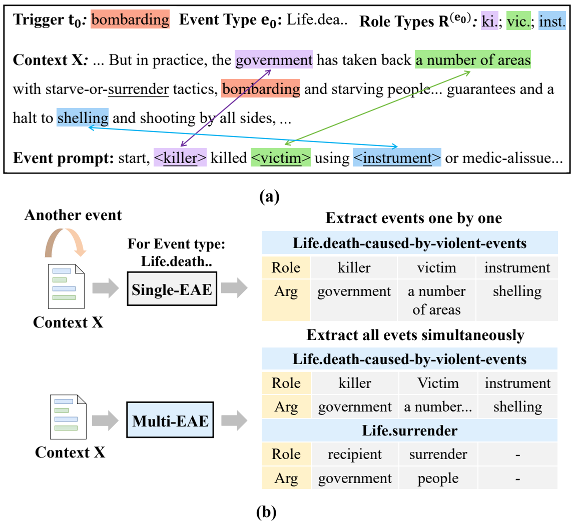
Beyond Single-Event Extraction: Towards Efficient Document-Level Multi-Event Argument Extraction

Wanlong Liu[1], **Li Zhou**[1], **Dingyi Zeng**[1], **Yichen Xiao**[1],
Shaohuan Cheng[1], **Chen Zhang**[2], **Grandee Lee**[3], **Malu Zhang**[1]*, **Wenyu Chen**[1]
[1]University of Electronic Science and Technology of China
[2] National University of Singapore
[3]Singapore University of Social Sciences
liuwanlong@std.uestc.edu.cn, maluzhang@uestc.edu.cn, cwy@uestc.edu.cn

Abstract

Recent mainstream event argument extraction methods process each event in isolation, resulting in inefficient inference and ignoring the correlations among multiple events. To address these limitations, here we propose a multiple-event argument extraction model DEEIA (***D****ependency-guided* ***E****ncoding and* ***E****vent-specific* ***I****nformation* ***A****ggregation*), capable of extracting arguments from all events within a document simultaneously. The proposed DEEIA model employs a multi-event prompt mechanism, comprising DE and EIA modules. The DE module is designed to improve the correlation between prompts and their corresponding event contexts, whereas the EIA module provides event-specific information to improve contextual understanding. Extensive experiments show that our method achieves new state-of-the-art performance on four public datasets (RAMS, WikiEvents, MLEE, and ACE05), while significantly saving the inference time compared to the baselines. Further analyses demonstrate the effectiveness of the proposed modules. Our implementation is available at https://github.com/LWL-cpu/DEEIA.



Trigger t_0: bombarding **Event Type e_0:** Life.dea.. **Role Types $R^{(e_0)}$:** ki.; vic.; inst.

Context X: ... But in practice, the government has taken back a number of areas with starve-or-surrender tactics, bombarding and starving people... guarantees and a halt to shelling and shooting by all sides, ...

Event prompt: start, <killer> killed <victim> using <instrument> or medic-alissue...

Extract events one by one

Life.death-caused-by-violent-events			
Role	killer	victim	instrument
Arg	government	a number of areas	shelling

Extract all evets simultaneously

Life.death-caused-by-violent-events			
Role	killer	Victim	instrument
Arg	government	a number...	shelling
Life.surrender			
Role	recipient	surrender	-
Arg	government	people	-

Figure 1: Subfigure (a) explains the prompt-based EAE task with one of the events in context X. The prompt is manually designed for the specified event type, with mentions of roles as **slots**, such as $\langle killer \rangle$. (b) shows the difference between traditional Single-EAE method and our Multi-EAE method, the latter is more difficult.

1 Introduction

Document-level event argument extraction (EAE) is a key process within Information Extraction (Hobbs, 2010; Grishman, 2015; Xia et al., 2022), focused on identifying event-related arguments and their respective roles in document-level texts. Recently, the leading-edge methods for this task delve into prompt-based techniques (Ma et al., 2022; Hsu et al., 2023), due to their great generalizability and competitive performance. Figure 1 (a) presents an example demonstrating EAE utilizing a prompt-based approach. Regarding the event e_0 triggered by “bombarding”, the approach defines a corresponding event prompt and identifies arguments: “government” as the *killer*, “a number of areas” as the *victim*, and “shelling” as the *instrument*.

Mainstream EAE works (Zhou et al., 2024; Liu et al., 2023c; Ren et al., 2023; Liu et al., 2023a) can only process one event at a time. When encountering documents containing multiple events, the limitations of these single-event argument extraction (Single-EAE) methods become evident. (1) As shown in Figure 1 (b), for a document containing multiple events, Single-EAE methods have to perform numerous iterations to extract event arguments for all events, which process the same document text repeatedly, leading to inefficient extraction. (2) Single-EAE methods fail to capture the beneficial event correlations among multiple events (He et al., 2023; Zeng et al., 2022b). Figure 1 (b) illustrates the argument overlapping phenomenon which reflects the semantic correlations among events. The event `Life.death` and event

*Corresponding author

Findings of the Association for Computational Linguistics: ACL 2024, pages 9470–9487
August 11-16, 2024 ©2024 Association for Computational Linguistics

`Life.surrender` share the argument "government" and there exists a strong event correlation between these two events. However, Single-EAE methods cannot utilize such correlations.

To tackle these limitations, this paper proposes a DEEIA (***D**ependency-guided **E**ncoding and **E**vent-specific **I**nformation **A**ggregation*) model, a multiple-event argument extraction (Multi-EAE) method capable of simultaneously extracting arguments for all events within the document. We construct our DEEIA model based on the state-of-the-art (SOTA) prompt-based Single-EAE model PAIE (Ma et al., 2022) and introduce a multi-event prompt mechanism to enable extracting arguments from multiple events simultaneously.

However, our Multi-EAE model faces the challenge of handling more complex information as it needs to simultaneously process different triggers, arguments roles, and prompts from multiple events. This requires the model with enhanced information extraction capabilities. Therefore, we design a Dependency-guided Encoding (DE) module to guide the model in correlating the various prompts with their respective event contexts. Furthermore, we propose an Event-specific Information Aggregation (EIA) module to provide event-specific contextual information for a better context understanding.

Figure 2 demonstrates that with the increase in the number of events within a document, the efficiency advantage of our DEEIA model becomes increasingly apparent. The performance surpassing Single-EAE baselines also demonstrates that our model effectively captures event correlations. The contributions of this paper are summarized as follows:

- We propose a multi-event argument extraction (Multi-EAE) method, capable of extracting the arguments of multiple events simultaneously, with the aim of improving the efficiency and performance of the EAE task.
- To tackle the challenges of Multi-EAE, we propose a Dependency-guided Encoding (DE) module and an Event-specific Information Aggregation (EIA) module, which provide dependency guidance and event-specific context information, respectively.
- Extensive experiments demonstrate that the proposed DEEIA model outperforms major benchmarks in terms of both performance and inference time. We provide comprehensive ablation studies and analyses.

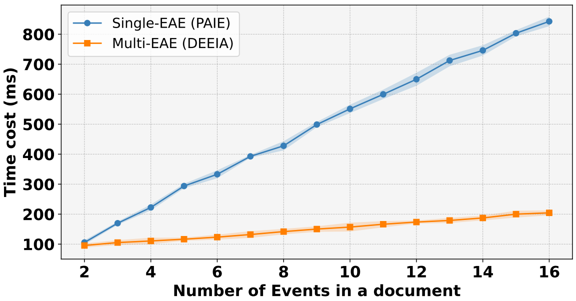


Figure 2: We select document samples containing different numbers of events and calculate the inference time on one sample for a Single-EAE method PAIE (Ma et al., 2022) and our Multi-EAE model DEEIA. The results are averaged on 100 repeated experiments. With the increase of event numbers within a document, the efficiency advantage of our Multi-EAE model becomes increasingly apparent.

2 Related Work

Recently, there has been an increasing interest in the task of document-level Event Argument Extraction (Wang et al., 2022; Liu et al., 2023b; Yang et al., 2023), a crucial component within the domain of Event Extraction (Ren et al., 2022; Yang et al., 2021). Current methods for document-level EAE can be classified into four main categories: (1) Span-based methods, which identify candidate spans and subsequently predict their roles (Zhang et al., 2020; Yang et al., 2023; Liu et al., 2017; Zhang et al., 2020; Liu et al., 2023c). (2) Generation-based methods (Li et al., 2021; Du et al., 2021; Wei et al., 2021; Huang et al., 2023) utilizing generative PLMs, such as BART (Lewis et al., 2019), to sequentially produce all arguments for the designated event. (3) Prompt-based methods (Ma et al., 2022; He et al., 2023; Nguyen, 2023; Wang et al., 2024), which use slotted prompts and leverage a generative slot-filling approach for argument extraction. (4) Large language model methods. Recently, some work (Zhang et al., 2024b; Zhou et al., 2023a) has attempted to explore to utilize large language models for EAE tasks, but the performance falls short of expectations. And the time and cost of inference are relatively high. Among them, prompt-based methods have been demonstrated superior generalizability and competitive performance (Hsu et al., 2023).

However, these EAE methods are Single-EAE methods, which can only process one event at a time. Recently, prompt-based EAE method, TabEAE (He et al., 2023) aims to capture event

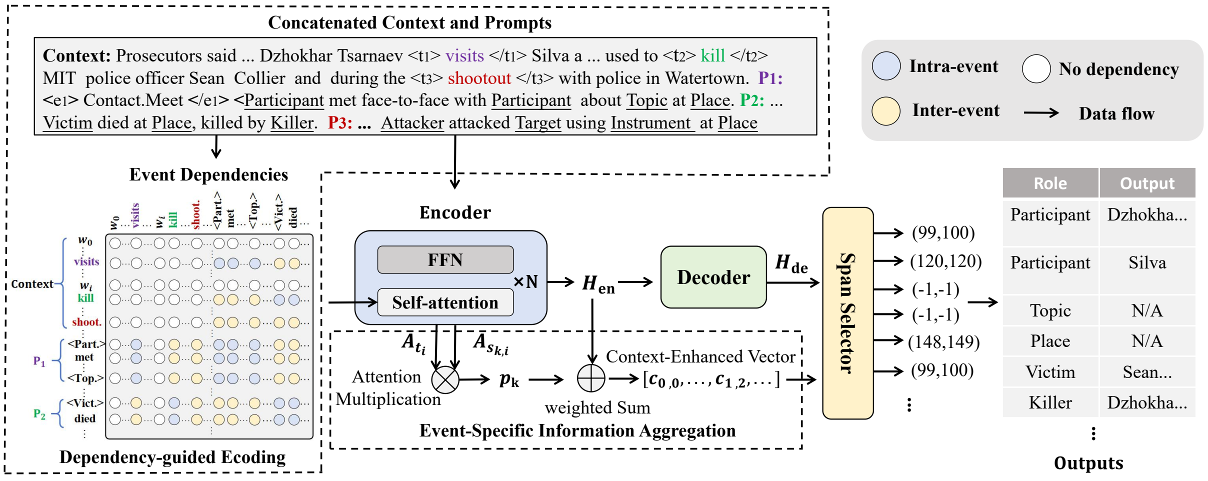

Concatenated Context and Prompts
Context: Prosecutors said ... Dzhokhar Tsarnaev <t1> visits </t1> Silva a ... used to <t2> kill </t2> MIT police officer Sean Collier and during the <t3> shootout </t3> with police in Watertown. P1: <e1> Contact.Meet </e1> <Participant met face-to-face with Participant about Topic at Place. P2: ... Victim died at Place, killed by Killer. P3: ... Attacker attacked Target using Instrument at Place
Intra-event
No dependency
Inter-event
Data flow
Event Dependencies
Context
P1
P2
Dependency-guided Ecoding
Encoder
FFN
Self-attention
×N
Hen
Decoder
Hde
Span Selector
Attention Multiplication
pk
weighted Sum
Context-Enhanced Vector
Event-Specific Information Aggregation
(99,100)
(120,120)
(-1,-1)
(-1,-1)
(148,149)
(99,100)
Role
Output
Participant
Dzhokha...
Participant
Silva
Topic
N/A
Place
N/A
Victim
Sean...
Killer
Dzhokha...
Outputs

Figure 3: The architecture of the proposed DEEIA model. For an input document, P_1, P_2, and P_3 represent simplified prompts.

co-occurrence (Zeng et al., 2022b) and trains the model on multi-event scheme. However, TabEAE requires separately processing prompts for different events, which is highly time-consuming. In this paper, we propose to extract event arguments concurrently, which significantly improves the efficiency of the EAE task in multi-event documents.

3 Methodology

In this section, we first provide a formal definition of multi-EAE task. Given an instance $(X, \{e_i\}_{i=1}^{K}, \{t_i\}_{i=1}^{K}, \{R^{(e_i)}\}_{i=1}^{K})$, where $X = (w_0, w_2, \dots, w_{N-1})$ represents the document text with N words, K is the number of target events, and e_i is the type of the i-th event. $t_i \subseteq X$ is the trigger word of the i-th event, and $R^{(e_i)}$ indicates the set of roles associated with event e_i. The task aims to extract a set of span $\mathcal{S}_i$ for each event e_i, which satisfies $\forall a^{(r)} \in \mathcal{S}_i, (a^{(r)} \subseteq X) \wedge (r \in R^{(e_i)})$. Most previous EAE methods are designed as Single-EAE methods, applicable when $K = 1$.

We then provide a detailed description of our DEEIA model, as shown in Figure 3. We first propose a multi-event prompt mechanism to enable extracting arguments from multiple events simultaneously (§ 3.1). Then we propose a Dependency-guided Encoding module (§ 3.2) and an Event-specific Information Aggregation module (§ 3.3) to tackle the complexity and challenge of simultaneously extracting arguments of multiple events.

3.1 Multi-event Prompt Mechanism

Multi-event document instances involve the prompts of multiple events. Therefore, we propose a multi-event prompt mechanism to enable extracting arguments from multiple events simultaneously. We first enhance the input text and event prompts and then concatenate them as the final input.

Preprocessed Text. Given an input text with a set of event triggers, each trigger is initially annotated with a unique pair of markers $(\langle t_i \rangle, \langle / t_i \rangle)$, where i counts the order of occurrence. Then we tokenize the marked text into X, where t_i is the i-th trigger:

$$\hat{X} = w_0 \, \langle t_0 \rangle \, t_0 \, \langle / t_0 \rangle \dots \langle t_i \rangle \, t_i \, \langle / t_i \rangle \dots w_N. \quad (1)$$

Prompt Enhancement. We concatenate the prompt of each event and obtain the final prompt P.[1] For each event prompt P_i, we utilize the event-schema prompts proposed by PAIE (Ma et al., 2022). We append the event type to the start of each corresponding prompt. This strategy helps the model distinguish different event prompts and integrates event type information, enriching the EAE process with more comprehensive information. Specifically, we wrap each event type with a unique pair of markers $(\langle e_i \rangle, \langle / e_i \rangle)$ and obtain the final P as follows:

$$P = \langle e_0 \rangle \, w_0^{e_0} \dots \langle / e_0 \rangle \, P_0 \dots \langle e_i \rangle \, w_0^{e_i} \dots \langle / e_i \rangle \, P_i, \quad (2)$$

where P_i is the prompt of the i-th event and $w_0^{e_i}$ is the first token of event type e_i.

Then we concatenate $\hat{X}$ and P and put them into our dependency-guided encoder (in § 3.2.2). For

[1] If multiple events are of the same type, we retain only one prompt and use it to predict arguments of all such events.

input sequences that exceed the maximum length[2], we employ a dynamic window (Zhou et al., 2021) technique to solve the problem, where the detailed algorithm is shown in .

3.2 Dependency-guided Encoding Module

3.2.1 Event Dependency Definition

Due to the fact that different events are associated with distinct trigger words, argument roles, and prompts, our model faces the challenge of information complexity (Bagga and Biermann, 1997; Li et al., 2023a) when processing multiple events simultaneously. Therefore, we propose to guide the model to associate the multi-event prompts with their corresponding event contexts with pre-defined dependencies. Considering that arguments can exhibit inter-event and intra-event relations within a context, we define the following two types of dependencies among triggers and prompts (including argument slots), which help to solve the information complexity problem of multiple events.

Intra-Event Dependency. (1) The connection between the trigger and prompt tokens within the same event. (2) The connection between prompt tokens within the same event prompt.

Inter-Event Dependency. (1) The connection between the trigger and prompt tokens of different events. (2) The connection between prompt tokens of different events.

These two dependencies not only reflect intra and inter-event relations within a multi-event context, but also establish interactions among triggers and argument slots, both within and across events.

Formally, for the input sequence $S = (x_1, x_2, ..., x_n)$ including triggers and prompts from multiple events, we introduce $D = \{dp_{ij}\}$ to represent such two dependencies, where $i, j \in \{0, 1, ..., n\}$ and $dp_{ij} \in \{\textit{Intra-event, Inter-event, NA}\}$ is a discrete variable denotes the dependency from x_i to x_j. *NA* indicates there is no dependency between x_i and x_j. Then the pre-defined event dependencies D will be integrated into the transformer to provide information guidance.

3.2.2 Dependency-guided Encoder

We improve vanilla self-attention mechanism (Vaswani et al., 2017) by adding a learnable attention bias, which integrates the event dependencies into the transformer.

For the input representation $\mathbf{x}_i \in \mathbb{R}^d$, it is first projected into query/key/value vector: $\mathbf{q}_i = \mathbf{x}_i\mathbf{W}_Q, \mathbf{k}_i = \mathbf{x}_i\mathbf{W}_K, \mathbf{v}_i = \mathbf{x}_i\mathbf{W}_V$. Then a learnable bias is added to the vanilla self-attention mechanism, which helps the model perceive dependency relations among multiple events. The attention score a_{ij} is produced as follows:

$$a_{ij} = \frac{\mathbf{q}_i\mathbf{k}_j^T}{\sqrt{d_k}} + \gamma \cdot bias_{ij}, \tag{3}$$

where $bias_{ij}$ is the learnable attention bias for the attention between tokens x_i and x_j. γ is a hyperparameter for adjusting the influence of the bias. d_k is the the hidden dimension of each attention head. Specifically, the attention bias depends on the dependency dp_{ij} and the context information, with specific parameters trained and then utilized in a compositional manner (Xu et al., 2021). We design the attention bias $bias_{ij}$ as follows:

$$bias_{ij} = \begin{cases} 0, & \text{if } dp_{ij} \textit{ is NA} \\ \frac{\mathbf{q}_i\mathbf{W}_{dp_{ij}}\mathbf{k}_j^T + b_{dp_{ij}}}{\sqrt{d_k}}, & \text{otherwise} \end{cases} \tag{4}$$

where $\mathbf{W}_{dp_{ij}} \in \mathbb{R}^{d_k \times 1 \times d_k}$ and $b_{dp_{ij}}$ are the trainable parameters corresponding to dependency dp_{ij}. The remaining operations are the same as transformer mechanism. Then we apply this mechanism to each layer of the encoder. Note that we do not apply this mechanism to the decoder layers, and a detailed analysis is provided in Appendix C.3.

3.3 Event-specific Information Aggregation

In this section, we design an event-specific information aggregation (EIA) module which adaptively aggregates useful information for specific events. We hope the model can make use of the context information relevant to the specific event and the argument when extracting an argument. Therefore, we consider using triggers (representing the event) and slots (representing the argument) to measure the relevance between the target argument and context (prompt) information.

Specifically, we utilize the attention heads of argument slots and their associated triggers, derived from the pre-trained transformer encoder, to calculate the attention product for the input sequence tokens (including both context and prompt). The dot product of attention is designed to measure the

[2] The length of event prompts is much shorter than that of the text, and only a very small number of docs in Wikievents and MLEE datasets exceed the length limit.

degree of association between the current event's argument and every token of input context.

We adopt an encoder-decoder architecture. The encoder is employed to encode the input text, while the decoder is tasked with deriving the event-oriented context and context-oriented prompt representation $\mathbf{H}_{\text{de}}$:

$$[\mathbf{A}; \mathbf{H}_{\text{en}}] = \text{Encoder}_s(S), \\ \mathbf{H}_{\text{de}} = \text{Decoder}(\mathbf{H}_{\text{en}}), \tag{5}$$

where the Encoder_s is the dependency-guided encoder and the Decoder is a transformer-based decoder. S is the input of the concatenation of context X and prompt P, and $\mathbf{A} \in \mathbb{R}^{H \times l_s \times l_s}$ is the multi-head attention matrix and $\mathbf{H}_{\text{en}}, \mathbf{H}_{\text{de}} \in \mathbb{R}^{l_s \times d}$. H is the attention head numbers and l_s is the length of input sequence S.

For the k-th argument slot $s_{k,i}$ to be predicted in the i-th event, we first get the contextual attention vectors $\mathbf{A}_{t_i} \in \mathbb{R}^{l_s}$ and $\mathbf{A}_{s_{k,i}} \in \mathbb{R}^{l_s}$ from $\mathbf{A}$, corresponding to the trigger t_i and the slot in the prompt respectively. These vectors are obtained by averaging across all attention heads and associated subtokens[3]. Then for the argument slot $s_{k,i}$, we obtain the context-enhanced vector $\mathbf{c}_{k,i} \in \mathbb{R}^d$ which adaptively aggregates useful context and prompt information for argument extraction.

$$\mathbf{p}_k = \text{softmax}(\mathbf{A}_{t_i} \cdot \mathbf{A}_{s_{k,i}}), \\ \mathbf{c}_{k,i} = {\mathbf{H}_{\text{en}}}^T \, \mathbf{p}_k, \tag{6}$$

where $\mathbf{p}_k \in \mathbb{R}^{l_s}$ is the computed attention weight vector for argument slot $s_{k,i}$. Then $\mathbf{c}_{k,i}$ is subsequently incorporated into the decoder output $\mathbf{h}_{s_{k,i}} \in \mathbb{R}^d$ of slot $s_{k,i}$ to get $\tilde{\mathbf{h}}_{s_{k,i}} \in \mathbb{R}^d$:

$$\tilde{\mathbf{h}}_{s_{k,i}} = \tanh(\mathbf{W}_1[\mathbf{h}_{s_{k,i}}; \mathbf{c}_{k,i}]), \tag{7}$$

where $\mathbf{W}_1 \in \mathbb{R}^{2d \times d}$ is learnable parameter.

3.4 Span Selection

After obtaining the final representation $\tilde{\mathbf{h}}_{s_{k,i}}$ for each slot within each event, we follow (Ma et al., 2022) and transform each of them into a set of span selector $\{\Phi^{\text{start}}_{s_{k,i}}, \Phi^{\text{end}}_{s_{k,i}}\}$:

$$\Phi^{\text{start}}_{s_{k,i}} = \tilde{\mathbf{h}}_{s_{k,i}} \circ \mathbf{w}_{\text{start}}, \\ \Phi^{\text{end}}_{s_{k,i}} = \tilde{\mathbf{h}}_{s_{k,i}} \circ \mathbf{w}_{\text{end}}, \tag{8}$$

where $\mathbf{w}_{\text{start}}, \mathbf{w}_{\text{end}} \in \mathbb{R}^d$ are learnable parameters and $\circ$ represents element-wise multiplication.

Then $\Phi^{\text{start}}_{s_{k,i}}$ and $\Phi^{\text{end}}_{s_{k,i}}$ determine the start and end positions of slot $s_{k,i}$ in the original text:

$$\text{logit}^{\text{start}}_{k,i} = \text{softmax}(\mathbf{H}_{\text{de}} \Phi^{\text{start}}_{s_{k,i}}) \in \mathbb{R}^{l_s}, \\ \text{logit}^{\text{end}}_{k,i} = \text{softmax}(\mathbf{H}_{\text{de}} \Phi^{\text{end}}_{s_{k,i}}) \in \mathbb{R}^{l_s}, \\ \text{score}_{k,i}(m,n) = \text{logit}^{\text{start}}_{k,i}(m) + \text{logit}^{\text{end}}_{k,i}(n), \\ (\hat{s}_{k,i}, \hat{e}_{k,i}) = \underset{(m,n)\in C}{\arg\max}\, \text{score}_{k,i}(m,n), \tag{9}$$

where $(\hat{s}_{k,i}, \hat{e}_{k,i})$ is the predicted argument span. $C = \{(m,n) \mid (m,n) \in {l_s}^2, 0 < n - m \leq l\} \cup \{(0,0)\}$ contains all spans not exceeding the threshold l, along with the empty span $(0,0)$.

Following (Ma et al., 2022), we utilize Bipartite Matching Loss (Carion et al., 2020), which provides further consideration for the assignment of golden argument spans during training.

$$\mathcal{L} = -\sum_{i=1}^{\mathcal{K}} \sum_{(\hat{s}_{k,i}, \hat{e}_{k,i}) \in \delta(\mathcal{S}_i)} [\log \text{logit}^{\text{start}}_{k,i}(\hat{s}_{k,i}) \\ + \log \text{logit}^{\text{end}}_{k,i}(\hat{e}_{k,i})], \tag{10}$$

where $\mathcal{K}$ is the number of target events and i represents the i-th event. $\delta(\mathcal{S}_i)$ denotes the optimal assignment (Ma et al., 2022) calculated through the Hungarian algorithm (Kuhn, 1955).

4 Experiments

4.1 Experimental Setup

Datasets We evaluate our model on three document-level EAE datasets, including RAMS (Ebner et al., 2020),WikiEvents (Li et al., 2021) and MLEE (Pyysalo et al., 2012). Moreover, we also extend our evaluation of the model to the sentence-level ACE05 dataset (Doddington et al., 2004), as it includes a significant number of instances with multiple events. The detailed dataset description and statistics are shown in Appendix B.1.

Evaluation Metrics Following previous works (Ma et al., 2022; He et al., 2023), we evaluate performance using two metrics: (1) strict argument identification F1 (Arg-I), where a predicted event argument is considered correct if its boundaries match those of any corresponding golden arguments. (2) Strict argument classification F1 (Arg-C), where a predicted event argument is considered correct only if both its boundaries and role type are accurate. We conduct experiments on 5 runs with different seeds and report the average results.

[3] We only take the start token $\langle t_i \rangle$ to represent the trigger.

Scheme	Method	PLM	RAMS		WikiEvents		MLEE	
			Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C
Span-based single-event	TSAR (2022)	BERT-b	-	48.1	70.8	65.5	72.3	71.3
	TSAR (2022)	RoBERTa-l	-	51.2	71.1	65.8	72.6	71.5
	SCPRG (2023c)	BERT-b	53.9*	48.9	70.1*	65.8*	-	-
	SCPRG (2023c)	RoBERTa-l	56.7*	52.3	71.3*	66.4*	-	-
Generation single-event	DocMRC (2021)	BERT-b	-	45.7	-	43.3	-	-
	EEQA (2020)	BART-l	48.7	46.7	56.9	54.5	70.3	68.7
	FEAE (2021)	BERT-b	53.5	47.4	-	-	-	-
	BART-Gen (2021)	BART-l	51.2	48.6	66.8	62.4	71.0	69.8
	HRA (2023)	T5-l	54.6	48.4	69.6	63.4	-	-
Prompt-based single-event	RKDE (2023)	BART-l	55.1	50.3	69.1	63.8	-	-
	PAIE (2022)	BART-l	56.8	52.2	70.5	65.3	72.1*	70.8*
	SPEAE (2023)	BART-l	58.0	53.3	**71.9**	66.1	-	-
	TabEAE (2023)	RoBERTa-l	57.0	52.5	70.8	65.4	71.9	71.0
Prompt-based multi-event	PAIE-multi	BART-l	55.9	50.9	67.2	61.7	71.3	69.5
	TabEAE-multi	RoBERTa-l	56.7	51.8	71.1	66.0	75.1	74.2
	DEEIA(Ours)	RoBERTa-l	**58.0**	**53.4**	71.8	**67.0**	**75.2**	**74.3**

Table 1: Comparison of performance on RAMS, WikiEvents and MLEE test set. * means we rerun their code based on their experimental settings. **Bold** and underline indicate the best and second-best experimental results.

Baselines Our baselines include: (1) Two SOTA span-based methods, *TSAR* (Xu et al., 2022) and *SCPRG* (Liu et al., 2023c); (2) Five typical generation-based methods, *DocMRC* (Liu et al., 2021), *EEQA* (Du and Cardie, 2020), *FEAE* (Wei et al., 2021), *BART-Gen* (Li et al., 2021), and *HRA* (Ren et al., 2023); (3) Four prompt-based approaches: *RKDE* (Hu et al., 2023), *PAIE*, SPEAE (Nguyen, 2023) and *TableEAE*. We also compare with *LLM approaches* and conduct a detailed analysis in the Appendix C.5.

Most baselines are originally proposed as Single-EAE methods. For the Multi-EAE baselines, we extend PAIE and TabEAE to obtain PAIE-multi[4] and TabEAE-multi[5]. Our experimental details are shown in Appendix B.2.

4.2 Main Results

Table 1 illustrates the performance comparison between our proposed DEEIA method and various baseline approaches across three datasets. (Experimental results on ACE05 dataset are shown in Appendix B.3.) Our approach consistently achieves optimal results across all datasets generally, irrespective of the evaluation metric employed. Further analyzing the experimental result data, we observe: (1) The performance of DEEIA outperforms that of Single-EAE baselines. This indicates our DEEIA can **effectively capture the beneficial event correlations**, enhancing the performance of EAE task. (2) PAIE-multi exhibits significantly lower performance compared to PAIE, which illustrates that simultaneously processing multiple events significantly increases the difficulty of the task. While our DEEIA significantly outperforms the baseline PAIE-multi on three datasets, demonstrating that our DEEIA effectively addresses the challenge of multi-event information complexity. (3) Compared to PAIE and TabEAE, the improvement of our DEEIA model on RAMS is around 0.9-1.2 F1, while on WikiEvents and MLEE, the improvement is around 1.2-1.7 F1 and 1.7-3.7 F1 respectively. Therefore, the improvement of DEEIA on WikiEvents and MLEE is more pronounced compared to RAMS. We hypothesize this may be because WikiEvents and MLEE contain a higher proportion of multi-event instances, in contrast to the RAMS dataset, which is primarily composed of single-event instances. (Distributions of event numbers on three datasets are shown in Appendix B.1.)

4.3 Ablation Study

To better illustrate the effectiveness of different components, we conduct ablation studies on RAMS and WikiEvents datasets in Table 2. We divide the dataset into two parts based on the number of events in each instance: those with # E > 1 and those with # E = 1, and report the Arg-C F1 scores to explore the impact of our modules on instances with single and multiple events.

Without Dependency-guided Encoding (DE).

[4] We extend the original PAIE (Ma et al., 2022) into the multi-EAE framework by annotating triggers within the context and concatenating prompts for multiple events. The rest of the approach remains consistent with the original PAIE.

[5] TabEAE (He et al., 2023) aims to capture event co-occurrence and trains the model on multi-event scheme but infers on single-event scheme. For a fair comparison, we train their method on multi-event instances and conduct inference on multi-event instances as well.

Model	RAMS			WikiEvents		
	All [871]	# E = 1 [587]	# E > 1 [284]	All [365]	# E = 1 [114]	# E > 1 [251]
PAIE-multi	50.86±0.22	51.75±0.34	48.94±0.33	61.74±0.62	65.01±1.20	60.18±0.81
TabEAE-multi	51.44±0.32	52.27±0.28	50.42±0.44	65.68±0.62	67.08±0.44	65.02±0.28
DEEIA(Ours)	**53.36±0.44**	53.64±0.60	**52.76±0.32**	**66.95±0.66**	**67.49±0.72**	**66.57±0.62**
w/o DE	52.38±0.49	**53.84±0.74**	49.86±0.78	64.90±1.07	66.96±1.06	63.79±0.77
w/o intra	51.49±0.55	53.06±0.62	48.34±0.64	64.65±0.56	66.86±0.44	63.12±0.75
w/o inter	52.18±0.44	53.56±0.52	49.16±0.58	65.65±0.66	67.16±0.48	65.00±0.66
w/o PE	52.41±0.37	53.08±0.43	51.06±0.38	66.02±0.70	67.06±0.58	65.59±0.72
w/o EIA	51.70±0.56	52.31±0.52	49.90±0.68	64.20±0.24	65.43±0.42	63.57±0.20
w/o DE & EIA	51.25±0.58	52.14±0.54	49.66±0.62	62.37±1.04	65.56±1.06	61.68±0.88

Table 2: Ablation study on RAMS and WikiEvents. Strict argument classification F1 scores (Arg-C) are reported. # E means the number of events in an instance and [] indicates the number of instances of this kind. **Bold** indicates the best experimental results. The reported results are averaged from 5 different random seeds.

We replace the Dependency-guided Encoding (DE) module with a vanilla transformer encoder. This results in performance reduction, primarily attributed to the decline in the performance of multi-event samples, which illustrates that DE module effectively provides dependency guidance for multi-event extraction. We further explore the effectiveness of intra and inter dependencies. It is observed that both intra and inter dependencies contribute positively to the model. When remaining only one type of dependency, the intra dependency has a beneficial effect on the model, but the inter dependency has a negative effect.

Without Event-specific Information Aggregation (EIA). The performance of both multi-event and single-event samples has significantly declined on two datasets. This indicates that our EIA module can provide beneficial event-specific information. Moreover, when removing both DE and EIA modules, the performance decay exceeds that when removing a single module, which explains that our two modules can work together.

Without Prompt Enhancing (PE). When removing the event type information defined in § 3.1, the performance on two datasets decays slightly, which indicates that event type helps the model distinguish between different event prompts and integrates wider information.

5 Analysis

5.1 Efficiency Analysis

Table 3 reports the efficiency of different prompt-based methods. First, compared to Single-EAE baseline PAIE, our method saves 8.76%, 32.96%, and **35.20%** inference time on RAMS, WikiEvents, MLEE datasets respectively. This fully demonstrates the efficiency superiority of our Multi-EAE approach. Additionally, our method significantly reduces 55.57%, 57.38% and **69.19%** inference time compared to TabEAE-multi, with almost no increase in the number of parameters. This illustrates that our DEEIA model enhances the efficiency of the document-level EAE task.

Method	Params	Inference Time		
		RAMS	Wikievents	MLEE
PAIE	406.21M	15.86	8.83	14.29
PAIE-multi	406.21M	12.53	5.15	8.57
TabEAE-multi	383.78M	32.59	13.89	30.06
DEEIA (Ours)	388.12M	14.47	5.92	9.26

Table 3: Inference time (second) for different models (large) on test set of three datasets. Experiments are run on one same Tesla A100 GPU.

5.2 Effect Analysis on Event Numbers

To further investigate the effectiveness of our method in addressing the multi-event information complexity problem, we divide the documents in the development sets of WikiEvents and MLEE into different groups based on the event numbers[6]. As illustrated in Figure 4, as the event number increases, we observe a decreasing trend in the performance of all models. We believe this is due to the fact that more events require the model to process more complex information and longer text, which is more difficult. Furthermore, we find that the baseline model PAIE-multi performs significantly worse on samples where the number of events exceeds two. In contrast, our model demonstrates a marked improvement in multi-event samples compared to PAIE-multi and TabEAE-multi, which shows the superiority of DEEIA in capturing the event correlations among multiple events. The results on WikiEvents dataset are in Appendix C.1.

[6] We do not use the RAMS dataset because the RAMS dataset has a low proportion of multi-event instances.

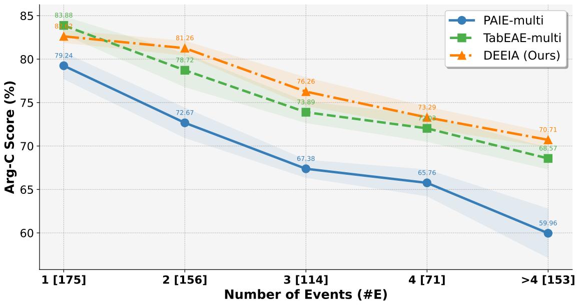
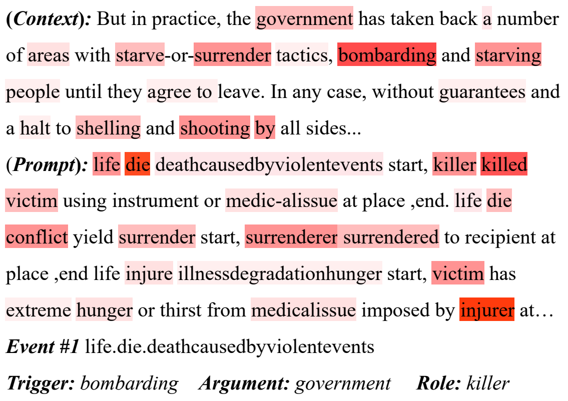


Figure 4: The averaged performance of the PAIE-multi, TabEAE-multi, and DEEIA models on samples with different event numbers in MLEE dataset. Our model achieves better results on samples with multiple events.



(Context): But in practice, the government has taken back a number of areas with starve-or-surrender tactics, bombarding and starving people until they agree to leave. In any case, without guarantees and a halt to shelling and shooting by all sides...

(Prompt): life die deathcausedbyviolentevents start, killer killed victim using instrument or medic-alissue at place ,end. life die conflict yield surrender start, surrenderer surrendered to recipient at place ,end life injure illnessdegradationhunger start, victim has extreme hunger or thirst from medicalissue imposed by injurer at…

Event #1 life.die.deathcausedbyviolentevents

Trigger: *bombarding* ***Argument:*** *government* ***Role:*** *killer*

Figure 5: Visualization of attentive weights in EIA module from an example in RAMS. We calculate the attentive weight $\mathbf{p}_k$ based on the representations of argument slot "government" and the trigger "bombarding".

5.3 Analysis of Two Modules

Dependency Guidance To investigate how the attentive biases influence the self-attention mechanism, we visualize all attentive biases (calculated in Eq. 4) for the test sets of all three datasets. We conducted a detailed analysis in Appendix C.2. We find that both inter and intra dependencies can provide effective information guidance for Multi-EAE task. In datasets with a larger proportion of multi-event documents, such as the MLEE dataset, both inter and intra dependencies exhibit significant effects, while in datasets with a smaller proportion of multi-event documents, such as RAMS, intra dependency plays a primary role.

Analysis of Information Aggregation To assess the effectiveness of our EIA module in capturing event-specific contextual information, we visualize the attentive weights $\mathbf{p}_k$ in Eq. 6 of an argument "government" of Figure 1. As shown in Figure 5, our EIA module gives high weights to the context words, such as *starving*, *shooting* and *surrender*, prompt words such as *die*, *killer* and *injurer*, which benefits the argument extraction of "government". Interestingly, some words such as *surrender* and *shelling* also act as triggers or arguments in other events, which reveals the EIA module's capability to capture event correlations.

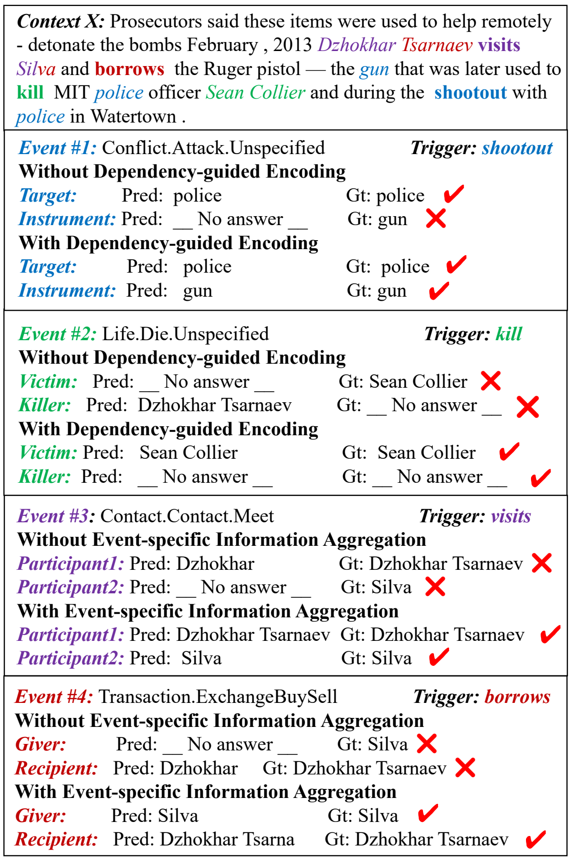
5.4 Error Analysis and Case Study

Error Analysis We further conduct error analysis to explore the effectiveness of our DEEIA model in Appendix C.4. We analyze all prediction errors on WikiEvents test set and categorize them into five classes. As shown in Figure 10, compared to the Single-EAE baseline PAIE, our DEEIA model reduces the number of errors from 312 to 292, indicating the effectiveness of DEEIA in capturing event correlations. Compared to PAIE-multi, DEEIA reduces the number of errors from 358 to 292. Additionally, our DE and EIA modules also significantly reduce specific types of errors. The detailed analysis is shown in Appendix C.4

Case Study We conducted the case study to further explore the effect of our proposed modules in multi-EAE. As shown in Figure 6, this is a complex document containing four events and there exists the argument overlapping phenomenon. First, without the dependency-guide encoding (DE), our model fails to identify arguments such as "Sean Collier" and "gun". However, with the DE module, our model correctly predicts the roles of these arguments. This demonstrates that the event dependencies provide beneficial guidance. Additionally, with the EIA module, our model is capable of extracting overlapping arguments like "Dzhokhar Tsarnaev" and "Silva", which indicates that the EIA module provides event-specific contextual information for a better context understanding.

6 Conclusion

In this paper, we propose a Multi-EAE model DEEIA, which overcomes the inefficiency limitations of traditional EAE methods. The proposed Dependency-guide Encoding (DE) module and Event-specific Information Extraction (EIA) module effectively enhances the model's ability to understand complex multi-event contexts. Our extensive experiments on three public benchmarks illustrate the superiority of our model in performance and efficiency.



Context X: Prosecutors said these items were used to help remotely - detonate the bombs February , 2013 *Dzhokhar Tsarnaev* **visits** *Silva* and **borrows** the Ruger pistol — the *gun* that was later used to **kill** MIT *police* officer *Sean Collier* and during the **shootout** with *police* in Watertown .

Event #1: Conflict.Attack.Unspecified ***Trigger:*** ***shootout***
Without Dependency-guided Encoding
Target: Pred: police Gt: police ✓
Instrument: Pred: __ No answer __ Gt: gun ✗
With Dependency-guided Encoding
Target: Pred: police Gt: police ✓
Instrument: Pred: gun Gt: gun ✓

Event #2: Life.Die.Unspecified ***Trigger:*** ***kill***
Without Dependency-guided Encoding
Victim: Pred: __ No answer __ Gt: Sean Collier ✗
Killer: Pred: Dzhokhar Tsarnaev Gt: __ No answer __ ✗
With Dependency-guided Encoding
Victim: Pred: Sean Collier Gt: Sean Collier ✓
Killer: Pred: __ No answer __ Gt: __ No answer __ ✓

Event #3*: Contact.Contact.Meet ***Trigger: ***visits***
Without Event-specific Information Aggregation
Participant1: Pred: Dzhokhar Gt: Dzhokhar Tsarnaev ✗
Participant2: Pred: __ No answer __ Gt: Silva ✗
With Event-specific Information Aggregation
Participant1: Pred: Dzhokhar Tsarnaev Gt: Dzhokhar Tsarnaev ✓
Participant2: Pred: Silva Gt: Silva ✓

Event #4: Transaction.ExchangeBuySell ***Trigger:*** ***borrows***
Without Event-specific Information Aggregation
Giver: Pred: __ No answer __ Gt: Silva ✗
Recipient: Pred: Dzhokhar Gt: Dzhokhar Tsarnaev ✗
With Event-specific Information Aggregation
Giver: Pred: Silva Gt: Silva ✓
Recipient: Pred: Dzhokhar Tsarna Gt: Dzhokhar Tsarnaev ✓

Figure 6: A multi-event test case from WikiEvents.

7 Limitations

The primary limitation of our method is the issue of input length. Concatenated text and prompts are more easy to exceed the maximum input length. Currently, our solution to this is to use a sliding window approach (Zhou et al., 2021; Zhang et al., 2021) to encode the sequences of different windows and average the overlapping token embeddings of different windows to obtain the final representation. However, this method is not the optimal solution for processing long texts, which leads to the information loss and results in suboptimal performance. Therefore, in the future, we will explore to address the challenge of long input text with the aim of enhancing our DEEIA model.

Acknowledgements

This work was supported by the National Science Foundation of China under Grant 62106038, and in part by the Sichuan Science and Technology Program under Grant 2023YFG0259.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Amit Bagga and Alan W Biermann. 1997. Analyzing the complexity of a domain with respect to an information extraction task. In *Proceedings of the 10th Research on Computational Linguistics International Conference*, pages 175–194.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer.

Yiming Chen, Yan Zhang, Chen Zhang, Grandee Lee, Ran Cheng, and Haizhou Li. 2021. Revisiting self-training for few-shot learning of language model. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9125–9135, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *Lrec*, volume 2, pages 837–840. Lisbon.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proc. of EMNLP*.

Xinya Du, Alexander M Rush, and Claire Cardie. 2021. Template filling with generative transformers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 909–914.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proc. of ACL*.

Ralph Grishman. 2015. Information extraction. *IEEE Intelligent Systems*, 30(5):8–15.

Yuxin He, Jingyue Hu, and Buzhou Tang. 2023. Revisiting event argument extraction: Can eae models learn better when being aware of event co-occurrences? *arXiv preprint arXiv:2306.00502*.

Hobbs. 2010. Information extraction. *Handbook of natural language processing*, 15:16.

I Hsu, Zhiyu Xie, Kuan-Hao Huang, Prem Natarajan, Nanyun Peng, et al. 2023. Ampere: Amr-aware prefix for generation-based event argument extraction model. *arXiv preprint arXiv:2305.16734*.

Ruijuan Hu, Haiyan Liu, and Huijuan Zhou. 2023. Role knowledge prompting for document-level event argument extraction. *Applied Sciences*, 13(5):3041.

Quzhe Huang, Yanxi Zhang, and Dongyan Zhao. 2023. From simple to complex: A progressive framework for document-level informative argument extraction. *arXiv preprint arXiv:2310.16358*.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Hao Li, Yanan Cao, Yubing Ren, Fang Fang, Lanxue Zhang, Yingjie Li, and Shi Wang. 2023a. Intra-event and inter-event dependency-aware graph network for event argument extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6362–6372.

Junpeng Li, Zixia Jia, and Zilong Zheng. 2023b. Semi-automatic data enhancement for document-level relation extraction with distant supervision from large language models. *arXiv preprint arXiv:2311.07314*.

Sha Li, Heng Ji, and Jiawei Han. 2021. Document-level event argument extraction by conditional generation. In *Proc. of NAACL*.

Jian Liu, Yufeng Chen, and Jinan Xu. 2021. Machine reading comprehension as data augmentation: A case study on implicit event argument extraction. In *Proc. of EMNLP*.

Jian Liu, Chen Liang, Jinan Xu, Haoyan Liu, and Zhe Zhao. 2023a. Document-level event argument extraction with a chain reasoning paradigm. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9570–9583, Toronto, Canada. Association for Computational Linguistics.

Jian Liu, Chen Liang, Jinan Xu, Haoyan Liu, and Zhe Zhao. 2023b. Document-level event argument extraction with a chain reasoning paradigm. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9570–9583.

Shulin Liu, Yubo Chen, Kang Liu, and Jun Zhao. 2017. Exploiting argument information to improve event detection via supervised attention mechanisms. In *Proc. of ACL*.

Wanlong Liu, Shaohuan Cheng, Dingyi Zeng, and Qu Hong. 2023c. Enhancing document-level event argument extraction with contextual clues and role relevance. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12908–12922.

Wanlong Liu, Dingyi Zeng, Li Zhou, Yichen Xiao, Weishan Kong, Malu Zhang, Shaohuan Cheng, Hongyang Zhao, and Wenyu Chen. 2024. Utilizing contextual clues and role correlations for enhancing document-level event argument extraction.

Wanlong Liu, Li Zhou, Dingyi Zeng, and Hong Qu. 2022. Document-level relation extraction with structure enhanced transformer encoder. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Danqing Luo, Chen Zhang, Yan Zhang, and Haizhou Li. 2024. CrossTune: Black-box few-shot classification with label enhancement. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 4185–4197, Torino, Italy. ELRA and ICCL.

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022. Prompt for extraction? paie: Prompting argument interaction for event argument extraction. *arXiv preprint arXiv:2202.12109*.

Thien Nguyen, Chien. 2023. Contextualized soft prompts for extraction of event arguments. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4352–4361.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Sampo Pyysalo, Tomoko Ohta, Makoto Miwa, Han-Cheol Cho, Jun'ichi Tsujii, and Sophia Ananiadou. 2012. Event extraction across multiple levels of biological organization. *Bioinformatics*.

Yubing Ren, Yanan Cao, Fang Fang, Ping Guo, Zheng Lin, Wei Ma, and Yi Liu. 2022. Clio: Role-interactive multi-event head attention network for document-level event extraction. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2504–2514.

Yubing Ren, Yanan Cao, Ping Guo, Fang Fang, Wei Ma, and Zheng Lin. 2023. Retrieve-and-sample: Document-level event argument extraction via hybrid retrieval augmentation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 293–306.

Hai-Long Trieu, Thy Thy Tran, Khoa NA Duong, Anh Nguyen, Makoto Miwa, and Sophia Ananiadou. 2020. Deepeventmine: end-to-end neural nested event extraction from biomedical texts. *Bioinformatics*, 36(19):4910–4917.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Guanghui Wang, Dexi Liu, Qizhi Wan, Xiping Liu, and Wanlong Liu. 2024. Degap: Dual event-guided adaptive prefixes for templated-based event argument extraction model with slot querying.

Sijia Wang, Mo Yu, Shiyu Chang, Lichao Sun, and Lifu Huang. 2022. Query and extract: Refining event extraction as type-oriented binary decoding. In *Proc. of ACL Findings*.

Kaiwen Wei, Xian Sun, Zequn Zhang, Jingyuan Zhang, Guo Zhi, and Li Jin. 2021. Trigger is not sufficient: Exploiting frame-aware knowledge for implicit event argument extraction. In *Proc. of ACL*.

Yuwei Xia, Ding Wang, Qiang Liu, Liang Wang, Shu Wu, and Xiaoyu Zhang. 2024a. Enhancing temporal knowledge graph forecasting with large language models via chain-of-history reasoning. *arXiv preprint arXiv:2402.14382*.

Yuwei Xia, Mengqi Zhang, Qiang Liu, Liang Wang, Shu Wu, Xiaoyu Zhang, and Liang Wang. 2024b. Metatkg++: Learning evolving factor enhanced meta-knowledge for temporal knowledge graph reasoning. *Pattern Recognition*, page 110629.

Yuwei Xia, Mengqi Zhang, Qiang Liu, Shu Wu, and Xiao-Yu Zhang. 2022. Metatkg: Learning evolutionary meta-knowledge for temporal knowledge graph reasoning. In *EMNLP*, pages 7230–7240.

Benfeng Xu, Quan Wang, Yajuan Lyu, Yong Zhu, and Zhendong Mao. 2021. Entity structure within and throughout: Modeling mention dependencies for document-level relation extraction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 14149–14157.

Derong Xu, Wei Chen, Wenjun Peng, Chao Zhang, Tong Xu, Xiangyu Zhao, Xian Wu, Yefeng Zheng, and Enhong Chen. 2023. Large language models for generative information extraction: A survey. *arXiv preprint arXiv:2312.17617*.

Runxin Xu, Peiyi Wang, Tianyu Liu, Shuang Zeng, Baobao Chang, and Zhifang Sui. 2022. A two-stream amr-enhanced model for document-level event argument extraction. *arXiv e-prints*.

Hang Yang, Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Taifeng Wang. 2021. 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 (Volume 1: Long Papers)*, pages 6298–6308.

Yuqing Yang, Qipeng Guo, Xiangkun Hu, Yue Zhang, Xipeng Qiu, and Zheng Zhang. 2023. An amr-based link prediction approach for document-level event argument extraction. *arXiv preprint arXiv:2305.19162*.

Dingyi Zeng, Wanlong Liu, Wenyu Chen, Li Zhou, Malu Zhang, and Hong Qu. 2023. Substructure aware graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11129–11137.

Dingyi Zeng, Li Zhou, Wanlong Liu, Hong Qu, and Wenyu Chen. 2022a. A simple graph neural network via layer sniffer. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5687–5691. IEEE.

Qi Zeng, Qiusi Zhan, and Heng Ji. 2022b. Improving consistency with event awareness for document-level argument extraction. *arXiv preprint arXiv:2205.14847*.

Chen Zhang, Luis D'Haro, Chengguang Tang, Ke Shi, Guohua Tang, and Haizhou Li. 2023. xDial-eval: A multilingual open-domain dialogue evaluation benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5579–5601, Singapore. Association for Computational Linguistics.

Chen Zhang, Luis Fernando D'Haro, Yiming Chen, Malu Zhang, and Haizhou Li. 2024a. A comprehensive analysis of the effectiveness of large language models as automatic dialogue evaluators. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19515–19524.

Ningyu Zhang, Xiang Chen, Xin Xie, Shumin Deng, Chuanqi Tan, Mosha Chen, Fei Huang, Luo Si, and Huajun Chen. 2021. Document-level relation extraction as semantic segmentation. *arXiv preprint arXiv:2106.03618*.

Xinliang Frederick Zhang, Carter Blum, Temma Choji, Shalin Shah, and Alakananda Vempala. 2024b. Ultra: Unleash llms' potential for event argument extraction through hierarchical modeling and pair-wise refinement. *arXiv preprint arXiv:2401.13218*.

Zhisong Zhang, Xiang Kong, Zhengzhong Liu, Xuezhe Ma, and Eduard Hovy. 2020. A two-step approach for implicit event argument detection. In *Proc. of ACL*.

Hanzhang Zhou, Junlang Qian, Zijian Feng, Hui Lu, Zixiao Zhu, and Kezhi Mao. 2023a. Heuristics-driven link-of-analogy prompting: Enhancing large language models for document-level event argument extraction. *arXiv preprint arXiv:2311.06555*.

Ji Zhou, Kai Shuang, Qiwei Wang, and Xuyang Yao. 2024. Eace: A document-level event argument extraction model with argument constraint enhancement. *Information Processing & Management*, 61(1):103559.

Li Zhou, Laura Cabello, Yong Cao, and Daniel Hershcovich. 2023b. Cross-cultural transfer learning for Chinese offensive language detection. In *Proceedings of the First Workshop on Cross-Cultural Considerations in NLP (C3NLP)*, pages 8–15, Dubrovnik, Croatia. Association for Computational Linguistics.

Li Zhou, Wenyu Chen, Dingyi Zeng, Malu Zhang, and Daniel Hershcovich. Rethinking relation classification with graph meaning representations. *Available at SSRN 4703687*.

Li Zhou, Antonia Karamolegkou, Wenyu Chen, and Daniel Hershcovich. 2023c. Cultural compass: Predicting transfer learning success in offensive language detection with cultural features. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12684–12702, Singapore. Association for Computational Linguistics.

Wenxuan Zhou, Kevin Huang, Tengyu Ma, and Jing Huang. 2021. Document-level relation extraction with adaptive thresholding and localized context pooling. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 14612–14620.

Dataset	RAMS	WikiEvents	MLEE
# Event types	139	50	23
# Events per text	1.25	1.78	3.32
# Args per event	2.33	1.40	1.29
# Events			
Train	7329	3241	4442
Dev	924	345	-
Test	871	365	2200

Table 4: Dataset Statistics.

Algorithm 1 Multi-EAE with Dynamic Windows

```
Require: Input context $X$, concatenated multi-event prompts $P$, window sizes $d_1$ and $d_2$ ($d_1 + d_2 <$ max length)
Ensure: Final representation of the sequence
if length$(X + P) >$ max length then
    Split $X$ into $\{X_1, X_2, \ldots, X_n\}$ using $d_1$
end if
for each $X_i$ in $\{X_1, X_2, \ldots, X_n\}$ do
    Identify number of events in $X_i$ based on triggers, get $P_i$
    if length$(X_i + P_i) >$ max length then
        Split $P_i$ into $\{P_i^1, \ldots, P_i^m\}$ using $d_2$
    end if
    for each $P_i^j$ in $\{P_i^1, \ldots, P_i^m\}$ do
        Concatenate $X_i$ with $P_i^j$
        Encode them to $S_i^j$
    end for
    Average pool $S_i^j$ to obtain final $S_i$
end for
Aggregate: Pool $S_i$ to get the final sequence representation
```

A Dynamic Window Algorithm

For sequences that surpass the maximum length of 512, we employ a sliding window approach to process longer sequences. For the general case of processing long input texts, we have designed the following Algorithm 1 based on sliding windows. In our implement, PAIE (Ma et al., 2022) has already utilized a sliding window to divide the long document into several instances. We only use the sliding window to process the prompts and ultimately obtain the final sequence representation through pooling. In our experiments, both d_1 and d_2 are set to 250.

B Experimental Details

B.1 Dataset Statistics

We evaluate our proposed method on four event argument extraction datasets.

RAMS (Ebner et al., 2020) is a document-level EAE dataset with 9,124 annotated events from English online news, annotated event-wise. Following (He et al., 2023), we employ a sliding window approach to aggregate events in the same context into single instances with multiple events, following the original train/dev/test split.

WikiEvents (Zhang et al., 2020) is a document-level EAE dataset featuring events from English Wikipedia and associated news articles. It includes co-reference links for arguments, but we only use the exact argument annotations in our experiments.

MLEE (Pyysalo et al., 2012) is a document-level event extraction dataset, contains manually annotated abstracts from bio-medical publications in English. We follow the preprocessing steps outlined by (Trieu et al., 2020). Since there is only train/test data split for the preprocessed dataset, we employ the training set as the development set.

ACE05 (Doddington et al., 2004) is a labeled corpus used for information extraction, consisting of newswire, broadcast news, and telephone conversations. We employ its English event annotations for sentence-level Event Argument Extraction (EAE). The data preprocessing follows the method described by (Ma et al., 2022).

The detailed dataset statistics of three datasets are shown in Table 4. We also calculate the distributions of the number of events per instance on the three dataset, which are shown in Figure 7. As shown in Figure 7, three datasets exhibit different data distributions between single-event samples and multi-event samples. For RAMS dataset, single events samples dominate the majority, while the proportion of multi-event samples is quite small. However, for WikiEvents and MLEE datasets, multi-event samples account for a significant proportion.

B.2 Experimental Details

According to TabEAE, using RoBERTa as the PLM outperforms BART across multiple approaches (such as PAIE and TabEAE). Therefore, we adopt RoBERTa as our PLM so as to compare to the SOTA method. Our implementation utilizes Pytorch and runs on a Tesla A100 GPU. We configure the encoder using the initial 17 layers of RoBERTa-large (Liu et al., 2019). The decoder's self-attention and feedforward layers inherit their weights from RoBERTa-large's subsequent 7 layers. This division of a 17-layer encoder and a 7-layer decoder is empirically determined as the most effective configuration (He et al., 2023). It's important to note that the decoder's cross-attention component is initialized randomly, with its learning rate set at 1.5 times that of other parameters. More detailed hyperparameter setting is shown in Table 5. We utilize the prompts proposed in PAIE (Ma et al., 2022), which are shown in Table 9.

Hyperparameters	**RAMS**	**Wiki**	**MLEE**
Training Steps	10000	10000	10000
Warmup Ratio*	0.1	0.1	0.2
Learning Rate*	2e-5	3e-5	3e-5
Max Gradient Norm	5	5	5
Batch Size*	4	4	4
Context Window Size	250	250	250
Max Span Length	10	10	10
Max Encoder Seq Length	500	500	500
Max Prompt Length*	210	360	360
Encoder Layers*	17	17	17
Decoder Layers*	7	7	7
Gamma*	0.01	0.1	0.1

Table 5: Hyperparameter settings. * means that we tuned the hyperparameters in our experiments. The rest of hyperparameters are set the same as PAIE (Ma et al., 2022).

B.3 Experimental Results on ACE05

We evaluate our proposed model on the ACE05 dataset (Doddington et al., 2004), and the specific experimental results are shown in Table 6 (The reported results are averaged from 5 different random seeds). Experimental results demonstrate that our proposed DEEIA model also performs well on the sentence-level ACE05 dataset, significantly improving the extraction performance of multi-event instances.

C Experimental Analysis

C.1 Effect Analysis on Event Numbers in WikiEvents

As illustrated in Figure 8, as the event number increases, we observe a decreasing trend in the performance of all models. We believe this is due to the fact that more events require the model to process more complex information and longer text, which is more difficult. Furthermore, we find that the baseline model PAIE performs significantly worse on samples where the number of events exceeds two. In contrast, our model demonstrates a marked improvement in multi-event samples compared to PAIE-multi and TabEAE-multi, which shows the advantages of DEEIA in Multi-EAE.

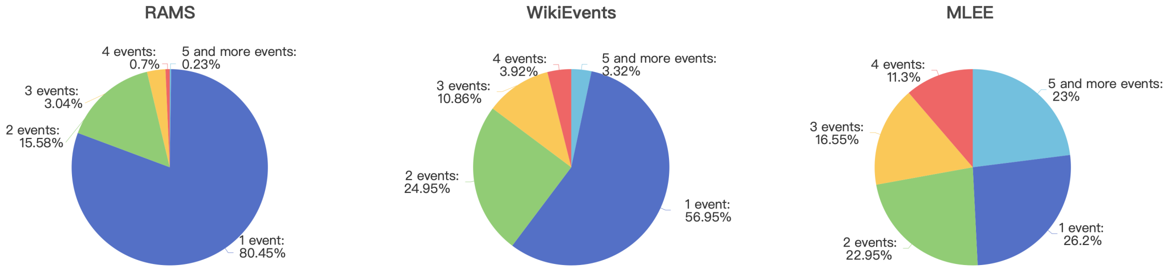
Method	PLM	ACE05	
		Arg-I	Arg-C
EEQA (2020)	BERT-l	70.5	68.9
EEQA (2020)	RoBERTa-l	72.1	70.4
BART-Gen (2021)	BART-l	69.9	66.7
PAIE (2022)	BART-l	75.7	72.7
PAIE (2022)	RoBERTa-l	76.1	73.0
TabEAE (2023)	RoBERTa-l	75.9	73.4
DEEIA (Ours)	RoBERTa-l	**76.3**	**74.1**

Table 6: Comparison of performance on ACE05 test set. * means we rerun their code based on their experimental settings. **Bold** indicates the best experimental results.

C.2 Analysis of Dependency Guidance

To investigate the manner in which the learnable attentive biases influence the self-attention mechanism, we collect all attentive biases (calculated in Eq. 4) for the test sets of all three datasets. These biases are then categorized based on dependency types and averaged across all attention heads and instances. As shown in Figure 9, the self-attention scores are primarily determined by vanilla self-attention, with minimal influence from dependency information at bottom layers. However, as the number of layers increases, the impact of learnable attentive biases gradually becomes significant, especially between layers 12 to 16.

Additionally, we observe that for different datasets, the attentive bias distributions corresponding to inter-event and intra-event dependencies are different. For RAMS dataset, the attentive bias associated with intra-event dependency is relatively positive, but that corresponding to inter-event dependency is relatively negative. We believe that in the RAMS dataset, there are more single-event samples, and the model focuses more on learning intra-event information associations. However, for WikiEvents and MLEE datasets, the attentive biases for both dependencies are mostly positive, indicating that in these datasets, samples with multiple events are more prevalent, and both dependencies provide beneficial guidance for the model to solve the information complexity problem.



(a) RAMS (b) WikiEvents (c) MLEE

Figure 7: Distributions of the number of events per instance on the three document-level datasets.

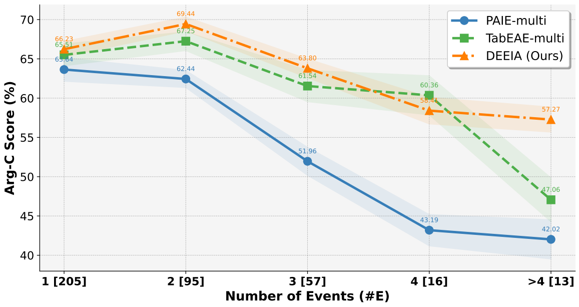


Figure 8: The averaged performance of the PAIE, TabEAE, and DEEIA models on samples with different event numbers in the WikiEvents dataset. Our model achieves better results on samples with multiple events.

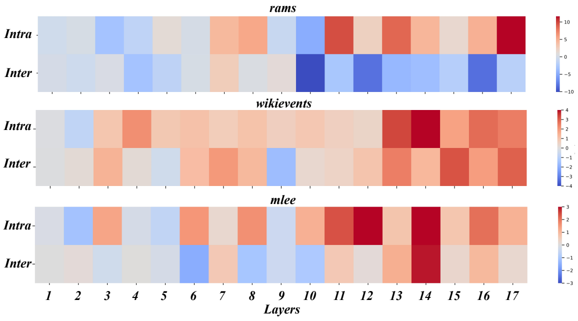


Figure 9: Visualization of the learnable attentive biases, where each cell represents the value of an attention bias. The horizontal axis represents intra-event and inter-event dependencies, and the vertical axis indicates each transformer layer incorporating structural guidance.

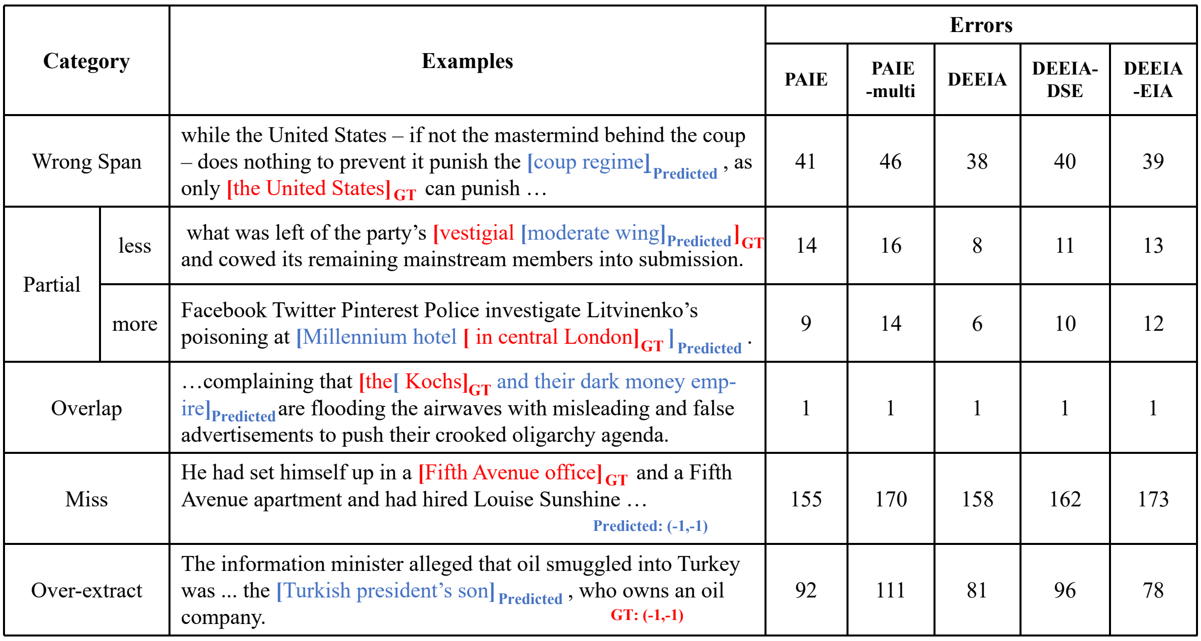
C.3 Architecture Variants

In this section, we explore the effect of different architectures and define two types of architecture variants. (1) To explore whether to integrate the dependency information into the decoder, we define **DEEIA**$_B$, which integrates the dependency information to both the encoder and decoder. (2) Since the prompts for events are defined based on their event types, the same event types will have the same prompts. Therefore, in a document, if there are multiple events of the same type, whether to concatenate repeated prompts becomes an option. We concatenate repeated prompts, and call this **DEEIA**$_M$.

As shown in Table 7, there is a minor performance drop across all three datasets when dependency information is integrated into the decoder. This implies that embedding dependency information during the encoding phase is adequate, and overloading the model with excessive informational guidance is not of benefit. Furthermore, concatenating repeated prompts results in a slight improvement in the performance for the RAMS and WikiEvents datasets, but a marginal decline for the MLEE dataset. Overall, the improvement is not substantial. We believe that while concatenating repeated prompts increases prompt diversity, it also extends the sequence length, thereby increasing the difficulty of long-distance reasoning.

Method	RAMS	WikiEvents	MLEE
DEEIA	53.4	67.0	74.3
DEEIA$_M$	53.5	67.2	74.4
DEEIA$_B$	53.2	66.5	74.6

Table 7: Analysis of variant architectures.



Category		Examples	Errors				
			PAIE	PAIE -multi	DEEIA	DEEIA-DSE	DEEIA -EIA
Wrong Span		while the United States – if not the mastermind behind the coup – does nothing to prevent it punish the [coup regime]$_{Predicted}$, as only [the United States]$_{GT}$ can punish …	41	46	38	40	39
Partial	less	what was left of the party's [vestigial [moderate wing]$_{Predicted}$]$_{GT}$ and cowed its remaining mainstream members into submission.	14	16	8	11	13
	more	Facebook Twitter Pinterest Police investigate Litvinenko's poisoning at [Millennium hotel [in central London]$_{GT}$]$_{Predicted}$.	9	14	6	10	12
Overlap		…complaining that [the[Kochs]$_{GT}$ and their dark money empire]$_{Predicted}$ are flooding the airwaves with misleading and false advertisements to push their crooked oligarchy agenda.	1	1	1	1	1
Miss		He had set himself up in a [Fifth Avenue office]$_{GT}$ and a Fifth Avenue apartment and had hired Louise Sunshine … Predicted: (-1,-1)	155	170	158	162	173
Over-extract		The information minister alleged that oil smuggled into Turkey was ... the [Turkish president's son]$_{Predicted}$, who owns an oil company. GT: (-1,-1)	92	111	81	96	78

Figure 10: Error Analysis on WikiEvents test set. We summarize the errors into five categories and count the number of errors for different models. Blue represents the model's predictions, while red represents the ground truth.

C.4 Error Analysis

To compare different models in greater detail, we conduct error analysis on dev sets of RAMS, WikiEvents and MLEE datasets. We divide the errors into five categories, which is shown in Figure 10. **Wrong span** refers to the case where the predicted span and the true span have no intersection; **Partial** refers to the case where the predicted span and the golden span partially overlap, which means the predicted span is a proper subset of the golden span or vice versa; **Overlap** occurs when there is a non-partial case, indicating that there is overlap between the predicted span and the golden span; **Over-extraction** refers to the case where the golden span is empty while other span is predicted; **Under-extraction** refers to the case where the golden span is not empty while the predicted span is empty.

As shown in Figure 10, compared to the Single-EAE baseline PAIE, our DEEIA model reduces the number of errors from 312 to 292, indicating the effectiveness of DEEIA in capturing event correlations. Compared to the baseline PAIE-multi, our DEEIA model reduces the number of errors from 358 to 292, especially decreasing the number of **Partial**, **Overlap**, and **Over-extract** errors. The ablation study shows that our proposed DE module mainly reduces the **Over-extract** and **Miss** errors, demonstrating that structural guidance can effectively help the model to deal with complex contexts. Meanwhile, the EIA module mainly reduces the **Miss** and **Partial** errors, indicating that this module offers event-specific contextual information for a better extraction of arguments.

C.5 Comparison with Large Language Models

Large language models (LLMs) have garnered substantial interest and attention from researchers, highlighting their extensive applicability across a wide array of tasks, such as text classification (Chen et al., 2021; Luo et al., 2024), dialogue (Zhang et al., 2023, 2024a), offensive language detection (Zhou et al., 2023c,b), graph tasks (Zeng et al., 2023, 2022a), and in particular, the formation Extraction (IE) tasks (Xu et al., 2023; Li et al., 2023b; Zhang et al., 2024b; Zhou et al.; Liu et al., 2022, 2024; Xia et al., 2024a,b). In this paper, we make a comparison with the recent state-of-the-art LLM-based approach presented in the work (Zhou et al., 2023a), which utilizes LLMs for the EAE task. We report their experimental results in Table 8. The experiments are conducted on three prominent large language models: text-davinci-003 (Ouyang et al., 2022), gpt-3.5-turbo (Ouyang et al., 2022) and GPT-4 (Achiam et al., 2023). These models are accessed via the

Method	RAMS	
	Arg-I	Arg-C
HD-LoA (Zhou et al., 2023a)		
text-davinci-003	46.1	39.5
gpt-3.5-turbo	38.3	31.5
gpt-4	50.4	42.8
DEEIA (Ours)	58.0	53.4

Table 8: Comparison with large language model method HD-LoA. We copy their experimental results.

public APIs from OpenAI's services[7]. As shown in Table 8, compared to the supervised learning models, LLMs still show a significant performance gap in the EAE task. Additionally, the operational costs of large models are inherently high. Our approach outperforms LLMs in terms of efficiency, cost, and effectiveness in the document-level EAE task.

[7] https://openai.com/api/

Dataset	Event Type	Natural Language Prompt
WikiEvents	ArtifactExistence. DamageDestroyDisableDismantle. Damage	Damager (and Damager) damaged Artifact (and Artifact) using Instrument (and Instrument) in Place (and Place).
	ArtifactExistence. DamageDestroyDisableDismantle. Destroy	Destroyer (and Destroyer) destroyed Artifact (and Artifact) using Instrument (and Instrument) in Place (and Place).
	ArtifactExistence. DamageDestroyDisableDismantle. DisableDefuse	Disabler (and Disabler) disabled or defused Artifact (and Artifact) using Instrument (and Instrument) in Place (and Place).
	ArtifactExistence. DamageDestroyDisableDismantle. Dismantle	Dismantler (and Dismantler) dismantled Artifact (and Artifact) using Instrument (and Instrument) from Components (and Components) in Place (and Place).
	ArtifactExistence. DamageDestroyDisableDismantle. Unspecified	DamagerDestroyer (and DamagerDestroyer) damaged or destroyed Artifact (and Artifact) using Instrument (and Instrument) in Place (and Place).
	ArtifactExistence. ManufactureAssemble. Unspecified	ManufacturerAssembler (and ManufacturerAssembler) manufactured or assembled or produced Artifact (and Artifact) from Components (and Components) using Instrument (and Instrument) at Place (and Place).
	Cognitive.IdentifyCategorize.Unspecified	Identifier (and Identifier) identified IdentifiedObject (and IdentifiedObject) as IdentifiedRole (and IdentifiedRole) at Place (and Place).
	Cognitive.Inspection.SensoryObserve	Observer (and Observer) observed ObservedEntity (and ObservedEntity) using Instrument (and Instrument) in Place (and Place).
RAMS	artifactexistence.artifactfailure. mechanicalfailure	Mechanical artifact failed due to instrument at place.
	artifactexistence.damagedestroy.n/a	DamagerDestroyer damaged or destroyed artifact using instrument in place.
	artifactexistence.damagedestroy.damage	Damager damaged artifact using instrument in place.
	artifactexistence.damagedestroy.destroy	Destroyer destroyed artifact using instrument in place.
	artifactexistence.shortage.shortage	Experiencer experienced a shortage of supply at place.
	conflict.attack.n/a	Attacker attacked target using instrument at place.
MLEE	Cell_proliferation	Cell proliferate or accumulate.
	Development	Anatomical Entity develop or form.
	Blood_vessel_development	Neovascularization or angiogenesis at Anatomical Location.
	Growth	Growth of Anatomical Entity.
	Death	Death of Anatomical Entity.
	Breakdown	Anatomical Entity degraded or damaged.
	Remodeling	Tissue remodeling or changes.
	Synthesis	Synthesis of Drug/Compound.

Table 9: Example of Prompts in Tabular Format