Structure and Co-occurrence aware for Document-level Event Argument Extraction

Anonymous ACL submission

Abstract

Document-level Event Argument Extraction (EAE) deals with longer texts, and more intricate relationships between events than sentence-level, which faced two problem: 1) semantic boundaries between events are difficult to distinguish; 2) redundant information distracts attention from events. To alleviate the aforementioned issues, we propose the Structure and Co-occurrence aware Event Argument Extraction model (SCEAE). SCEAE utilizes the PAIE architecture as the underlying framework. Building upon this framework, we incorporates two different knowledge-aware prefixes to tackle these problems. The Co-occurrence-aware prefix leverages knowledge of event co-occurrence to enhance the model's perception of semantic boundaries between events. The Structure-aware prefix helps the model establish structured relationships between the sentence. We tested our model on the RAMS, WikiEvents and MLEE datasets. The experiments showed that our model achieved gains of 2.1%, 2.3%, and 3.2% in the Arg-C F1 metric compared to PAIE on RAMS, WikiEvents and MLEE respectively. Furthermore, our model achieved new state-of-the-art performance. We will make all the progress publicly available at https://github.com/—.

1 Introduction

Event Argument Extraction (EAE) is an important subfield of Event Extraction (EE), which aims to identify arguments and assign them the correct roles. The structured text output from the EAE task plays a significant role in various downstream tasks such as question answering(Costa et al., 2020), dialogue systems(Zhang et al., 2020), and recommendation systems(Li et al., 2020b). In recent times, the success of Pre-trained Language Models (PLM) has led to extensive research on document-level EAE, enabling the extraction of events from documents consisting of multiple sentences.

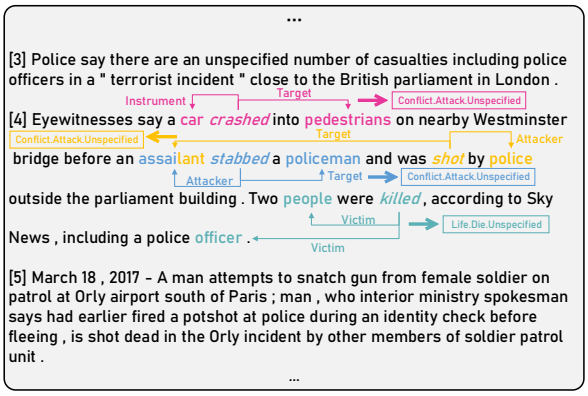

Figure 1: An EAE example from the WikiEvents dataset involves a sentence with the document ID 4, where four events are triggered by different trigger words. The arguments of these four events are closely distributed, and there may even be instances of overlapping events.

As shown in the figure 1, it illustrates an example of document-level event argument extraction. In the document, a sentence ID 4 contains four events. The argument distribution of these events is extremely dense, and different events can share the same token span as arguments corresponding to different roles. These dense and overlapping events make the semantic boundaries between them blurry, which increases the difficulty of extraction. Moreover, as can be seen from the figure 1, the explosion in the length of document-level data brings a significant amount of additional information from surrounding sentences. Some of this information is beneficial for EAE, while others introduce redundant information that can mislead the extraction process. For example, in the sentence number 5, the presence of person nouns such as "man," "female," "soldier," "spokesman," "police," and "soldier" can mislead the extraction of the "victim" argument for the "Life.Die.Unspecified" event triggered by "killed." These redundant pieces of information will cause the attention of the model to be distracted, thereby hindering the EAE process.

To alleviate the above issues, we have strength-

ened the model's boundaries from both the event and sentence perspectives, emphasizing key information in the extraction task. At the event level we have incorporated knowledge of event co-occurrence, which refers to multiple events occurring simultaneously in event mentions. There is a strong potential causal link between these co-occurring events. For example, in the depicted graph 1, the "Conflict.Attack.Unspecified" event triggered by "stabbed" and the "Conflict.Attack.Unspecified" event triggered by "shot" share the same text span "assailant." By considering event co-occurrence, we can capture these relationships. At the sentence level, although event mentions are document-level data, the information and components about events is typically contained within a single sentence. For instance, in the WikiEvents dataset, over 94% of arguments are located in the same sentence as the trigger word. In the RAMS dataset, this proportion exceeds 82%, and in the MLEE dataset, it more than 99%. This indicates the importance of the trigger sentence. To highlight this importance, We take the sentence where the trigger word is located as the core, and construct the structural relationship between all sentences and this sentence. Such structured relationships help the model evaluate the usefulness of all sentences to the current EAE work, and carefully select the useful information and discard the redundant and irrelevant information.

We have adopted PAIE (Ma et al., 2022) as our foundational model, inheriting its encoding and span selection modules. Inspired by (Li and Liang, 2021; Hsu et al., 2023), we utilize prefixes as carriers of information, condensing the aforementioned information into concise prefixes. These prefixes are intelligently integrated into the PLM, aiding in the generation of PAIE's event-oriented context representation and context-oriented template representation. This integration facilitates the model's performance in EAE, and we refer to the complete model as SCEAE.

We summarize our contributions as follow:

• We introduce structure-aware to build document-level structural information, increase the amount of trigger sentence information, and discard the interference caused by redundant information.

• We introduced Co-occurrence-aware to introduce additional information about event co-occurrence to help the model capture semantic boundaries between events.

• Our model, compared to PAIE, has achieved improvements in the Arg-C F1 metric of 2.1%, 2.3%, and 3.2% on RAMS, WikiEvents datasets, and MLEE, respectively. In comparison to the sota model, SCEAE outperforms them by 1.7%, 1.6%, and 0.4% in the Arg-C F1 metric, respectively.

2 Related Work

2.1 Classification-based Event Extraction

Currently, research on EE can be broadly categorized into two approaches: classification-based methods and generation-based methods. Within the classification-based approach, (Chen et al., 2015; Nguyen et al., 2016) require prior extraction of entities and non-entities from the text to form argument sets, which are then used for role classification.(Sheng et al., 2021; Wang et al., 2022; Yang et al., 2023; Xu et al., 2022; Lin et al., 2020; Wadden et al., 2019) no longer rely on separate entity identification tasks and directly extract arguments and perform role classification using labeling techniques. (Li et al., 2020a; Du and Cardie, 2020a) redefine EE as a reading comprehension task and use predefined templates to extract arguments.

2.2 Generation-based Event Extraction

Based on the generative approach, which avoids the problem of error propagation due to its end-to-end nature, it has received wide attention in recent years. (Lu et al., 2021) extract all arguments at once by constraining the generation of generative PLM. (Li et al., 2021; Hsu et al., 2022; Ma et al., 2022; Hsu et al., 2023; He et al., 2023) decode arguments from the generated results using potential argument relationships in the prompts.

2.3 Prefix with Generation-based Extraction

The Prefix-tuning proposed by (Li and Liang, 2021), is a lightweight alternative approach for fine-tuning in natural language generation (NLG) tasks. Due to its significant impact on generative tasks, there has been a recent emergence in the EE community of using prefix to assist extraction tasks. (Liu et al., 2022) introduces a prompt dynamic prefix event extraction method that utilizes prefixes to learn context-specific prefixes for sentence-level event mentions. (Cao et al., 2023) utilizes prefixes in the Cross-Lingual EAE domain by initializing prefixes based on language-universal dendency structures to handle differences between source

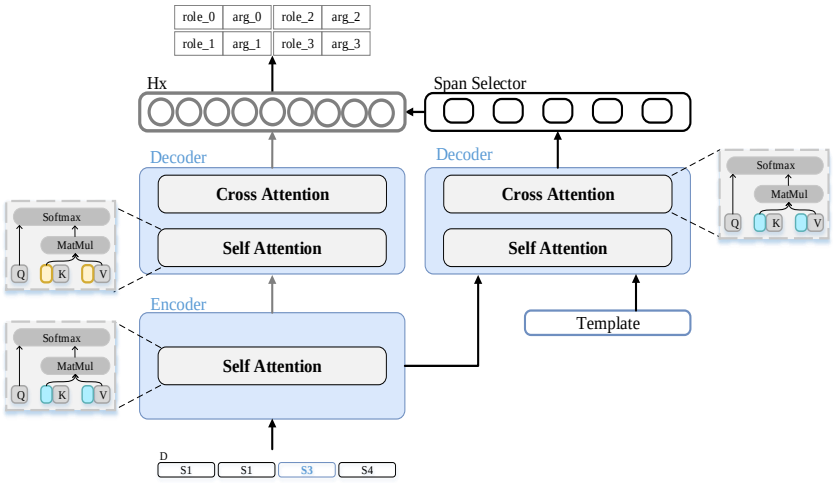

Figure 2: Overview of SCEAE: The entire model has incorporated different Information-aware prefixes in the computation of specific parts of multi-head attention. represents the Co-occurrence-aware prefix, represents the Structure-aware prefix. These prefixes participate in the attention computation of each layer in the PLM's Encoder or Decoder, influencing the generation of the final results.

and target languages. (Hsu et al., 2023) employs prefixes to overcome the heterogeneity between natural language form and structured AMR, thus integrating AMR information into sentence-level EAE.

Our work applies the use of prefixes in document-level EAE. Building upon the PAIE (Ma et al., 2022) model, we incorporate event Co-occurrence-aware information and Structural-aware information of event mentions, aiding the model in capturing semantic boundaries between events and avoiding the dispersal of event attention caused by redundant information.

3 Methodology

In this section, we will provide a description of the basic architecture of SCEAE in section 3.1. Subsequently, in sections 3.2 to 3.4, we will provide detailed explanations of the construction and utilization of each submodule in SCEAE.

3.1 SCEAE

As shown in the Figure 2, given an event mention D, it is input into the model, and the Encoder of the model encodes it to obtain H_X^{enc}, which is then passed through the Decoder of the model to obtain the encoded event-oriented context representation H_X In this case, the Encoder and Decoder have already concatenated prefixes to incorporate additional auxiliary information. Specifically, the Encoder incorporates the Structure-aware prefix, while the Decoder incorporates the Structure-aware prefix and Co-occurrence-aware prefix:

$$H_X^{enc} = Encoder_{Sap}(D), \quad (1)$$

$$H_X = Decoder_{Cap}(H_X^{enc}, H_X^{enc}). \quad (2)$$

Where Sap represents Structure-aware prefix, Cap represents Co-occurrence-aware prefix.

To construct the span selector θ, we need to enable deep interaction between each token of D and the template. First, we input H_X^{enc} and the template together into the Decoder for encoding. Then, we concatenate the Structure-aware prefix to the cross-attention part of the Decoder, namely:

$$H_{pt} = Decoder_{Sap}(H_X^{enc}, Template). \quad (3)$$

Where H_{pt} represents the context-oriented template representation. Based on H_{pt}, we can construct a set of span selector θ specific to each role. Then, θ utilizes the information from H_X to predict one or multiple token spans as arguments for that role.

It should be noted that the two Decoders depicted in the Figure 2 are actually the same Decoder, sharing the same pretrained parameters, but with different prefixes concatenated. We will provide detailed explanations of the construction and application of all modules in the following sections.

3.2 Structure-aware

Compared to sentence-level data, document-level data leads to a significant increase in data length. While this provides more valuable information to

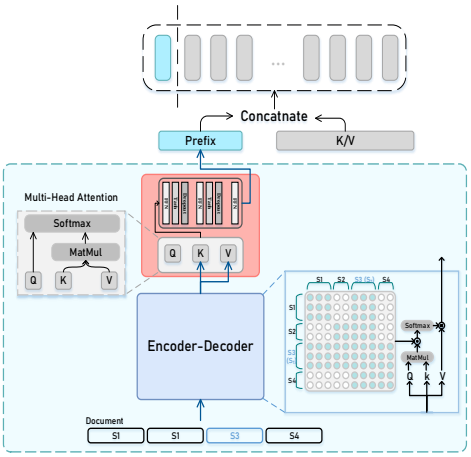

Figure 3: The above figure illustrates the creation of a document-level Structure-aware prefix, where the prefix encapsulates the structured relationship information of the document.

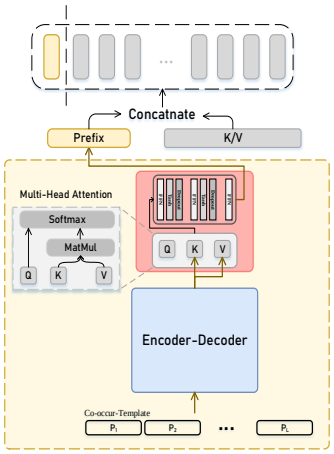

Figure 4: Overview of creating a Co-occurrence-aware prefix. Note that the input to this module is the concatenated templates.

assist the model in extraction, it also introduces redundant information that disperses the model's attention. To address this issue we attempt to construct structured relationships between sentences. Taking the sentence where the trigger is located as the core, we establish document-level structured relationships between this sentence and other sentences by limiting the receptive field of the remaining sentences.

Doc-Structure-aware prefix Self-attention. As shown in Figure 3, we designed a Doc-Structure-aware Self-attention mask, denoted as M_s, which operates at the sentence level and trains the model to be structure-aware for the entire document. Specifically, given a document-level event mention $D = \{S_1, S_2, \ldots, S_m\}$ and the trigger token T_j of the current event to be extracted, where T_j is located in sentence S_T, M_s restricts the receptive field of all sentences except S_T. These sentences can only attend to themselves and S_T, while S_T can attend to all other sentences. We formalize this as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top + M_s}{\sqrt{d_m}}\right)V, \tag{4}$$

$$M_s = \begin{cases} 0, & \text{SEN}(w_i) \in \{\text{SEN}(w_i), \text{SEN}(T_j)\} \\ 0, & \text{SEN}(T_j) \\ -\infty, & Otherwise \end{cases}. \tag{5}$$

where $SEN()$ is the sentence that the word w_i belongs. Finally, we obtain the Structure-aware matrix W_S for the event mention D as follows: $W_S = Encoder - Decoder(D, M_s)$.

Prefix. After constructing the Structure-aware matrix W_S for the event mention D, we condense its structural information into a prefix (Li and Liang, 2021; Hsu et al., 2023). As shown in Figure 3, firstly, we introduce a learnable vector of length l as the Query vector for multi-head attention, where l is a tunable hyperparameter that controls the length of the prefix to be concatenated into the PLM. Next, we utilize W_S as the Key and Value vectors in the multi-head attention calculation, and perform multi-head attention computation with the Q vector. Upon completion of the multi-head attention calculation, we obtain a set of compressed dense vectors P. These vectors P then undergo a series of linear layers for further processing. Finally, we evenly divide the processed P into L segments: $P = \{P_1, P_2, \ldots, P_L\}$, where each segment has a length of l. Here, L represents the number of layers in the PLM. These segmented vectors P_i, $i \in [1, L]$, each with a length of l, can be concatenated into the PLM for computational purposes.

Please note that the Encoder-Decoder used for encoding in this context, as well as the Encoder-Decoder used in the subsequent Co-occurrence-aware prefix generation in Figure 4 and the Encoder and Decoder depicted in Figure 2, all belong to the same pre-trained language model (PLM). They share the same training parameters. However, in this case, no prefixes are added to the PLM.

3.3 Co-occurrence-aware

Document-level data often contains more events, and the relationships between events become more

complex. In order to enhance the model's ability to distinguish the semantic boundaries between events we attempted to incorporate knowledge of event co-occurrence.

Specifically, we introduced event co-occurrence knowledge from two aspects: Co-occurrence-aware Context Labeling, which involves marking the trigger words corresponding to all events mentioned in D, and Co-occurrence-aware prefix generation, which involves extracting template information associated with all events mentioned in D.

Co-occurrence-aware Context Labeling. Given a sentence-level or document-level event mention $D = \{t_1, t_2, \ldots, t_n\}$, where t_i represents the i-th token in the D. Let $E = \{E_0, E_1, \ldots, E_m\}$, denote all the events E_i that appear in D, and m represents the number of events appearing in D. Given the trigger words corresponding to all events in E: $T = \{T_0, T_1, \ldots, T_m\}$, where T_i represents the trigger word for event E_i. For the token span corresponding to the trigger word T_j of the currently extracted event E_j, we annotate it in D using special tokens $< t - -1 >$ and $< /t - -1 >$. Thus, the modified D will be $D = \{t_1, t_2, \ldots, < t - -1 > T_j < /t - -1 >, \ldots, t_n\}$. For other trigger words T_k corresponding to events that exist in D, we will annotate them using $< t - k >$ and $< /t - k >$ based on their positions in D in the order of appearance. Here, k starts from 0, and for each annotated trigger word, k is incremented by 1.

Co-occurrence-aware prefix. In the prompt-based generative approach, templates are directly associated with event types, and the interplay of role-related information depicted in the templates is regarded as a key factor (Hsu et al., 2022) in facilitating EE and EAE. To fully leverage this information, as shown in Figure 4, we attempt to create prefixes from the template information corresponding to all events mentioned in D, and then integrate these prefixes into the PLM to assist in EAE. Specifically, we start by concatenating the templates $P = \{P_0, P_1, \ldots, P_L\}$ corresponding to all events $E = \{E_0, E_1, \ldots, E_L\}$ that occur in the current event mention D. Next, we encode this concatenated template sequence using an Encoder-Decoder PLM, resulting in a dense vector representation W_C that captures the templates associated with co-occurring events. Finally, following the same approach described in section 3.2, we integrate the information from W_C into prefix.

It is important to note that the Encoder-Decoder PLM used for encoding does not have any prefixes added.

3.4 Span Selection

After obtaining the context-oriented template representation H_pt, we extract the slot representation ψ_k corresponding to the pre-defined roles from H_pt, where k represents the k-th slot. Then, we convert ψ_k into a span selector specific to that slot θ_k (Ma et al., 2022; Du and Cardie, 2020b). Next, apply the span selector θ_k directly to the event-oriented context representation to determine argumentH_X to determine the argument's token span $[p_k^{(start);p_k^{(end)}}]$.

$$\begin{aligned}
\psi_k^{(start)} &= \psi_k \circ w^{(start)} \in R^h, \\
\psi_k^{(end)} &= \psi_k \circ w^{(end)} \in R^h, \\
\text{logit}_k^{(start)} &= \psi_k^{(start)} H_X \in R^L, \\
\text{logit}_k^{(end)} &= \psi_k^{(end)} H_X \in R^L, \\
p_k^{(start)} &= \text{Softmax}(\text{logit}_k^{(start)}) \in R^L, \\
p_k^{(end)} &= \text{Softmax}(\text{logit}_k^{(end)}) \in R^L.
\end{aligned} \tag{6}$$

Where $\theta = [w^{(start)}; w^{(end)}] \in R^{h \times 2}$ is a learnable parameter matrix shared by all span selectors, o represents element-wise multiplication. $\theta_k = [\psi_k^{(start)}; \psi_k^{(end)}]$ is the span selector specific to the slot corresponding to the role.

We define the loss function L as follows:

$$\begin{aligned}
\mathcal{L}_k(X) &= -(\log p_k^{(start)}(s_k) + \log p_k^{(end)}(e_k)), \\
\mathcal{L} &= \sum_{X \in D} \sum_k \mathcal{L}_k(X).
\end{aligned} \tag{7}$$

where D ranges over all context in dataset and k ranges over all slots in prompt for X.

During the inference phase, we predefine spans that cover all possible spans within a predefined length and include a special span (0, 0) to represent the absence of any corresponding argument. Then, we utilize the span selector θ_k to compute scores for all spans using the following method:

$$\text{score}_k(i, j) = \text{logit}_k^{(start)}(i) + \text{logit}_k^{(end)}(j). \tag{8}$$

Where i and j represent the start and end indices of each span in the set of spans.

Based on the scores, we determine the predicted final span by selecting the span with the highest

score.

$$(\widehat{s_k}, \widehat{e_k}) = \arg \max_{(i,j) \in \mathcal{C}} \text{score}_k(i, j). \quad (9)$$

For the issue of multiple arguments of the same role, we utilize the Hungarian algorithm to fine-tune our model (Kuhn, 1955; Ma et al., 2022). For the problem of allocating multiple slots corresponding to a single role, we employ Bipartite Matching (Carion et al., 2020; Yang et al., 2021; Ma et al., 2022).

4 Experiments

4.1 Datasets

We conducted comprehensive ablation and comparative experiments on SCEAE using three datasets: RAMS(Ebner et al., 2020), WikiEvents(Li et al., 2021), and MLEE(Pyysalo et al., 2012). RAMS and WikiEvents are the latest datasets widely used for document-level EE/EAE models. RAMS and WikiEvents are derived from news events and MLEE focuses on the biomedical domain. Since the MLEE dataset does not have a separate validation set, we used the training set as our validation set during model training. We followed the data preprocessing techniques from (Ma et al., 2022; He et al., 2023; Hsu et al., 2023) for the document-level datasets RAMS, WikiEvents, and MLEE.

We incorporated the prompts proposed in (Ma et al., 2022) specifically designed for the RAMS and WikiEvents datasets. Additionally, we utilized the prompts suggested in (He et al., 2023) tailored for the MLEE dataset. Appendix A provides a detailed introduction to the four mentioned datasets.

4.2 Experiment Setups and Evaluation Metrics

We follow previous works for prompt-based generative event extraction models using PLM with Encoder-Decoder Transformer structure, such as BART (Lewis et al., 2020) and T5 (Raffel et al., 2020). MTCM uses the BART model as a PLM, which is a standard Transformer-based PLM consisting of both an Encoder and a Decoder. Detailed experiment setups and hyperparameters are listed in Appendix B.

We consider the same evaluation criteria in prior works (Li et al., 2021; Hsu et al., 2022; Ma et al., 2022; He et al., 2023) for all dataset. Since it is a generative task, we place greater emphasis on the F1 score. Therefore, in this experiment, we will report the F1 score for argument identification (Arg-I) and argument classification (Arg-C).

• Arg-I: an argument is correctly identified from event mention.

• Arg-C: an argument is correctly classified if its offset and the role's label both match the ground truth.

4.3 Baseline Method

We will categorize the models used for comparison with SCEAE into two groups. The first group is about classification-based methods. The classification-based models include:

• EEQA (Du and Cardie, 2020a): redefines the EE task as a question-answering task.

• TSAR (Xu et al., 2022): the model use Two-Stream Abstract meaning Representation to EAE.

The second group is about generation-based methods.

• BART-Gen(Li et al., 2021): redefines the EE task as a seq-to-seq conditional generation.

• PAIE (Ma et al., 2022): the model utilizes a span selector for decoding and extracting arguments.

• TabEAE (He et al., 2023): the model extends the PAIE into a non-autoregressive generation framework.

4.4 Main Resuilts

We evaluate the proposed model SCEAE and baseline methods under all benchmarks. As shown in Table 1, comparing to the baseline model PAIE, SCEAE demonstrates comprehensive improvements across all datasets. On the RAMS dataset, SCEAE achieves a 2.3% gain and a 2.2% gain in the Arg-I and Arg-C metric. On the WikiEvents dataset, SCEAE shows a 2.0% improvement in the Arg-I metric and a 2.3% improvement in the Arg-C metric. The largest gains are observed on the MLEE dataset, where SCEAE achieves significant improvements of 3.0% and 3.2% in the Arg-I and Arg-C metrics, respectively.

In comparison to the previous state-of-the-art models, SCEAE also achieves comprehensive improvements across the three document-level benchmarks. Specifically, on the RAMS dataset, SCEAE outperforms TabEAE(m2s) with a 1.4% improvement in the Arg-I metric and a 1.8% improvement in the Arg-C metric. On the WikiEvents dataset, SCEAE surpasses TabEAE(m2s) with a 1.2% improvement in the Arg-I metric and a 1.6% improvement in the Arg-C metric. On the MLEE dataset,

Model	PLM	RAMS		WikiEvents		MLEE	
		Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C
classification-based							
EEQA	BERT	48.7	46.7	56.9	54.5	68.4	66.7
TSAR	BERT	56.1	51.2	70.8	65.5	72.3	71.3
generation-based							
BART-Gen	BART	51.2	47.1	66.8	62.4	71.0	69.8
TabEAE(s)	RoBERTa	56.2	51.4	69.7	64.9	-	-
TabEAE(m)	RoBERTa	55.9	50.9	70.3	64.6	74.0	72.9
PAIE	BART	55.3	51.0	68.9	64.2	71.3	70.1
SCEAE	BART	**57.5**	**53.1**	**70.9**	**66.5**	**74.3**	**73.3**

Table 1: The table above presents a comparison of the Arg-I F1 and Arg-C F1 metrics between SCEAE and all baseline methods on three datasets. All experiments utilized a large-scale PLM with 24 Transformer layers. The highest scores are highlighted in bold, and the second-best scores are underlined. TabEAE(s) means the use of a Multi-Single Training-inference Scheme, TabEAE(m) means the use of a Multi-Multi Training-inference Scheme

SCEAE achieves a 0.3% improvement in the Arg-I metric and a 0.4% improvement in the Arg-C metric compared to TabEAE(m2m). These results highlight the powerful extraction capabilities of SCEAE. ‘

5 Analysis

5.1 Capturing the Event Semantic Boundary

We investigated the ability of SCEAE to capture event semantic boundaries from two perspectives, similar to TabEAE: inter-event semantics and intra-event semantics.

Inter-event semantics. We conducted an analysis on the WikiEvents and MLEE datasets, dividing the test sets into two categories: those containing overlapping events and those without overlapping events. Overlapping events refer to multiple events that share the same token span as arguments. As shown in the Table 2, SCEAE outperforms PAIE in both benchworks, with the highest improvement occurring when handling the Overlap events in the MLEE dataset, where the Arg-C F1 score improves by 4.5%. Even when compared to TabEAE, which utilizes different reasoning methods, SCEAE still achieves excellent performance. This indicates that SCEAE is effective in distinguishing the semantic boundaries between overlapping events.

Intra-event semantics. we measure the distance between roles and trigger words. Since a role may correspond to multiple different arguments, the model predicts all the arguments corresponding to a role at once. We calculate the distance by subtracting the head token index of an argument from the head token index of its corresponding trigger word. We define the distance (d) between a role and a trigger word as the maximum distance between all the arguments within that role and the trigger word. From the Figure 5, it can be observed that PAIE struggles to handle roles with distances (d) greater than or equal to 15 or less than or equal to -15. SCEAE shows significant improvements compared to PAIE when dealing with roles at long distances. This indicates that SCEAE is capable of effectively capturing the arguments surrounding event boundaries.

Model	WikiEvents		MLEE	
	N_O	Overlap	N_O	Overlap
	296	69	734	1460
TabEAE(s)	65.4	63.0		
TabEAE(m)			77.0	67.6
PAIE	63.9	65.0	75.7	63.4
SCEAE	66.0	68.4	77.8	67.9

Table 2: The table above compares the performance of EAE models in extracting the arguments of overlapping events. We only measure the Arg-C F1 metric. "N_O" indicates "no overlap," and the number below represents the quantity of event mentions without overlap in the corresponding dataset's test set.

5.2 Structure-aware for Document

To validate that the performance of Structure-aware, we conducted experiments on the RAMS dataset, which has the highest proportion of arguments and trigger words in different sentences. As shown in the Table 4, we defined the D as the distance between the argument and the trigger word in terms of sentences. We classified roles where all the

Model	RAMS		WikiEvents		MLEE	
	Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C
w/o str & occur	55.3	51.0	68.9	64.2	71.3	70.1
only str	55.8	52.0	70.5	64.8	72.0	70.9
only occur	55.9	51.6	70.5	65.9	73.9	72.9
SCEAE	57.5	53.1	70.9	66.5	74.3	73.3

Table 3: The table above shows SCEAE's ablation experiments on all datasets. str: Structure-aware prefix. occur: Co-occur-aware prefix. Str & occur: Structure-aware and Co-occur-aware prefixes.

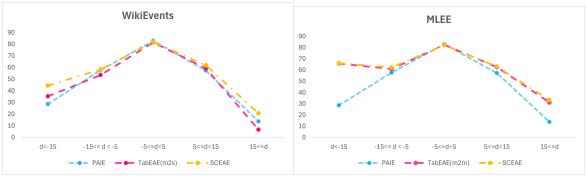

Figure 5: The Figures above displays the performance of different EAE models in extracting arguments at different distances from the triggers on the WikiEvents and MLEE datasets. Negative numbers represent positions to the left of the trigger words, while positive numbers represent positions to the right of the trigger words. We only measure the Arg-C F1 metric.

Model	RAMS		
	$D = 0$	$D \neq 0$	All
PAIE	58.7	35.3	51.0
TabEAE(s)	61.2	31.8	51.4
SCEAE	61.9	35.5	53.1

Table 4: The table above shows the performance of different EAE models on the RAMS datasets with varying sentence distances between triggers and arguments. "All" refers to all the data in the test set. We only measure the Arg-C F1 metric.

arguments corresponded to the same sentence as $D = 0$. If at least one argument within a role was in a different sentence from the trigger word, the role was classified as $D \neq 0$. From Table 4, it can be observed that SCEAE achieves a gain of 3.2% over PAIE and outperforms TabEAE by 0.7% in the $D = 0$ cases. On the other hand, SCEAE achieves a gain of 0.2% over PAIE and outperforms TabEAE by 3.7% in the $D \neq 0$ cases. The improvements in the both aspects in SCEAE enable it to achieve outstanding performance, surpassing PAIE by 2.1% and TabEAE by 1.7%. This indicates the effectiveness of enhancing the attention on trigger sentence in the model for argument extraction.

5.3 Ablation Studies

As shown in the Table 3, we conducted comprehensive ablation experiments on SCEAE. It can be seen from the Table 3 that certain datasets experience significant improvements due to sensitivity to specific knowledge-aware. For example, the Occur-aware prefix improves the Arg-I F1 and Arg-C F1 metrics by 1.6% and 1.7% respectively on WikiEvents and 2.6% and 1.8% respectively on MLEE datasets, while the Structure-aware prefix improves the Arg-I F1 and Arg-C F1 metrics by 0.6% and 1.0% respectively on the RAMS dataset. In the SCEAE, it can be observed that when integrating two different prefixes, the models retain sensitivity to individual knowledge-aware without being disturbed by the incorporation of additional knowledge-aware, and resulting in substantial improvements in performance across all datasets for SCEAE.

6 Conclusion

The increasing length of document-level data brings about more complex events and redundant information, posing challenges for document-level Event Argument Extraction (EAE) in distinguishing semantic boundaries between events and redundant information distracts attention from event. To address these issues, we propose SCEAE. SCEAE incorporates the Co-occurrence-aware prefix to help the model capture event semantic boundaries, and the Structure-aware prefix to build structured information of the entire document. Experimental results demonstrate that our model achieves improvements on three datasets: RAMS, WikiEvents, and MLEE. It achieves a new state-of-the-art performance.

7 Limitation

Indeed, there are still areas in our model that need further improvement.

• The limitation of manually designing templates remains a significant issue in prompt-based generative methods. This manual design restricts the flexibility and adaptability of the model, as it relies on predefined templates for generating outputs.

• The model is not able to directly process extremely long data (data length greater than 250). Typically, preprocessing is required to handle such data, which has been a longstanding challenge in the Event Extraction (EE) community.

References

Pengfei Cao, Zhuoran Jin, Yubo Chen, Kang Liu, and Jun Zhao. 2023. Zero-shot cross-lingual event argument extraction with language-oriented prefix-tuning. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 12589–12597. AAAI Press.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. *CoRR*, abs/2005.12872.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 167–176. The Association for Computer Linguistics.

Tarcísio Souza Costa, Simon Gottschalk, and Elena Demidova. 2020. Event-qa: A dataset for event-centric question answering over knowledge graphs. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 3157–3164. ACM.

Xinya Du and Claire Cardie. 2020a. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 671–683. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020b. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 671–683. Association for Computational Linguistics.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8057–8077. Association for Computational Linguistics.

Yuxin He, Jingyue Hu, and Buzhou Tang. 2023. Revisiting event argument extraction: Can EAE models learn better when being aware of event co-occurrences? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 12542–12556. Association for Computational Linguistics.

I-Hung Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. 2022. DEGREE: A data-efficient generation-based event extraction model. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 1890–1908. Association for Computational Linguistics.

I-Hung Hsu, Zhiyu Xie, Kuan-Hao Huang, Prem Natarajan, and Nanyun Peng. 2023. AMPERE: amr-aware prefix for generation-based event argument extraction model. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 10976–10993. Association for Computational Linguistics.

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, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020a. Event extraction as multi-turn question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pages 829–838. Association for Computational Linguistics.

Manling Li, Alireza Zareian, Ying Lin, Xiaoman Pan, Spencer Whitehead, Brian Chen, Bo Wu, Heng Ji, Shih-Fu Chang, Clare R. Voss, Daniel Napierski, and Marjorie Freedman. 2020b. GAIA: A fine-grained

multimedia knowledge extraction system. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020, Online, July 5-10, 2020*, pages 77–86. Association for Computational Linguistics.

Sha Li, Heng Ji, and Jiawei Han. 2021. Document-level event argument extraction by conditional generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 894–908. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7999–8009. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022. Dynamic prefix-tuning for generative template-based event extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5216–5228. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2795–2806. Association for Computational Linguistics.

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. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 6759–6774. Association for Computational Linguistics.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 300–309. The Association for Computational Linguistics.

Sampo Pyysalo, Tomoko Ohta, Makoto Miwa, Han-Cheol Cho, Junichi Tsujii, and Sophia Ananiadou. 2012. Event extraction across multiple levels of biological organization. *Bioinform.*, 28(18):575–581.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Jiawei Sheng, Shu Guo, Bowen Yu, Qian Li, Yiming Hei, Lihong Wang, Tingwen Liu, and Hongbo Xu. 2021. Casee: A joint learning framework with cascade decoding for overlapping event extraction. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 164–174. Association for Computational Linguistics.

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

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5783–5788. Association for Computational Linguistics.

Sijia Wang, Mo Yu, Shiyu Chang, Lichao Sun, and Lifu Huang. 2022. Query and extract: Refining event extraction as type-oriented binary decoding. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 169–182. Association for Computational Linguistics.

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. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 5025–5036. Association for Computational Linguistics.

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, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6298–6308. Association for Computational Linguistics.

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. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 12876–12889. Association for Computational Linguistics.

Tianran Zhang, Muhao Chen, and Alex A. T. Bui. 2020. Diagnostic prediction with sequence-of-sets representation learning for clinical events. In *Artificial Intelligence in Medicine - 18th International Conference on Artificial Intelligence in Medicine, AIME 2020, Minneapolis, MN, USA, August 25-28, 2020, Proceedings*, volume 12299 of *Lecture Notes in Computer Science*, pages 348–358. Springer.

A Dataset statistics

RAMS, commonly used for document-level Event Extraction (EE) and is derived from English online news. Each document in the dataset comprises five sentences. Since the original dataset is stored on an event-by-event basis, we followed the (He et al., 2023) method to merge data from different events within the same document while retaining the 'sents' field. We used the original train/dev/test splits.

WikiEvents, typically used for document-level EE, is collected from English articles in Wikipedia. In our experiments, we used the exact argument annotations. We also employed the data handling approach for excessively long data (length greater than 250) described in (He et al., 2023).

MLEE, commonly used for document-level EE, is derived from abstracts of biomedical publications. We preprocessed the data using (Trieu et al., 2020) method and then applied the data handling approach for excessively long data (length greater than 250) as described in (He et al., 2023). Since the dataset does not have a separate validation set, we used the training set for model training.

Statistics of the datasets and their detailed information are recorded in the table 5.

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

Table 5: The table above shows the basic information for the all datasets, where Args stands for Arguments.

B Experiment Setups

We utilized the BART-Large model, provided by Facebook on the Huggingface website, as the pretrained language model for SCEAE. The model has approximately 406 million parameters. We conducted the training of SCEAE and replicated other paper's experiments using a single NVIDIA A40 Tensor Core GPU with a capacity of 45GB. We selects the best model based on the development set results. For optimization, we employed the AdamW (Loshchilov and Hutter, 2019) optimizer, setting the learning rate l to 40, as described in section 3.2. The batch size was set to 4. The average length of time spent training the model was five hours.