling* or *max pooling* operation. (4) To address this problem, DEEDP proposes the use of Transformer-M, which adds a novel *SF-ATT* module to the transformer, which helps enhance the features. (5) Because DEEDP requires more input features, it is difficult to perform model migration.

7. Conclusions

In this study, we have proposed a multi-granularity DEE model, DEEDP, for tackling the troublesome challenges of arguments-scattering and multi-events. Multiple intrinsic experiments and ablation tests revealed that the *Transformer-M* module and the aggregation of global and local features can help improve the performance of DEE. These findings will serve as a basis for future studies and researchers can explore different feature fusion modules and multi-granularity feature extraction modules to further improve DEE performance. In future work, we will explore other feature enhancing modules and test our model on other domain-specific datasets to verify its performance.

Author Contributions: Conceptualization, H.L. and X.Z.; methodology, H.L. and X.Z.; software, L.Y.; validation, Y.Z., L.Y. and J.Z.; formal analysis, X.Z.; investigation, H.L. and Y.Z.; resources, X.Z. and L.Y.; data curation, Y.Z. and L.Y.; writing—original draft preparation, H.L.; writing—review and editing, X.Z. and Y.Z.; visualization, J.Z.; supervision, H.L.; project administration, X.Z.; funding acquisition, Y.Z. All authors have read and agreed to the published version of the manuscript.

Funding: The work of Yixin Zhao has been funded by National Natural Science Foundation, grant number 62102183; Jiangsu Province Natural Science Foundation, grant number BK20180462; and Fundamental Research Funds for the Central Universities, grant number 30920021141.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The MUC-4 and ChiFiAnn data used to support the findings of this study have been deposited in the repository https://github.com/mcclanelee/deedp, accessed on 10 January 2023.

Acknowledgments: We gratefully acknowledge the assistance of Ke Tu in preparing the pre-training corpus. We also thank Jiahao Zhou and Xiang Huang for providing language help.

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

References

1. Jin, Y.; Jiang, W.; Yang, Y.; Mu, Y. Zero-Shot Video Event Detection With High-Order Semantic Concept Discovery and Matching. *IEEE Trans. Multimed.* **2022**, *24*, 1896–1908. [CrossRef]
2. Zhang, J.; Hong, Y.; Zhou, W.; Yao, J.; Zhang, M. Interactive Learning for Joint Event and Relation Extraction. *Int. J. Mach. Learn. Cybern.* **2020**, *11*, 449–461. [CrossRef]
3. Li, P.; Zhou, G. Joint Argument Inference in Chinese Event Extraction with Argument Consistency and Event Relevance. *IEEE-ACM Trans. Audio Speech Lang.* **2016**, *24*, 612–622. [CrossRef]
4. Yu, W.; Yi, M.; Huang, X.; Yi, X.; Yuan, Q. Make It Directly: Event Extraction Based on Tree-LSTM and Bi-GRU. *IEEE Access* **2020**, *8*, 14344–14354. [CrossRef]
5. Li, L.; Jin, L.; Zhang, Z.; Liu, Q.; Sun, X.; Wang, H. Graph Convolution Over Multiple Latent Context-Aware Graph Structures for Event Detection. *IEEE Access* **2020**, *8*, 171435–171446. [CrossRef]
6. Zhang, J.; Huang, W.; Ji, D.; Ren, Y. Globally Normalized Neural Model for Joint Entity and Event Extraction. *Inf. Process. Manag.* **2021**, *58*, 102636. [CrossRef]
7. Doddington, G.R.; Mitchell, A.; Przybocki, M.A.; Ramshaw, L.A.; Strassel, S.M.; Weischedel, R.M. The Automatic Content Extraction (Ace) Program-Tasks, Data, and Evaluation. In *Proceedings of the Lrec*; European Language Resources Association (ELRA): Lisbon, Portugal, 2004; pp. 837–840.
8. 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.
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 the ACL 2018, System Demonstrations, Melbourne, Australia, 15–20 July 2018; pp. 50–55.
10. Sha, L.; Qian, F.; Chang, B.; Sui, Z. Jointly Extracting Event Triggers and Arguments by Dependency-Bridge RNN and Tensor-Based Argument Interaction. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32, pp. 5916–5923.
11. 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 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Online, 1–6 August 2021; Volume 1, pp. 3533–3546.
12. Yang, H.; Sui, D.; Chen, Y.; Liu, K.; Zhao, J.; Wang, T. Document-Level Event Extraction via Parallel Prediction Networks. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Online, 1–6 August 2021; Volume 1, pp. 6298–6308.
13. Balali, A.; Asadpour, M.; Campos, R.; Jatowt, A. Joint Event Extraction along Shortest Dependency Paths Using Graph Convolutional Networks. *Knowl.-Based Syst.* **2020**, *210*, 106492. [CrossRef]

14. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Long Beach, CA, USA, 2017; pp. 5999–6009.
15. Diao, Y.; Lin, H.; Yang, L.; Fan, X.; Wu, D.; Yang, Z.; Wang, J.; Xu, K. FBSN: A Hybrid Fine-Grained Neural Network for Biomedical Event Trigger Identification. *Neurocomputing* **2020**, *381*, 105–112. [CrossRef]
16. Yang, H.; Chen, Y.; Liu, K.; Zhao, J.; Zhao, Z.; Sun, W. Multi-Turn and Multi-Granularity Reader for Document-Level Event Extraction. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **2022**, *22*, 1–16. [CrossRef]
17. Li, Q.; Ji, H.; Huang, L. Joint Event Extraction via Structured Prediction with Global Features. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, 4–9 August 2013; Volume 1, pp. 73–82.
18. Li, X.; Nguyen, T.H.; Cao, K.; Grishman, R. Improving Event Detection with Abstract Meaning Representation. In Proceedings of the First Workshop on Computing News Storylines, Beijing, China, 31 July–1 August 2015; pp. 11–15.
19. Li, P.; Wang, Q. A Multichannel Model for Microbial Key Event Extraction Based on Feature Fusion and Attention Mechanism. *Secur. Commun. Netw.* **2021**, *2021*, e7800144. [CrossRef]
20. Feng, X.; Qin, B.; Liu, T. A Language-Independent Neural Network for Event Detection. *Sci. China Inf. Sci.* **2018**, *61*, 1–12. [CrossRef]
21. Trieu, H.-L.; Tran, T.T.; Duong, K.N.A.; Nguyen, A.; Miwa, M.; Ananiadou, S. DeepEventMine: End-to-End Neural Nested Event Extraction from Biomedical Texts. *Bioinformatics* **2020**, *36*, 4910–4917. [CrossRef]
22. Liu, X.; Luo, Z.; Huang, H. Jointly Multiple Events Extraction via Attention-Based Graph Information Aggregation. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 1247–1256.
23. Zhang, J.; He, Q.; Zhang, Y. Syntax Grounded Graph Convolutional Network for Joint Entity and Event Extraction. *Neurocomputing* **2021**, *422*, 118–128. [CrossRef]
24. Li, Q.; Peng, H.; Li, J.; Wu, J.; Ning, Y.; Wang, L.; Yu, P.S.; Wang, Z. Reinforcement Learning-Based Dialogue Guided Event Extraction to Exploit Argument Relations. *IEEE-ACM Trans. Audio Speech Lang.* **2022**, *30*, 520–533. [CrossRef]
25. Yang, B.; Mitchell, T. Joint Extraction of Events and Entities within a Document Context. *arXiv* **2016**, arXiv:1609.03632.
26. Liang, Y.; Jiang, Z.; Yin, D.; Ren, B. RAAT: Relation-Augmented Attention Transformer for Relation Modeling in Document-Level Event Extraction. *arXiv* **2022**, arXiv:2206.03377.
27. Liu, J.; Liang, C.; Xu, J. Document-Level Event Argument Extraction with Self-Augmentation and a Cross-Domain Joint Training Mechanism. *Knowl.-Based Syst.* **2022**, *257*, 109904. [CrossRef]
28. Li, J.; Hu, R.; Zhang, K.; Liu, H.; Ma, Y. DEERE: Document-Level Event Extraction as Relation Extraction. *Mob. Inf. Syst.* **2022**, *2022*, e2742796. [CrossRef]
29. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the NAACL HLT 2019—2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Minneapolis, MN, USA, 2–7 June 2019; Volume 1, pp. 4171–4186.
30. Li, H.; Yu, L.; Zhang, J.; Lyu, M. Fusion Deep Learning and Machine Learning for Heterogeneous Military Entity Recognition. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 1103022. [CrossRef]
31. Li, H.; Yang, X.; Zhao, X.; Yu, L.; Zheng, J.; Sun, W. MLRIP: Pre-Training a Military Language Representation Model with Informative Factual Knowledge and Professional Knowledge Base. *arXiv* **2022**, arXiv:2207.13929.
32. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Zettlemoyer, L.; Levy, O. Spanbert: Improving Pre-Training by Representing and Predicting Spans. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 64–77. [CrossRef]
33. 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; Volume 2: Short Papers, pp. 414–419.
34. Kingma, D.P.; Ba, J.L. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015; pp. 1–15.
35. Patwardhan, S.; Riloff, E. A Unified Model of Phrasal and Sentential Evidence for Information Extraction. In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Singapore, 6–7 August 2009; Association for Computational Linguistics: Cedarville, OH, USA, 2009; pp. 151–160.
36. 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: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; Association for Computational Linguistics: Cedarville, OH, USA, 2011; pp. 1137–1147.
37. Huang, R.; Riloff, E. Modeling Textual Cohesion for Event Extraction. In *Proceedings of the Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 2–9 February 2021*; pp. 1664–1670.
38. Du, X.; Cardie, C. Document-Level Event Role Filler Extraction Using Multi-Granularity Contextualized Encoding. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Online, 5–10 July 2020; pp. 8010–8020.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.