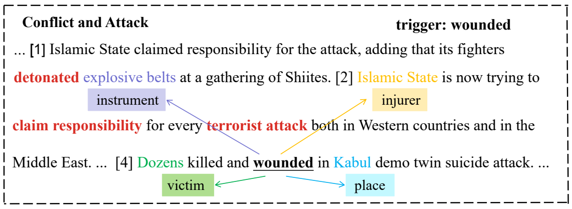
Enhancing Document-level Event Argument Extraction with Contextual Clues and Role Relevance

Wanlong Liu[1], **Shaohuan Cheng**[1], **Dingyi Zeng**[1], **Hong Qu**[1]*
[1]University of Electronic Science and Technology of China, Chengdu, China
{liuwanlong, shaohuancheng, zengdingyi}@std.uestc.edu.cn, hongqu@uestc.edu.cn

Abstract

Document-level event argument extraction poses new challenges of long input and cross-sentence inference compared to its sentence-level counterpart. However, most prior works focus on capturing the relations between candidate arguments and the event trigger in each event, ignoring two crucial points: a) non-argument contextual clue information; b) the relevance among argument roles. In this paper, we propose a SCPRG (**S**pan-trigger-based **C**ontextual **P**ooling and latent **R**ole **G**uidance) model, which contains two novel and effective modules for the above problem. The **S**pan-**T**rigger-based **C**ontextual **P**ooling (STCP) adaptively selects and aggregates the information of non-argument clue words based on the context attention weights of specific argument-trigger pairs from pre-trained model. The **R**ole-based **L**atent **I**nformation **G**uidance (RLIG) module constructs latent role representations, makes them interact through role-interactive encoding to capture semantic relevance, and merges them into candidate arguments. Both STCP and RLIG introduce no more than 1% new parameters compared with the base model and can be easily applied to other event extraction models, which are compact and transplantable. Experiments on two public datasets show that our SCPRG outperforms previous state-of-the-art methods, with 1.13 F1 and 2.64 F1 improvements on RAMS and WikiEvents respectively. Further analyses illustrate the interpretability of our model.



Conflict and Attack **trigger: wounded**

... [1] Islamic State claimed responsibility for the attack, adding that its fighters **detonated** explosive belts at a gathering of Shiites. [2] Islamic State is now trying to **claim responsibility** for every **terrorist attack** both in Western countries and in the Middle East. ... [4] Dozens killed and **wounded** in Kabul demo twin suicide attack. ...

Figure 1: A document from RAMS (Ebner et al., 2020) dataset. Event *Conflict and Attack* is triggered by *wounded*, with four arguments of different roles scattering across the document. Words in red are non-argument clue words meaningful for argument extraction.

1 Introduction

Event argument extraction (EAE) aims to identify the arguments of events formed as entities in text and predict their roles in the related event. As the key step of event extraction (EE), EAE is an important NLP task with widespread applications, such as recommendation systems (Li et al., 2020) and dialogue systems (Zhang et al., 2020a) for presenting unstructured text containing event information in structured form. Compared with previous works (Liu et al., 2018; Wadden et al., 2019; Tong et al., 2020) focusing on sentence-level EAE, more and more recent works tend to explore document-level EAE (Wang et al., 2022b; Yang et al., 2021; Xu et al., 2022), which needs to solve long-distance dependency (Ebner et al., 2020) and cross-sentence inference (Li et al., 2021) problems. Therefore, many works (Zhang et al., 2020b; Pouran Ben Veyseh et al., 2022) try to construct graphs based on heuristic rules (Xu et al., 2021; Liu et al., 2022a) or syntactic structures (Xu et al., 2022) and model logical reasoning with Graph Neural Networks (Kipf and Welling, 2016; Zeng et al., 2023b,a). However, all of state-of-the-art works ignore two crucial points: (a) the non-argument clue information; (b) the relevance among argument roles.

Non-argument clues are contextual text except target arguments that can provide important guiding information for the prediction of many complex argument roles. For example, in Figure 1, for the event *Conflict and Attack*, non-argument clues *detonated*, *claim responsibility* and *terrorist attack* can provide significant clue information for identifying arguments *explosive belts* and *Islamic State*. However, many previous works (Li et al., 2021; Xu et al., 2022) only utilize pre-trained transformer-

*Corresponding author: Hong Qu

Findings of the Association for Computational Linguistics: ACL 2023, pages 12908–12922
July 9-14, 2023 ©2023 Association for Computational Linguistics

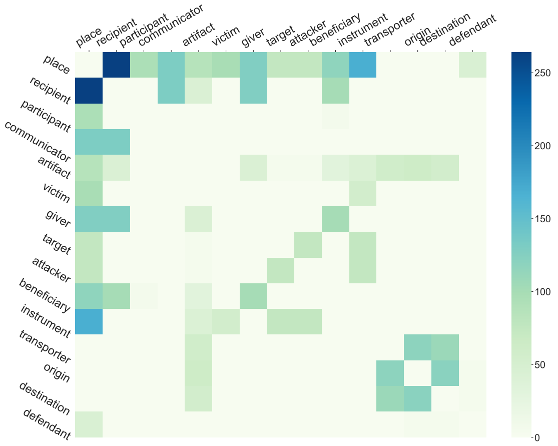


Figure 2: Visualization of the co-occurrence frequency between 15 most frequent roles on RAMS test set. we have reserved and set the co-occurrence number with itself to zero. The full figure is included in Appendix B.

based encoder to obtain global context information implicitly, ignoring that for different arguments appearing in events, they should pay attention to context information highly relevant to the entity (Zhou et al., 2021) and target event (Ebner et al., 2020). Therefore in this paper, we design a **S**pan-**T**rigger-based **C**ontextual **P**ooling (STCP) module, which merges the information of non-argument clues for each argument-trigger pair based on the their contextual attention product from pre-trained model, enhancing the candidate argument representation with additional relevant context information.

Some argument roles have close semantic relevance that is beneficial for argument extraction. For example, in Figure 1, there is close semantic relevance between roles *injurer* and *victim*, which can provide significant information guidance for the argument extraction of these two roles in the target event *Conflict and Attack*. Moreover, many roles co-occur in multiple events (Ebner et al., 2020; Li et al., 2021), which may have close semantic relevance. Specifically, we count and visualize the frequency of co-occurrence between 15 most frequent roles in RAMS dataset in Figure 2. For example, roles *attacker*, *target* and *instrument* co-occur frequently, demonstrating that they are more semantically relevant than other roles. In this paper, we propose a **R**ole-based **L**atent **I**nformation **G**uidance (RLIG) module, consisting of role-interactive encoding and role information fusion. Specifically, we design a role-interactive encoder with roles added into the input sequence, where role embeddings can not only learn latent semantic information of roles, but capture semantic relevance among roles. The latent role embeddings are then merged into candidate arguments through pooling and concatenating operations, providing information guidance for document-level EAE.

In this paper, we propose an effective document-level EAE model named SCPRG (**S**pan-trigger-based **C**ontextual **P**ooling and **R**ole-based latent information **G**uidance) containing STCP module and RLIG module for the the aforementioned two problems respectively. Notably, these two modules leverage the well-learned attention weights from the pre-trained language model with no more than 1% new parameters introduced and are easily applied to other event extraction models, which are compact and transplantable. Moreover, we try to eliminate noise information by excluding argument-impossible spans. Our contributions are summarized as follows:

- We propose a span-trigger-based contextual pooling module, which adaptively selects and aggregates the information of non-argument clues, enhancing the candidate argument representation with relevant context information.

- We propose a role-based latent information guidance module, which provides latent role information guidance containing semantic relevance among roles.

- Extensive experiments show that SCPRG outperforms previous start-of-the-art models, with 1.13 F1 and 2.64 F1 improvements on public RAMS and WikiEvents (Li et al., 2021) datasets. We further analyse the attention weights and latent role representations, which shows the interpretability of our model[1].

2 Method

We formulate document-level event argument extraction as a multi-class classification problem. Given a document $\mathcal{D}$ consisting of N words, i.e. $\mathcal{D} = \{w_1, w_2, ..., w_N\}$, pre-defined event types set $\mathcal{E}$, the corresponding role set $\mathcal{R}_e$ and trigger $t \in \mathcal{D}$ for each event $e \in \mathcal{E}$, this task aims at predicting all (r, s) pairs for each event in document $\mathcal{D}$, where $r \in \mathcal{R}_e$ is an argument role for event $e \in \mathcal{E}$ and $s \subseteq \mathcal{D}$ is a contiguous text span in $\mathcal{D}$. Following (Ebner et al., 2020; Xu et al., 2022), we

[1] Our implementation is available at https://github.com/LWL-cpu/SCPRG-master

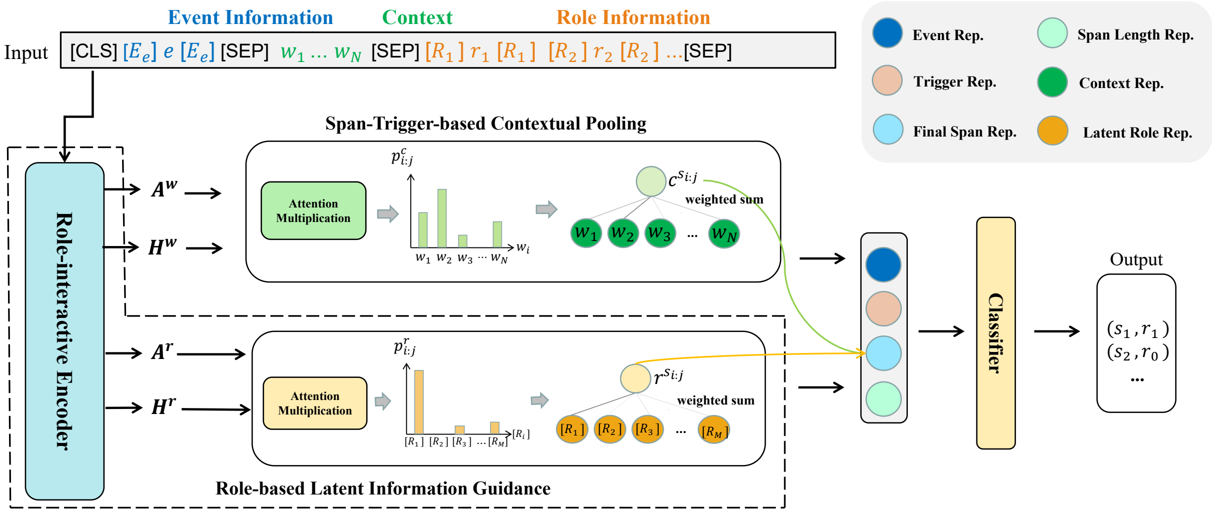


Figure 3: The main architecture of SCPRG. The input sequence with roles is fed into the role-interactive encoder, with context representations, role representations and attention heads as output. STCP adaptively fuses non-argument contextual clues into a context vector based on the attention product between the trigger and arguments. RLIG constructs latent role embeddings through role-interactive encoding and fuses them into a latent role vector by pooling operation. The context vector and latent role vector are merged into the final span representation and the classification module predicts argument roles for all candidate spans.

extract event arguments for each event in a document independently and Figure 3 shows the overall architecture of our SCPRG.

2.1 Role-interactive Encoder

Role Type Representation In order to capture semantic relevance among roles, we add role type information into the input sequence and make interaction among context and roles by multi-head attention, which obtains context and role representations in a shared knowledge space. Specifically, we construct latent embeddings of roles with different special tokens [2] in the pre-trained model, where each role type has a specific latent representation. On account that role names also contain valuable semantic information (Wang et al., 2022b), we wrap role names with special role type tokens and take the embedding of the start special toke as the role embedding. Taking the role *Place* as an example, we finally represent it as $[R_0]$ *Place* $[R_0]$, where $[R_0]$ is the special role type token of *Place*.

Role-interactive Encoding For the input document $\mathcal{D} = \{w_1, w_2, ..., w_N\}$, the target event e and the corresponding role set $\mathcal{R}_e = \{r_1, r_2, r_3, ...\}$, we concatenate them into a sequence as follows:

$$\begin{aligned} S = &\ [\text{CLS}]\ [E_e]\ e\ [E_e]\ [\text{SEP}]\ w_1 \dots w_N\ [\text{SEP}] \\ &\ [R_1]\ r_1\ [R_1]\ [R_2]\ r_2\ [R_2]\ \dots\ [\text{SEP}], \end{aligned}$$

where $[E_e]$ is the special event token of event e. $[R_1]$ and $[R_2]$ are the special role type tokens of r_1 and r_2. We use the last [SEP] to represent none category. Next, we leverage the pre-trained language model as an encoder to obtain the embedding of each token as follows:

$$\mathbf{H}^s = \text{Encoder}(S). \tag{1}$$

Then we can obtain event representation $\mathbf{H}^e \in \mathbb{R}^{1 \times d}$ of the start $[E_e]$, context representation $\mathbf{H}^w \in \mathbb{R}^{l_w \times d}$, and role representation $\mathbf{H}^r \in \mathbb{R}^{l_r \times d}$ respectively from $\mathbf{H}^s$, where l_w is the length of word pieces list and l_r is the length of role list. For input sequences longer than 512, we leverage a dynamic window to encode the whole sequence and average the overlapping token embeddings of different windows to obtain the final representation.

Significantly, through role-interactive encoding, the role embeddings can capture semantic relevance and adapt to the target event and context, which better guides the argument extraction.

2.2 Span-Trigger-based Contextual Pooling

Argument-impossible Spans Exclusion In order to eliminate noise information of useless spans, we reduce the number of candidate spans by excluding some argument-impossible spans, e.g. spans with comma in the middle. With such improvement, we reduce a quarter of candidate spans on

[2] In our implement, we utilize [unused] tokens for BERT (Devlin et al., 2019) and add special tokens for RoBERTa (Liu et al., 2019).

average and make our model pay attention to candidate spans with useful information.

Span-Trigger-based Contextual Pooling For a candidate span ranging from w_i to w_j, most previous span-based methods(Zhang et al., 2020b; Xu et al., 2022) represent it through the average pooling of the hidden state of tokens within this span: $\frac{1}{j-i+1}\sum_{k=i}^{j}\mathbf{h}_k^w$, where $\mathbf{h}_k^w$ is the k^{th} token embedding from $\mathbf{H}^w$.

However, average pooling representation ignores the significant clue information of other non-argument words. Although self-attention mechanism of the pre-trained encoder can model token-level interaction, such global interaction is specific to the event and candidate arguments. Therefore, we propose to select and fuse useful contextual information highly related to each tuple consisting of a candidate span and the event trigger word, i.e. $(s_{i:j}, t)$. We directly utilize the attention heads of pre-trained transformer-based encoder for span-trigger-based contextual pooling, which transfers the well-learned dependencies from the pre-trained language model without learning new attention layers from scratch (Zhou et al., 2021).

Specifically, we use the token-level attention heads $\mathbf{A}^w \in \mathbb{R}^{H\times l_w\times l_w}$ of context from the last transformer layer in the pre-trained language model. Then we can obtain the context attention $\mathbf{A}_{i:j}^C \in \mathbb{R}^{l_w}$ of each candidate span ranging from w_i to w_j with average pooling:

$$\mathbf{A}_{i:j}^C = \frac{1}{H(j-i+1)}\sum_{h=1}^{H}\sum_{m=i}^{j}\mathbf{A}_{h,m}^w. \tag{2}$$

Then for span-trigger pair $(s_{i:j}, t)$, we obtain the contextual clue information $\mathbf{c}^{s_{i:j}} \in \mathbb{R}^d$ that are important to candidate span by multiplying the attentions followed by normalization:

$$\begin{aligned}\mathbf{p}_{i:j}^c &= softmax(\mathbf{A}_{i:j}^C \cdot \mathbf{A}_t^C\ ),\\ \mathbf{c}^{s_{i:j}} &= \mathbf{H}^w\ \mathbf{p}_{i:j}^c,\end{aligned} \tag{3}$$

where $\mathbf{A}_t^C \in \mathbb{R}^{l_w}$ is the contextual attention of trigger t and $\mathbf{p}_{i:j}^c \in \mathbb{R}^{l_w}$ is the computed attention weight vector for context.

2.3 Role-based Latent Information Guidance

RLIG module constructs latent role embeddings through role-interactive encoding in Sec. 2.1 and performs role information fusion through pooling operation, which provides valuable latent role information guidance.

Role Information Fusion In order to make each candidate argument get the useful information guidance of roles, we modify our span-trigger-based contextual pooling method to select role information adaptively. We get the latent role information $\mathbf{r}^{s_{i:j}} \in \mathbb{R}^d$ for $s_{i:j}$ through contextual pooling, by modifying the operation in Eq. 2 and Eq. 3:

$$\begin{aligned}\mathbf{A}_{i:j}^R &= \tfrac{1}{H(j-i+1)}\textstyle\sum_{h=1}^{H}\sum_{m=i}^{j}\mathbf{A}_{h,m}^r,\\ \mathbf{p}_{i:j}^r &= softmax(\mathbf{A}_{i:j}^R \cdot \mathbf{A}_t^R\ ),\\ \mathbf{r}^{s_{i:j}} &= \mathbf{H}^r\ \mathbf{p}_{i:j}^r,\end{aligned} \tag{4}$$

where $\mathbf{A}^r \in \mathbb{R}^{H\times l_w\times l_r}$ are attention heads of roles from the last transformer layer in the pre-trained language model. $\mathbf{A}_{i:j}^R \in \mathbb{R}^{l_r}$ is the role attention for each candidate span and $\mathbf{A}_t^R \in \mathbb{R}^{l_r}$ is the role attention of trigger t. $\mathbf{p}_{i:j}^r \in \mathbb{R}^{l_r}$ is the computed attention weight vector for roles.

For a candidate span $s_{i:j}$, we fuse the average pooling representation, contextual clue information $\mathbf{c}^{s_{i:j}}$ and latent role information $\mathbf{r}^{s_{i:j}}$ as follows:

$$\mathbf{s}_{i:j} = tanh(\mathbf{W}_1[\frac{1}{j-i+1}\sum_{k=i}^{j}\mathbf{h}_k^w;\mathbf{c}^{s_{i:j}};\mathbf{r}^{s_{i:j}}]), \tag{5}$$

where $\mathbf{W}_1 \in \mathbb{R}^{3d\times d}$ is learnable parameter.

2.4 Classification Module

Boundary Loss Since we extract arguments in span level, whose boundary may be ambiguous, we construct start and end representation with fully connected neural networks to enhance the representation of candidate spans: $\mathbf{H}^{start} = \mathbf{W}^{start}\mathbf{H}^s, \mathbf{H}^{end} = \mathbf{W}^{end}\mathbf{H}^s$, where $\mathbf{H}^s$ is the hidden representation of input sequence S. On this basis, we enhance the start and end representation by integrating context and role information with span-trigger-based contextual pooling as follows:

$$\begin{aligned}\mathbf{z}_{i:j}^{start} &= \mathbf{H}^{start}\ \mathbf{p}_{i:j},\\ \mathbf{z}_{i:j}^{end} &= \mathbf{H}^{end}\ \mathbf{p}_{i:j},\\ \mathbf{h}_{i:j}^{start} &= tanh(\mathbf{W}_2[\mathbf{h}_i^{start};\mathbf{z}_{i:j}^{start}]),\\ \mathbf{h}_{i:j}^{end} &= tanh(\mathbf{W}_3[\mathbf{h}_j^{end};\mathbf{z}_{i:j}^{end}]),\end{aligned} \tag{6}$$

where $\mathbf{h}_i^{start}$ and $\mathbf{h}_j^{end}$ are the i^{th} and j^{th} vector of $\mathbf{H}^{start}$ and $\mathbf{H}^{end}$. $\mathbf{p}_{i:j}$ is the computed attention vector for both context and roles which is calculated similarly to Eq. 3 or Eq. 4 and $\mathbf{W}_2, \mathbf{W}_3 \in \mathbb{R}^{2d\times d}$ are learnable parameters. Then we obtain the final representation $\widetilde{\mathbf{s}}_{i:j}$ for a candidate span

Dataset	Split	# Doc.	# Event	# Argument	# Event Types	# Role Types
RAMS	Train	3,194	7,329	17,026	139	65
	Dev	399	924	2,188	131	62
	Test	400	871	2,023	-	-
WikiEvents	Train	206	3,241	4,542	49	57
	Dev	20	345	428	35	32
	Test	20	365	566	34	44

Table 1: Detailed statistics of two datasets.

as follows: $\widetilde{\mathbf{s}}_{i:j} = \mathbf{W}^s[\mathbf{h}_{i:j}^{start}; \mathbf{s}_{i:j}; \mathbf{h}_{i:j}^{end}]$, where $\mathbf{W}^s \in \mathbb{R}^{3d\times d}$ is the learnable model parameter.

Finally, the boundary loss is defined to detect the start and end position following (Xu et al., 2022):

$$\begin{aligned}\mathcal{L}_b = -\textstyle\sum_{i=1}^{|\mathcal{D}|}[y_i^s \log P_i^s + (1-y_i^s)\log(1-P_i^s) \\ + y_i^e \log P_i^e + (1-y_i^e)\log(1-P_i^e)],\end{aligned} \tag{7}$$

where y_i^s and y_i^e denote golden labels and $P_i^s = \text{sigmoid}(\mathbf{W}_4\mathbf{h}_i^{start})$ and $P_i^e = \text{sigmoid}(\mathbf{W}_5\mathbf{h}_i^{end})$ are the probabilities of the word w_i predicted to be the first or last word of a golden argument span.

Classification Loss For a candidate span $s_{i:j}$ in event e, we concatenate the span representation $\widetilde{\mathbf{s}}_{i:j}$, trigger representation $\mathbf{h}_t$, their absolute difference $|\mathbf{h}_t - \widetilde{\mathbf{s}}_{i:j}|$, element-wise multiplication $\mathbf{h}_t \odot \widetilde{\mathbf{s}}_{i:j}$, event type embedding $\mathbf{H}^e$ and span length embedding $\mathbf{E}_{len}$ and get the prediction $P(r_{i:j})$ of the candidate span $s_{i:j}$ via a feed-forward network:

$$\mathbf{I}_{i:j} = [\widetilde{\mathbf{s}}_{i:j}; \mathbf{h}_t; |\mathbf{h}_t - \widetilde{\mathbf{s}}_{i:j}|; \mathbf{h}_t \odot \widetilde{\mathbf{s}}_{i:j}; \mathbf{H}^e; \mathbf{E}_{len}], \tag{8}$$

$$P(r_{i:j}) = \text{FFN}(\mathbf{I}_{i:j}). \tag{9}$$

Considering most candidate arguments are negative samples and the imbalanced role distribution, we adopt focal loss (Lin et al., 2017) to make the training process focus more on useful positive samples, where α and γ are hyperparameters.

$$\begin{aligned}\mathcal{L}_c = -\textstyle\sum_{i=1}^{|\mathcal{D}|}\sum_{j=1}^{|\mathcal{D}|} \alpha[1 - P(r_{i:j} = y_{i:j}))]^\gamma \\ \cdot \log P(r_{i:j} = y_{i:j}).\end{aligned} \tag{10}$$

Finally, we have the train loss consisting of $\mathcal{L}_c$ and $\mathcal{L}_b$ with hyperparameter λ:

$$\mathcal{L} = \mathcal{L}_c + \lambda\mathcal{L}_b. \tag{11}$$

3 Experiments

3.1 Experimental Setup

Datasets and Metrics We evaluate the proposed model on two large-scale public document-level EAE datasets, RAMSv1.0 (Ebner et al., 2020) and WikiEvents (Li et al., 2021) following the official train/dev/test split, whose detailed data statistic are shown in Table 1. Following (Xu et al., 2022), we report the Span F1 and Head F1 on dev and test sets for RAMS dataset. Span F1 requires the predicted argument spans to fully match the golden ones, while Head F1 evaluates solely on the head word[3] of the argument span. Additionally, for WikiEvents dataset, we report the Head F1 and Coref F1 scores on test set for argument identification task (Arg IF) and argument classification (Arg CF) task respectively following (Li et al., 2021). The Coref F1 evaluates the coreference between extracted arguments and golden arguments as used by (Ji and Grishman, 2008) and the model achieves Coref F1 if extracted arguments are coreferential with golden arguments.

Baselines We compare different categories of document-level EAE models which mainly consist of tagging-based methods such as **BERT-CRF** (Shi and Lin, 2019), **BERT-CRF$_{\text{TCD}}$** (Ebner et al., 2020), span-based methods like **Two-Step** (Zhang et al., 2020b), **Two-Step$_{\text{TCD}}$** (Ebner et al., 2020), **TSAR** (Xu et al., 2022), and other generation-based methods such as **FEAE** (Wei et al., 2021), **BERT-QA** (Du and Cardie, 2020b), **BART-Gen** (Li et al., 2021), **EA^{2}E** (Zeng et al., 2022). Moreover, we use BERT$_{\text{base}}$ (Devlin et al., 2019) and RoBERTa$_{\text{large}}$ (Liu et al., 2019) as the pre-trained transformer-based encoder.

Hyperparameters Setting We set the dropout rate to 0.1, batch size to 8, and train our SCPRG using Adam (Kingma and Ba, 2014) as optimizer with 3e-5 learning rate. The hidden dimension d is 768 for SCPRG$_{\text{base}}$ and 1024 for SCPRG$_{\text{large}}$. In order to mitigate imbalanced role distribution problem, we set the weight ratio α of empty class and other classes to 10:1. We set hyperparameters γ

[3] The head word of a span is defined as the word that has the smallest arc distance to the root in the dependency tree.

to 2 and boundary loss weight λ to 0.1 for both two datasets. We train SCPRG for 50 epochs for RAMS dataset and 100 epochs for WikiEvents dataset.

3.2 Main Results

Table 2 shows the experimental results on both dev and test set in RAMS dataset. Compared with previous tagging-based and span-based methods like BERT-CRF and Two-Step, our SCPRG equipped with BERT$_{\text{base}}$ yields an improvement of **+8.46/+9.64** $\sim$ **+6.36/+7.14** Span F1 and **+7.68/+9.00** $\sim$ **+5.38/+6.40** Head F1 on dev/test set, showing that our SCPRG framework has superiority in excluding impossible candidate spans and solving the imbalance of data distribution problem. Significantly, SCPRG with RoBERTa$_{\text{large}}$ also outperforms previous state-of-the-art models BART-Gen$_{\text{large}}$[4] (**+3.68/+2.34** Span/Head F1 on test set) and TSAR$_{\text{large}}$ (**+1.14/+1.13** Span/Head F1 on test set). These results demonstrate the superior extraction ability of our model, benefiting from the effect of contextual clue information and latent role representation with semantic relevance.

Moreover, we further validate our SCPRG on WikiEvents and achieve new state-of-the-art performance in both tasks with base and large pre-trained models, which can be viewed in Table 3. Our SCPRG outperforms previous competitive methods like TSAR and EA^{2}E. Compared with TSAR$_{\text{large}}$, our SCPRG improves up to **+0.64/+0.58** Head/Coref F1 for argument identification and **+1.22/+1.29** Head/Coref F1 for argument classification on the test set. Besides, SCPRG also outperforms recent competitive generation-based method EA^{2}E$_{\text{large}}$ in argument identification (**+2.64/+0.33** Head/Coref F1) and argument classification (**+2.31/+0.38** Head/Coref F1) tasks. These experimental improvements demonstrate the great advantage of our framework fused with argument-event specific context information and the helpful guidance of latent role information.

Method	Dev		Test	
	Span F1	Head F1	Span F1	Head F1
BERT-CRF	38.1	45.7	39.3	47.1
BERT-CRF$_{\text{TCD}}$	39.2	46.7	40.5	48.0
Two-Step	38.9	46.4	40.1	47.7
Two-Step$_{\text{TCD}}$	40.3	48.0	41.8	49.7
TSAR$_{\text{base}}$	45.23	51.70	48.06	55.04
FEAE	-	-	47.40	-
SCPRG$_{\text{base}}$ (Ours)	**46.56**	**53.38**	**48.94**	**56.10**
BART-Gen$_{\text{large}}$	-	-	48.64	57.32
TSAR$_{\text{large}}$	49.23	56.76	51.18	58.53
SCPRG$_{\text{large}}$ (Ours)	**50.53**	**57.66**	**52.32**	**59.66**

Table 2: Main results of RAMS.

Method	Arg IF		Arg CF	
	Head F1	Coref F1	Head F1	Coref F1
BERT-CRF	69.83	72.24	54.48	56.72
BERT-QA	61.05	64.59	56.16	59.36
BERT-QA-Doc	39.15	51.25	34.77	45.96
TSAR$_{\text{base}}$	75.52	73.17	68.11	66.31
SCPRG$_{\text{base}}$ (Ours)	**76.13**	**74.90**	**68.91**	**68.33**
TSAR$_{\text{large}}$	76.62	75.52	69.70	68.79
BART-Gen$_{\text{large}}$	71.75	72.29	64.57	65.11
EA^{2}E$_{\text{large}}$	74.62	75.77	68.61	69.70
SCPRG$_{\text{large}}$ (Ours)	**77.26**	**76.10**	**70.92**	**70.08**

Table 3: Main results of WikiEvents.

3.3 Ablation Study

To better illustrate the capabilities of our components, we conduct ablation study on RAMS dataset as shown in Table 4. We also provide ablation study results on WikiEvents datasets in Appendix A.

First, when we remove span-trigger-based contextual pooling (STCP) module, both Span F1 and Head F1 score of SCPRG$_{\text{base}}$/ SCPRG$_{\text{large}}$ drop by **1.61/1.43** and **1.42/2.09** on test set, which indicates that our STCP plays a vital role in capturing the clue information of non-argument context that is crucial for document-level EAE.

Additionally, when removing role-based latent information guidance (RLIG) module[5], the performance of SCPRG$_{\text{base}}$/ SCPRG$_{\text{large}}$ drops sharply by **1.03/1.04** Span F1 and **1.58/1.2** Head F1 on RAMS test set. It suggests that our RLIG module effectively guides argument extraction with meaningful latent role representations containing semantic relevance among roles. When removing both STCP and RLIG module, the performance decay exceeds that when removing a single module, which explains that our two modules can work together to improve the performance.

Moreover, when removing argument-impossible spans exclusion (ASE) operation, both SCPRG$_{\text{base}}$ and SCPRG$_{\text{large}}$ have a performance decay, which

[4] BART-Gen$_{\text{large}}$ is based on BART$_{\text{large}}$ (Lewis et al., 2019) which is pre-trained on the same corpus.

[5] We also remove the corresponding role tokens added in the input sequence.

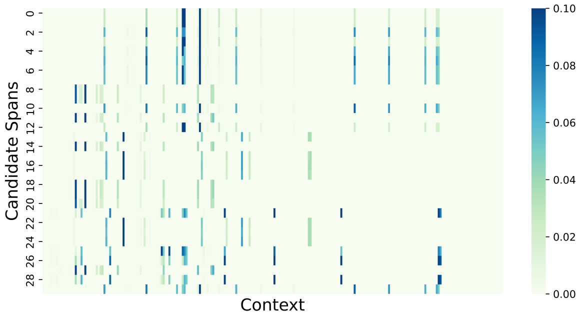


Figure 4: Visualization on attention weights to the context based on different candidate spans in an event.

indicates that excluding argument-impossible candidate spans eliminates noise information and contributes to argument extraction. Focal Loss helps to balance the representation of positive and negative samples, facilitating smooth convergence of the model during training. However, it does not contribute to improving the performance of the model.

3.4 Analysis of Context Attention Weights

To assess the effectiveness of STCP in capturing useful contextual information for candidate arguments, we visualize the contextual weights $\mathbf{p}^c_{i:j}$ in Eq. 3 of an example of Figure 1. As shown in Figure 5, our STCP gives high weights to non-argument words such as *attack*, *responsibility* and *terrorist attack*, which are most relevant to the span-trigger pair (*Islamic State, wounded*). Interestingly, our STCP also gives relatively high attention weights to words in other arguments like *explosive*, *Dozens* and *Kabul*, which means that these argument words provide important information for the role prediction of *Islamic State*. The visualization demonstrates that our STCP can not only capture the non-argument clue information that is related to candidate spans, but model the information interaction among related arguments in an event.

Additionally, we also explore the attention weights based on different span-trigger pairs in an event. In Figure 4, we randomly select 30 candidate spans in an event and draw the heat map based on their attention weights to the context. The heat map shows that different candidate arguments focus on different context information, indicating that our STCP can adaptively select contextual information according to candidate argument spans.

3.5 Analysis of Role Information Guidance

Method	Params	Dev		Test	
		Span F1	Head F1	Span F1	Head F1
SCPRG$_{\text{base}}$	119.37M	46.56	53.38	48.94	56.10
-STCP	118.78M	46.18	52.87	47.33	54.68
-RLIG	118.78M	45.27	52.36	47.91	54.52
-STCP&RLIG	118.19M	45.07	51.59	45.76	53.16
-ASE	119.37M	45.92	52.61	48.26	55.63
SCPRG$_{\text{large}}$	372.90M	50.53	57.66	52.32	59.66
-STCP	371.85M	49.94	56.55	50.89	57.57
-RLIG	371.68M	49.96	57.32	51.28	58.46
-STCP&RLIG	370.63M	48.33	54.04	47.52	55.61
-ASE	372.90M	49.80	56.31	51.73	58.48

Table 4: Ablation Study on RAMS for SCPRG.

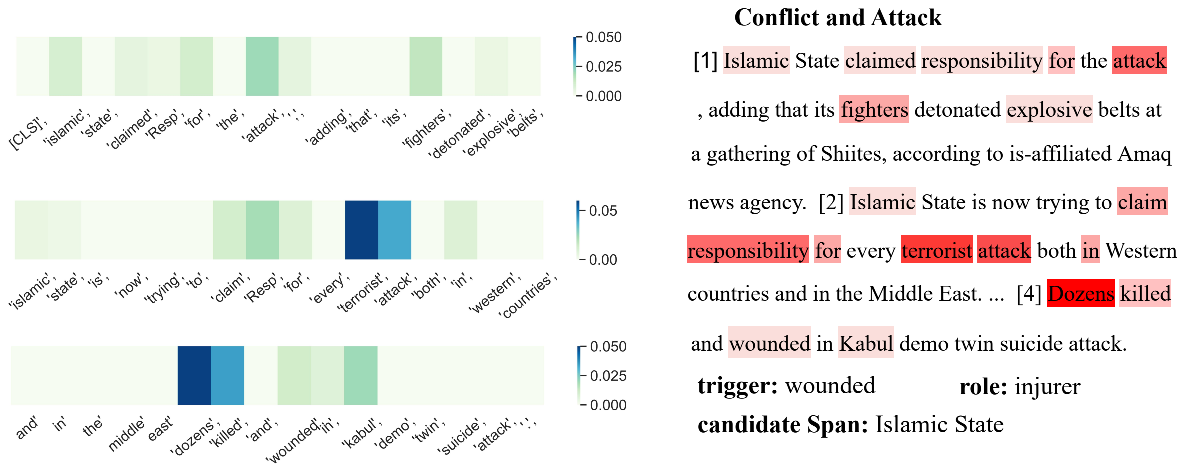
To verify that our model can capture semantic relevance among roles, we visualize the cosine similarity between latent role representations from two events in RAMS dataset in Fig 6. As the figure shows, roles *origin* and *destination*, *attacker* and *target* have similar representations, which agrees with their semantics, demonstrating that our model can capture the semantic relevance among roles.

Moreover, in order to verify the beneficial guidance of role representations, we display the t-SNE (van der Maaten and Hinton, 2008) visualization of arguments belonging to two different roles that co-occur in 5 different documents, along with corresponding latent role embeddings. As Figure 7a shows, arguments belonging to the same role in different documents are scattered over the whole embedding space due to their different target events and context. Notably, fused with latent role embeddings, in Figure 7b, the representation of arguments belonging to *victim* or *place* is more adjacent, which illustrates our RLIG provides beneficial latent role information guidance.

3.6 Analysis of Complexity and Compatibility

SCPRG is a simple but effective framework for document-level EAE, where both STCP and RLIG introduce few parameters. Specifically, STCP leverages the well-learned attention heads from the pre-trained encoder and makes multiplication and normalization operation, which only introduces about 0.28% new parameters as shown in Table 4. Our RLIG only introduces about 0.3% new parameters in the role embedding layer[6] and feature fusion layer. This makes the parameter quantity of our model approximate to the transformer-based encoder plus a MLP classifier.

[6] We add new special tokens for role types and therefore the RLIG module introduces more parameters in SCPRG$_{\text{large}}$.



Conflict and Attack

[1] Islamic State claimed responsibility for the attack , adding that its fighters detonated explosive belts at a gathering of Shiites, according to is-affiliated Amaq news agency. [2] Islamic State is now trying to claim responsibility for every terrorist attack both in Western countries and in the Middle East. ... [4] Dozens killed and wounded in Kabul demo twin suicide attack.

trigger: wounded **role:** injurer

candidate Span: Islamic State

Figure 5: Context weights of an example from RAMS. We visualize the weight of context tokens based on the span-trigger pair (*Islamic State, wounded*). We use different shades of color to represent attention weights.

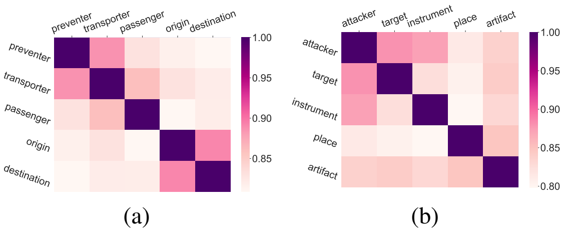
Additionally, the two proposed techniques STCP and RLIG have good transportability, which can be easily applied to other event extraction models, leveraging the attention heads of pre-trained transformer encoder such as BERT.

4 Related Works

Although deep learning has achieved significant success in many areas of computer vision (Li et al., 2022; Wang et al., 2023, 2022a; Pan et al., 2023; Wang and Chen, 2023) like 3D Scene Graph Generation (Liu et al., 2022c; Chen and Kou, 2023) and Image Semantic Segmentation (Zhang et al., 2022), its impact on Event Argument Extraction in natural language processing has been relatively limited. This is primarily due to the complexity and ambiguity inherent in natural language, which presents significant challenges for the accurate identification and extraction of event-related information.

4.1 Sentence-level Event Extraction

Previous approaches focus on extracting the event trigger and its arguments from a single sentence. (Chen et al., 2015) firstly propose a neural pipeline model for event extraction and (Nguyen et al., 2016; Nguyen and Grishman, 2015; Liu et al., 2017; Zhou et al., 2020) further extend the pipeline model to recurrent neural networks and convolutional neural networks. To model the dependency of words in a sentence, (Liu et al., 2018; Yan et al., 2019; Fernandez Astudillo et al., 2020) leverage dependency trees to model semantic and syntactic relations. (Wadden et al., 2019) enumerates all possible spans and construct span graphs with graph neural networks to propograte information. Some methods using transformer-based pre-trained model (Wadden et al., 2019; Wang et al., 2019; Tong et al., 2020; Lu et al., 2021; Liu et al., 2022b) also achieve remarkable performance.



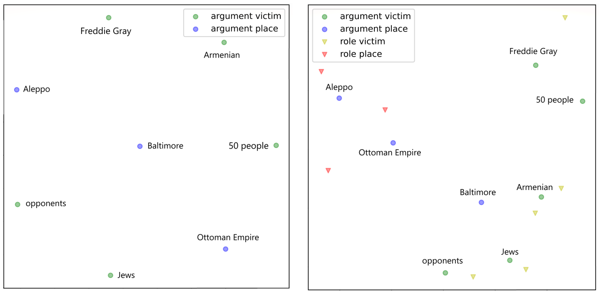
(a) (b)

Figure 6: The visualization of cosine similarity between role representations from two examples in RAMS dataset.

4.2 Document-level Event Extraction

In real-world scenarios, a large number of event elements are expressed across sentences and therefore recent works begin to explore document-level event extraction (DEE). DEE focuses on extracting event arguments from an entire document and faces the challenge of the long distance dependency (Wang et al., 2022b; Xu, 2022).

For document-level EAE, the key step of DEE, most of previous works mainly fall into three categories: (1) tagging-based methods; (2) span-based methods; (3) generation-based methods. (Wang et al., 2021; Du and Cardie, 2020a) utilize the sequence labeling model BiLSTM-CRF (Zhang et al., 2015) for DEE. (Zheng et al., 2019) propose a transformer-based architecture and model



(a) Without latent role guidance. (b) With latent role guidance.

Figure 7: A t-SNE visualization example from RAMS, where embeddings of arguments and roles are from 5 different documents. We use average pooling representations encoded by BERT for arguments in (a) and representations fused with latent role embeddings in (b).

DEE as a serial prediction paradigm, where arguments are predicted in a predefined role order. Base on their architecture, (Xu et al., 2021) construct a heterogeneous graph and a tracker module to capture the interdependency among events. However, tagging-based methods are inefficient due to the restriction to the extraction of individual arguments, and the former extraction will not consider the latter extraction results. (Yang et al., 2021) propose an encoder-decoder framework that extracts structured events in a parallel manner. Besides, (Ren et al., 2022) integrate argument roles into document encoding to aware tokens of multiple role information for nested arguments problem. Other span based methods (Ebner et al., 2020; Zhang et al., 2020b) predict the argument roles for candidate text spans with a maximum length limitation. Moreover, (Xu et al., 2022) propose a two-stream encoder with AMR-guided graph to solve long-distance dependency problem. On another aspect, (Li et al., 2021) formulate the problem as conditional generation and (Du et al., 2021) regards the problem as a sequence-to-sequence task. (Wei et al., 2021) reformulate the task as reading a comprehension task.

5 Conclusion

In this paper, we propose a novel SCPRG framework for document-level EAE that mainly consists of two compact, effective and transplantable modules. Specifically, our STCP adaptively aggregates the information of non-argument clue words and RLIG provides latent role information guidance containing semantic relevance among roles. Experimental results show that SCPRG outperforms existing state-of-the-art EAE models and further analyses demonstrate that our method is both effective and explainable. For future works, we hope to apply SCPRG to more information extraction tasks such as relation extraction and multilingual extraction, where contextual information plays a significant role.

6 Limitations

Although our experiments prove the superiority of our SCPRG model, it is only applicable to document-level EAE tasks with known event triggers because both STCP and RLIG calculate the attention product of the trigger and candidate spans. However, in real-life scenarios, event triggers are not always available. In view of this problem, we have a preliminary solution and plan to improve our model in the next work. The core idea of our method is to select and integrate context and role information based on candidate arguments and target events. Based on this idea, we briefly provide two solutions for the above limitation. First, we can make the model predict the best candidate trigger words. Second, we can replace trigger words with special event tokens. In the next work, we plan to extend our model to document-level EAE tasks without trigger words and evaluate it through extensive experiments.

Acknowledgements

This work was supported by the National Science Foundation of China under Grant 61976043, and in part by the Science and technology support program of Sichuan Province under Grant 2022YFG0313.

References

Jialu Chen and Gang Kou. 2023. Attribute and structure preserving graph contrastive learning. In *Proc. of AAAI*.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proc. of ACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of AACL*.

Xinya Du and Claire Cardie. 2020a. Document-level event role filler extraction using multi-granularity contextualized encoding. In *Proc. of ACL*.

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

Xinya Du, Alexander Rush, and Claire Cardie. 2021. GRIT: Generative role-filler transformers for document-level event entity extraction. In *Proc. of EACL*.

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

Ramón Fernandez Astudillo, Miguel Ballesteros, Tahira Naseem, Austin Blodgett, and Radu Florian. 2020. Transition-based parsing with stack-transformers. In *Proc. of EMNLP Findings*.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: HLT*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*.

Haoquan Li, Laoming Zhang, Daoan Zhang, Lang Fu, Peng Yang, and Jianguo Zhang. 2022. Transvlad: Focusing on locally aggregated descriptors for few-shot learning. In *Proc. of ECCV*.

Manling Li, Alireza Zareian, Ying Lin, Xiaoman Pan, Spencer Whitehead, Brian Chen, Bo Wu, Heng Ji, Shih-Fu Chang, Clare Voss, Daniel Napierski, and Marjorie Freedman. 2020. GAIA: A fine-grained multimedia knowledge extraction system. In *Proc. of ACL*.

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

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proc. of ICCV*.

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, Li Zhou, Dingyi Zeng, and Hong Qu. 2022a. Document-level relation extraction with structure enhanced transformer encoder. In *Proc. of IJCNN*.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022b. Dynamic prefix-tuning for generative template-based event extraction. In *Proc. of ACL*.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proc. of EMNLP*.

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. *CoRR*.

Yuanyuan Liu, Chengjiang Long, Zhaoxuan Zhang, Bokai Liu, Qiang Zhang, Baocai Yin, and Xin Yang. 2022c. Explore contextual information for 3d scene graph generation. *IEEE Transactions on Visualization and Computer Graphics*.

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 *Proc. of ACL*.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proc. of AACL*.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *Proc. of ACL*.

Zhenyu Pan, Anshujit Sharma, Jerry Yao-Chieh Hu, Zhuo Liu, Ang Li, Han Liu, Michael Huang, and Tony Tong Geng. 2023. Ising-traffic: Using ising machine learning to predict traffic congestion under uncertainty. In *Proc. of AAAI*.

Amir Pouran Ben Veyseh, Minh Van Nguyen, Franck Dernoncourt, Bonan Min, and Thien Nguyen. 2022. Document-level event argument extraction via optimal transport. In *Proc. of ACL Findings*.

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 *Proc. of COLING*.

Peng Shi and Jimmy Lin. 2019. Simple BERT models for relation extraction and semantic role labeling. *CoRR*.

Meihan Tong, Bin Xu, Shuai Wang, Yixin Cao, Lei Hou, Juanzi Li, and Jun Xie. 2020. Improving event detection via open-domain trigger knowledge. In *Proc. of ACL*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proc. of EMNLP*.

Likang Wang and Lei Chen. 2023. FTSO: Effective NAS via First Topology Second Operator. *Preprints*.

Likang Wang, Yue Gong, Xinjun Ma, Qirui Wang, Kaixuan Zhou, and Lei Chen. 2022a. Is-mvsnet: Importance sampling-based mvsnet. In *Proc. of ECCV*.

Likang Wang, Yue Gong, Qirui Wang, Kaixuan Zhou, and Lei Chen. 2023. Flora: dual-frequency loss-compensated real-time monocular 3d video reconstruction. In *Proc. of AAAI*.

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

Xiaozhi Wang, Xu Han, Zhiyuan Liu, Maosong Sun, and Peng Li. 2019. Adversarial training for weakly supervised event detection. In *Proc. of AACL*.

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juanzi Li, and Jie Zhou. 2021. CLEVE: Contrastive Pre-training for Event Extraction. In *Proc. of ACL*.

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*.

Jinghua Xu. 2022. Xu at SemEval-2022 task 4: Pre-BERT neural network methods vs post-BERT RoBERTa approach for patronizing and condescending language detection. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 479–484, Seattle, United States. Association for Computational Linguistics.

Runxin Xu, Tianyu Liu, Lei Li, and Baobao Chang. 2021. Document-level event extraction via heterogeneous graph-based interaction model with a tracker. In *Proc. of ACL*.

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*.

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. In *Proc. of EMNLP*.

Hang Yang, Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Taifeng Wang. 2021. Document-level event extraction via parallel prediction networks. In *Proc. of ACL*.

Dingyi Zeng, Wenyu Chen, Wanlong Liu, Li Zhou, and Hong Qu. 2023a. Rethinking random walk in graph representation learning. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Dingyi Zeng, Wanlong Liu, Wenyu Chen, Li Zhou, Malu Zhang, and Hong Qu. 2023b. Substructure aware graph neural networks. In *Proc. of AAAI*.

Qi Zeng, Qiusi Zhan, and Heng Ji. 2022. EA^{2}E: Improving consistency with event awareness for document-level argument extraction. In *Proc. of ACL Findings*.

Daoan Zhang, Chenming Li, Haoquan Li, Wenjian Huang, Lingyun Huang, and Jianguo Zhang. 2022. Rethinking alignment and uniformity in unsupervised image semantic segmentation. *arXiv preprint arXiv:2211.12875*.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional long short-term memory networks for relation classification. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*.

Tianran Zhang, Muhao Chen, and Alex A. T. Bui. 2020a. Diagnostic prediction with sequence-of-sets representation learning for clinical events. *medRxiv*.

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

Shun Zheng, Wei Cao, Wei Xu, and Jiang Bian. 2019. Doc2EDAG: An end-to-end document-level framework for Chinese financial event extraction. In *Proc. of EMNLP*.

Li Zhou, Tingyu Wang, Hong Qu, Li Huang, and Yuguo Liu. 2020. A weighted gcn with logical adjacency matrix for relation extraction. In *ECAI 2020*.

Wenxuan Zhou, Kevin Huang, Tengyu Ma, and Jing Huang. 2021. Document-level relation extraction with adaptive thresholding and localized context pooling. In *Proc. of AAAI*.

Method	Arg IF		Arg CF	
	Head F1	Coref F1	Head F1	Coref F1
SCPRG$_{\text{base}}$	76.13	74.90	68.91	68.33
-STCP	74.64	73.46	67.48	67.07
-RLIG	75.59	73.83	68.26	67.49
-STCP&RLIG	73.92	73.22	66.90	65.98
-ASE	75.86	74.37	68.41	68.01
SCPRG$_{\text{large}}$	77.26	76.10	70.92	70.08
-STCP	75.54	73.37	69.67	68.77
-RLIG	76.30	74.08	69.87	68.96
-STCP&RLIG	75.45	73.55	68.63	67.30
-ASE	76.57	74.22	69.65	68.01

Table 5: Ablation Study on WikiEvent for SCPRG.

A Ablation Study

In the main body of the paper, we conduct ablation study on RAMS dataset for SCPRG$_{\text{base}}$ and SCPRG$_{\text{large}}$. In order to fully evaluate the effect of different components on our model, we also provide the results of the ablation study on WikiEvents for for SCPRG$_{\text{base}}$ and SCPRG$_{\text{large}}$.

As shown in Table 5, when we remove STCP module, both Head F1 and Coref F1 score of SCPRG$_{\text{base}}$ drop by **1.49/1.44** and **1.43/1.26** on test set for argument identification task (Arg IF) and argument classification (Arg CF) task, which demonstrates that our STCP captures the clue information of non-argument context that is significant for document-level EAE.

Additionally, when removing RLIG module, the performance of SCPRG$_{\text{large}}$ drops sharply by **0.96/2.02** Head F1 and **1.05/1.12** Coref F1 on Wikievent test set for both two tasks. Moreover, when we remove argument-impossible spans exclusion (ASE), both SCPRG$_{\text{base}}$ and SCPRG$_{\text{large}}$ have a performance decay. These results indicate that both STCP and ASE are beneficial.

B Co-occurrence Frequency Matrix

In this section, we show the complete co-occurrence frequency matrix which contains all roles in RAMS test set. We count the frequency of co-occurrence between every two roles and draw the heat map according to the frequency in Figure 8. It can be seen from the figure that the co-occurrence phenomenon exists between many roles, especially those occur in the same event, which indicates that there is semantic relevance among roles.

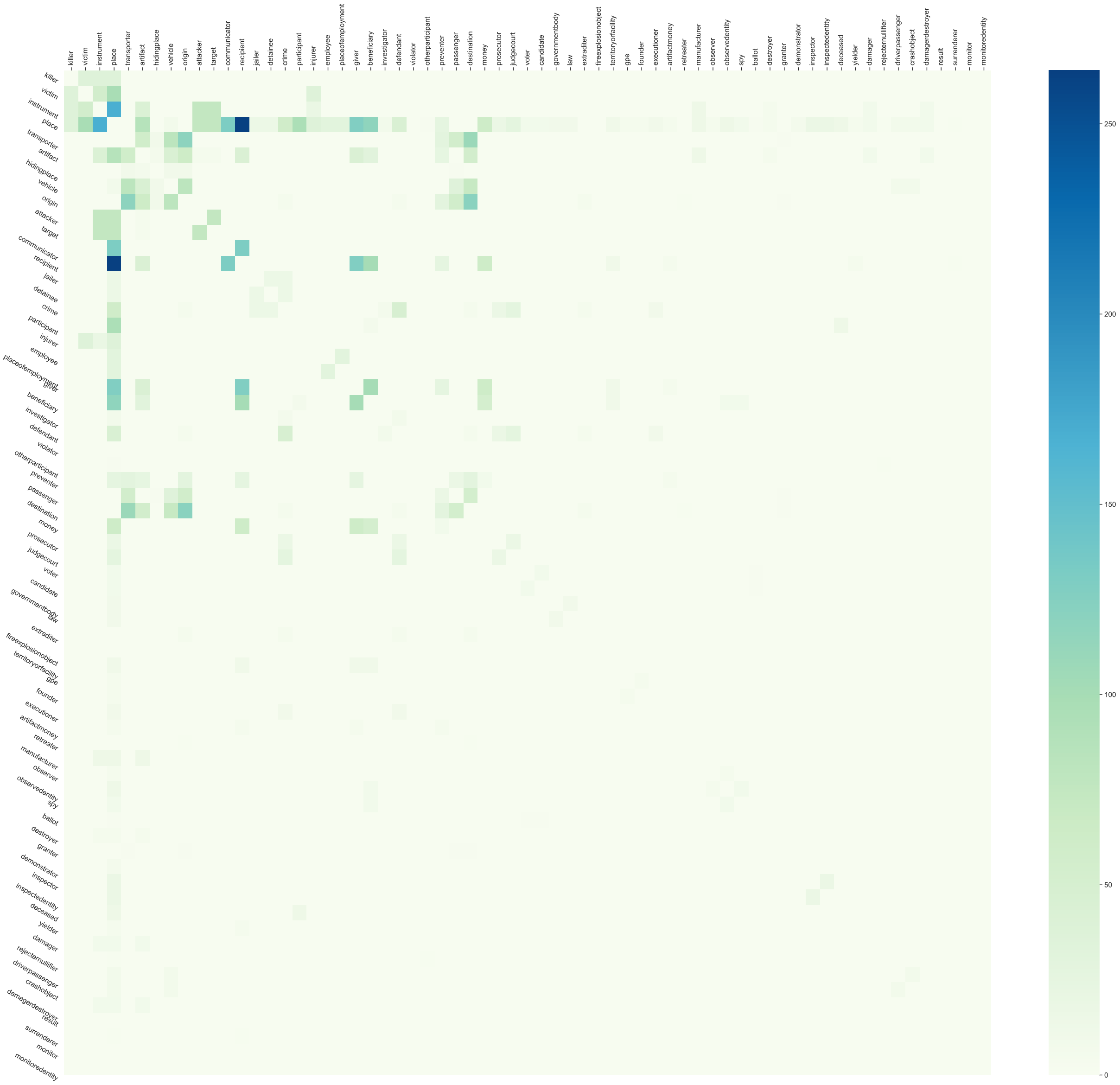


Figure 8: Visualization of the co-occurrence frequency between all roles in RAMS test set. we have reserved and set the co-occurrence number with itself to zero.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
6 Limitations

☒ A2. Did you discuss any potential risks of your work?
Our work does not involve potential risks

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1 Introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

□ B1. Did you cite the creators of artifacts you used?
No response.

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

□ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

3.6 Analysis of Complexity and Compatibility

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
3.6 Analysis of Complexity and Compatibility

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
3.1 Experimental Setup

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
3.2 Main Results

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
3.1 Experimental Setup

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.