ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10446551

SORTING, REASONING, AND EXTRACTION: AN EASY-TO-HARD REASONING FRAMEWORK FOR DOCUMENT-LEVEL EVENT ARGUMENT EXTRACTION

Hao Li[1,2], *Yanan Cao*[1,2*], *Yubing Ren*[1,2], *Fang Fang*[1,2], *Lanxue Zhang*[1,2], *Yingjie Li*[1,2], *Shi Wang*[3]

[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

ABSTRACT

Document-level event argument extraction is a crucial task to help understand event information. Existing methods mostly ignore the different extraction difficulties of arguments, and the lack of task planning significantly affects the extraction and reasoning abilities of the model. In this paper, we innovatively analyze the difficulty of arguments and propose a novel framework for reasoning from easy to hard, aiming to use the information of simple arguments to help the extraction of difficult arguments in a human-like way. Specifically, our framework consists of three core modules: sorting, reasoning, and extraction. The sorting module first sorts the argument roles according to the current context and plans the reasoning path from easy to hard. Then, the reasoning module performs information reasoning based on the reasoning path to help capture the information of difficult arguments. Finally, the extraction module utilizes the reasoning information to complete argument extraction. Experimental results on the RAMS and WikiEvents datasets show the great advantages of our proposed approach. In particular, we obtain new state-of-the-art (SOTA) performance in multiple scenarios.

***Index Terms*—** Document-level Event Argument Extraction, Information Extraction, Natural Language Processing

1. INTRODUCTION

Extracting structured event knowledge from large amounts of unstructured text is a critical, yet unsolved goal of natural language processing, especially when dealing with document-level text. Document-level Event Argument Extraction (Document-level EAE) aims to identify the arguments of a given event from a document and recognize the specific roles they play, which benefits many downstream applications, such as information retrieval, question answering, and event graph reasoning. As illustrated by the example in Fig. 1, given a trigger word *robbed* for a stealrobhijack event, an event argument extractor is expected to identify *two male teenagers*, *a knife*, *One coach*, *his wallet* as the event arguments and predict their roles as Attacker, Instrument, Target, and Artifact.

Typical efforts in document-level EAE can be roughly divided into classification-based models [1, 2, 3, 4, 5, 6], question answering (QA)-based models [7, 8], generation-based models [9, 10], and prompt-based models [11, 12]. These methods usually define a unified paradigm to solve the document-level EAE task. When faced with complex scenarios, only a few methods additionally propose dedicated architectures to help the model alleviate challenges, such as scattered arguments and multi-word arguments. However, how to exploit the different extraction difficulties of arguments is often ignored, and the lack of task planning significantly affects the extraction and reasoning abilities of the model.

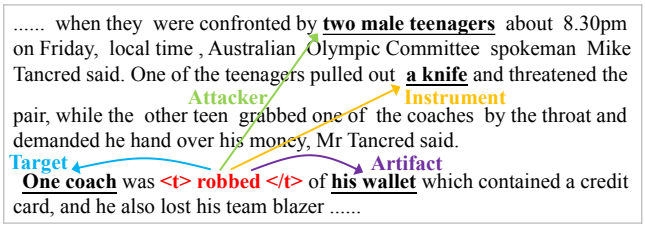

Fig. 1. An example of event argument extraction from RAMS dataset, its arguments reside in multiple sentences. Trigger words are included in special tokens ⟨**t**⟩ and ⟨**/t**⟩. Underlined words denote arguments and arcs denote roles.

In reality, humans often complete tasks in a multi-step manner from easy to hard. Before tackling a complex task, humans typically plan the task according to difficulty. They first tackle the easy parts to learn the knowledge related to the task, and then use this knowledge to tackle the difficult parts step by step until the whole task is completed. Taking the document-level EAE task as an example, its goal is to extract all arguments of the event in the document. To solve it, humans usually locate events in documents based on the trigger word, first extracting simple arguments, and gradually extracting complex arguments by leveraging existing results until all arguments are obtained. In Fig. 1, we observe that *One coach* and *his wallet* are simple arguments because they contain only two words and are located in the same sentence as the trigger word. However, the extraction of *two male teenagers* is more difficult, because its span length is longer and far away from the trigger word, and it needs to combine more complex context reasoning information. To accurately extract all arguments in the event, we should first extract *One coach* and *his wallet*, then *a knife* and *two male teenagers*. This strategy of tackling tasks from easy to hard helps humans gradually develop the ability to tackle complex tasks.

Built on these motivations, we raise our research question: *How do we extract arguments from easy to hard in a human-like manner?* In this paper, we innovatively analyze the difficulty of arguments and propose a novel framework for reasoning from easy to hard, aiming to use the information of simple arguments to help the extraction of difficult arguments in a human-like way. Specifically, our framework consists of three core modules: **S**orting, **R**easoning, and **E**xtr**A**ction (called **SREA**). The sorting module first sorts the argument roles according to the current context and plans the reasoning path from easy

*Corresponding author.

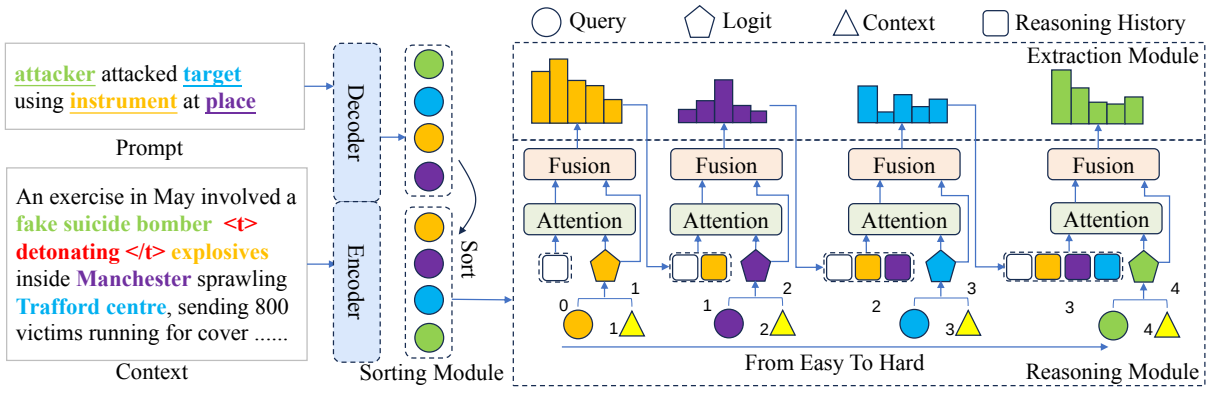

Fig. 2. The overall architecture of our SREA, which includes encoding and three core modules: sorting, reasoning, and extraction. We use a specific example to show the whole process, with text descriptions in each module. Colors represent different argument roles, and solid lines with arrows indicate the flow of data. The context here is shortened due to space limitations.

to hard. Then, the reasoning module performs information reasoning based on the reasoning path to help capture the information of difficult arguments. Finally, the extraction module utilizes the reasoning information to complete argument extraction.

We summarize our contributions as follows: (1) We innovatively analyze the difficulty of arguments and propose a novel framework, which extracts arguments from easy to hard in a human-like manner. **(2)** We introduce a new argument sorting and reasoning module that extracts simple arguments first while using the information from already obtained simple arguments to help reason about difficult arguments. **(3)** We conduct extensive experiments and detailed analysis on two widely used document-level EAE datasets, and the results show that our model achieves new SOTA.

2. METHOD

In this section, we first formulate the document-level EAE task, then introduce the overall framework of our model, as illustrated in Fig. 2, which contains encoding and three core modules: sorting, reasoning, and extraction module.

Task Definition. We first describe the task formalization of document-level EAE. Formally, given a context $\mathbf{x} = \{x_1, ..., x_n\}$ of n words, the event trigger word $e^{tri} \in \mathbf{x}$, the event type $et \in \mathcal{T}$, where $\mathcal{T} = \{et_1, ..., et_u\}$ is the set of all u different event types and the set of event-specific role types $\mathcal{R}^{et}$. The EAE task aims to extract all the arguments related to e^{tri} in $\mathbf{x}$ and assign a $role \in \mathcal{R}^{et}$ to each extracted argument a_i, where a_i is a text span in $\mathbf{x}$.

2.1. Encoding

The encoding module encodes context and argument roles. Benefiting from the development of pre-trained language models (PLM) [13, 14], we define the task paradigm as prompt tuning. In particular, to achieve a fair comparison, we utilize the same prompt for each event type as in previous work [12] that connects all argument roles with natural language. As shown in Fig. 2, for the event type *bombing*, the prompt Pt to indicate argument roles is *attacker attacked target using instrument at place*, where each underlined part represents an argument role slot[1].

Specifically, given the current context $\mathbf{x}$ and trigger word e^{tri}, we first use special tokens $\langle \mathbf{t} \rangle$ and $\langle \mathbf{/t} \rangle$ to markup the position of the trigger word, inserting them into context $\mathbf{x}$ before and after the trigger word e^{tri} to create an input context $\tilde{\mathbf{x}}$. Next, the BART encoder is employed to encode the input context $\tilde{\mathbf{x}}$ to obtain the sequence representation $\mathbf{H}_{\tilde{\mathbf{x}}}^{enc}$. Finally, the BART decoder is applied to learn richer representations for the context and prompt through the mutual interaction of cross-attention layers in the decoder module, returning representations $\mathbf{H}_{\mathbf{x}} = \text{Decoder}(\mathbf{H}_{\tilde{\mathbf{x}}}^{enc}, \mathbf{H}_{\tilde{\mathbf{x}}}^{enc})$ and $\mathbf{H}_{pt} = \text{Decoder}(Pt, \mathbf{H}_{\tilde{\mathbf{x}}}^{enc})$ for the context and prompt.

2.2. Sorting

The goal of the sorting module is to sort argument roles in the context using their difficulty and plan a reasoning path from easy to hard. By analyzing the performance of document-level EAE, we carefully select different principles and finally use the two principles to determine the difficulty score of each argument role, in order of priority: **(1) Argument Span Length** α: Multi-word arguments contain more information and are more difficult to extract. **(2) Trigger-Argument Distance** β: Arguments farther away from the trigger word require more complex contextual information for model reasoning.

According to the above principles, we construct the gold argument difficulty score $\mathcal{S}^{gold} = [s_1^{gold}, ..., s_L^{gold}]$, whose value indicates the relative difficulty, where $s_k^{gold} = \gamma * \alpha_k + \beta_k$ represents the gold difficulty score of the k-th argument role slot[2] and L represents the number of argument role slots in prompt Pt.

Guided by $\mathcal{S}^{gold}$, we aim to learn a reasoning path from easy to hard. Specifically, we first score each argument role in the context:

$$s_k^{pred} = \text{Sigmoid}(\text{FFN}([\mathbf{q}_k^{start}; \mathbf{q}_k^{end}])), \tag{1}$$

where $\mathbf{q}_k^{start}$ and $\mathbf{q}_k^{end}$ respectively represent the start and end queries of the argument role slot $\mathbf{h}_{pt_k} \in \mathbf{H}_{pt}$, FFN represents the feed-forward network, and $[;]$ means the concatenation operation. In order to facilitate understanding and description, we use $t \in \{start, end\}$ to represent in the following content.

$$\mathbf{q}_k^t = \mathbf{h}_{pt_k} \odot \mathbf{W}_q^t \in \mathbb{R}^h, \tag{2}$$

where $\mathbf{W}_q^t$ denotes the learnable parameters, $\odot$ denotes element-wise multiplication and h is the dimension of the embedding vector.

[1]Note that a role may have multiple slots, connected using *and*, and the maximum number is determined according to the training set.

[2]γ represents the priority weight, we set 1e3 (the value greater than β).

Based on the score s_k^{pred} of each argument role in the context, we sort in ascending order to get the predicted reasoning path ρ^{pred}, which represents the reasoning process from easy to hard.

During the model training phase, we adopt ListMLE [15] to calculate the sorting loss. Specifically, we first obtain the prediction scores for all argument roles $\mathcal{S}^{pred} = [s_1^{pred}, ..., s_L^{pred}]$, then sort $\mathcal{S}^{pred}$ according to the order of $\mathcal{S}^{gold}$ to get $\hat{\mathcal{S}}^{pred}$, and finally calculate the sorting loss:

$$\begin{aligned} \mathcal{L}_s &= \sum_{\mathbf{x}\in\mathcal{D}} -\log(P(\hat{\mathcal{S}}^{pred}|\mathbf{x})), \\ P(\hat{\mathcal{S}}^{pred}|\mathbf{x}) &= \prod_{k=1}^{L} \frac{\exp(\hat{s}_k^{pred})}{\sum_{i=k}^{L}\exp(\hat{s}_i^{pred})}, \end{aligned} \tag{3}$$

where $\mathcal{D}$ ranges over all context in dataset.

2.3. Reasoning

In the order of the predicted reasoning path ρ^{pred}, the reasoning module performs information reasoning from easy to hard. We first fuse query and context features to compute the start and end logits for each role:

$$\mathbf{c}_k^t = \mathbf{q}_k^t \mathbf{H}_\mathbf{x} \in \mathbb{R}^n. \tag{4}$$

Following the easy-to-hard reasoning path, we use information from simple arguments reasoned in previous stages to help the model reason about difficult arguments. Specifically, we apply the scaled dot-product attention mechanism [16] to compute the relevance score $\mathbf{z}_{k-1}^t$ of arguments in the previous stages of ρ^{pred}, and then obtain the reasoning information $\mathbf{m}_{k-1}^t$ of the previous simple arguments:

$$\begin{aligned} \mathbf{z}_{k-1}^t &= \mathrm{Softmax}(\frac{\mathbf{c}_k^t(\mathbf{r}_{k-1}^t)^\mathbf{T}}{\sqrt{d}}), \\ \mathbf{m}_{k-1}^t &= \mathbf{z}_{k-1}^t \mathbf{r}_{k-1}^t, \end{aligned} \tag{5}$$

where $\mathbf{r}_{k-1}^t = [\mathbf{g}_1^t, ..., \mathbf{g}_{k-1}^t]$ represents the reasoning probability distribution of arguments in the previous stage.

Finally, we apply the gating mechanism to incorporate reasoning information from previous stages to obtain the reasoning probability distribution for the current argument role:

$$\begin{aligned} \mathbf{f}_o^t &= \mathrm{Sigmoid}(\mathbf{W}_o^t[\mathbf{c}_k^t; \mathbf{m}_{k-1}^t]), \\ \mathbf{g}_k^t &= \mathbf{f}_o^t \odot \mathbf{c}_k^t + (1 - \mathbf{f}_o^t) \odot \mathbf{m}_{k-1}^t, \end{aligned} \tag{6}$$

where $\mathbf{f}_o^t$ is a gate that controls information redundancy.

2.4. Extraction

Based on the reasoning information from the previous steps, the extraction module extracts all the arguments in the current context. We first calculate the probabilities of the start and end positions of each argument:

$$\mathbf{p}_k^t = \mathrm{Softmax}(\mathbf{g}_k^t) \in \mathbb{R}^n. \tag{7}$$

and define the argument extraction loss function as:

$$\begin{aligned} \mathcal{L}_k(\mathbf{x}) &= -(\log \mathbf{p}_k^{start}(\phi_k^{start}) + \log \mathbf{p}_k^{end}(\phi_k^{end})), \\ \mathcal{L}_e &= \sum_{\mathbf{x}\in\mathcal{D}} \sum_{k=1}^{L} \mathcal{L}_k(\mathbf{x}), \end{aligned} \tag{8}$$

where ϕ_k^{start} and ϕ_k^{end} represent the start and end positions of the ground truth argument span. For an argument role slot with no corresponding argument, $(\phi_k^{start}, \phi_k^{end})$ is an empty span $(0, 0)$.

Model	PLM	RAMS		WikiEvents		
		Arg-I	Arg-C	Arg-I	Arg-C	Head-C
FEAE	BERT-b	53.5	47.4	-	-	-
DocMRC	BERT-b	-	45.7	-	43.3	-
EEQA	BERT-b	46.4	44.0	54.3	53.2	56.9
	BERT-l	48.7	46.7	56.9	54.5	59.3
BART-Gen	BART-b	50.9	44.9	47.5	41.7	44.2
	BART-l	51.2	47.1	66.8	62.4	65.4
EEQA-BART	BART-b	49.4	46.3	60.3	57.1	61.4
	BART-l	51.7	48.7	61.6	57.4	61.3
PAIE	BART-b	54.7	49.5	68.9	63.4	66.5
	BART-l	56.8	52.2	70.5	65.3	68.4
SREA	BART-b	**55.6**	**50.4**	**70.8**	**65.1**	**67.5**
	BART-l	**57.7**	**53.9**	**71.8**	**66.5**	**68.5**

Table 1. Overall performance in two mainstream datasets. **b** and **l** in the **PLM** column represent the base model and large model.

The overall loss function is divided into two parts: argument extraction loss $\mathcal{L}_e$, sorting loss $\mathcal{L}_s$, and λ is a hyper-parameter.

$$\mathcal{L} = \mathcal{L}_e + \lambda \mathcal{L}_s. \tag{9}$$

During the inference phase, we consider all candidate spans of the event argument as $\mathcal{C} = \{(i,j)|(i,j) \in n^2, 0 < j - i \leq \delta\} \cup \{(0,0)\}$, ensuring that the length of all spans does not exceed the threshold δ and $(0,0)$ means there is no argument. The prediction score for each span is obtained according to the probability $\mathbf{g}_k^{start}(i) + \mathbf{g}_k^{end}(j)$. The span with the highest score will be chosen as the argument for the prediction.

3. EXPERIMENTS

3.1. Experiment Setup

Dataset. We conduct experiments on two commonly used document-level EAE datasets: RAMS [17] and WikiEvents [9]. RAMS focuses on a document-level EAE task, including 139 event types and 65 argument roles, with more than 9k events. WikiEvents is another widely used document-level EAE dataset, which contains 50 event types and 59 argument roles, containing more than 3.9k events. We follow the official data split of each dataset.

Evaluation Metrics. We adopt three evaluation metrics. **(1)** Argument Identification (Arg-I): an event argument is correctly identified if its offsets and event type match those of any of the argument mentions. **(2)** Argument Classification (Arg-C): an event argument is correctly classified if its role type is also correct. **(3)** For WikiEvents dataset, we follow previous work [9], additionally evaluate Argument Head Classification (Head-C), which only concerns the matching of the headword of an argument. The F-measure (F1) score is used to evaluate the performance of the model.

Baselines. The following representative and competitive methods are chosen as baselines: **FEAE** [7] is a QA-based method that uses event frame-level knowledge to reason and capture the long-range dependency. **DocMRC** [8] is another QA-based method, assisted by two data augmentation regimes: implicit knowledge transfer and explicit data augmentation. **EEQA** [18] treats sentence-level EAE as a QA task, and obtains the start and end offsets of the argument spans through question answering. **BART-Gen** [9] defines EAE as a sequence-to-sequence task and generates corresponding arguments in a predefined format. **PAIE** [12] defines EAE as a new prompt tuning paradigm. It extracts argument spans by constructing role templates and is the current SOTA model.

Model	Arg-I	Arg-C	Head-C
SREA	**70.8**	**65.1**	**67.5**
SREA (use gold path)	**71.4**	**65.5**	**69.1**
trigger-distance priority	69.8	64.3	67.1
w/o sorting module	68.3	62.2	66.3
w/o reasoning module	68.5	62.5	66.7

Table 2. Ablation study on WikiEvents.

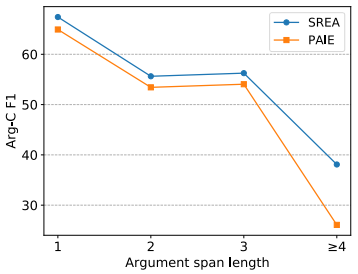

Fig. 3. Results of different argument span lengths on WikiEvents.

3.2. Main Results

Table 1 compares our method with all baselines on the RAMS and WikiEvents datasets. The most important observation in the table is that our proposed method significantly outperforms all baseline methods for both the base and large versions of the PLM model. We also perform a t-test with p-value < 0.05 on these improved metrics. Specifically, our method achieves an average Arg-C F1 improvement of 2.6% and 2.3% over the SOTA on the RAMS and WikiEvents datasets. Compared with the Head-C metrics (only headword is considered), our method achieves greater performance improvement on the stricter Arg-C metrics on the WikiEvents dataset. This clearly demonstrates that our sorting-reasoning-extraction framework is efficient in document-level EAE, and the reasoning path from easy to hard improves the argument extraction ability of the model.

3.3. Analysis And Discussion

Ablation Study. We conduct ablation studies to investigate the effectiveness of each component of our approach, and the results are shown in Table 2. We can find that each module can help boost the performance, removing any module or changing the sorting priority will lead to different degrees of performance degradation. To be specific, **(1)** Removing the sorting and reasoning modules leads to the worst performance (the Arg-C F1 drop by 2.9 and 2.6), illustrating that the sorting and reasoning processes are essential in our overall framework. **(2)** Changing the sorting priority achieves suboptimal performance, illustrating that the reasoning path from easy to hard is helpful for the document-level EAE.

Analysis of Reasoning Path. We explore how different reasoning paths affect performance in Table 2. Compared to the default argument order (w/o sorting module), our method and the trigger-distance priority method perform reasoning based on the difficulty order of arguments, thus obtaining 2.9 and 2.1 Arg-C F1 gains. It is worth noting that our method can achieve better performance when the gold reasoning path is known, which shows that the gold reasoning path is reasonable and beneficial for the task.

Analysis of Argument Length. We explore the performance of the model under different argument span lengths. As shown in Fig. 3, compared with single-word arguments, multi-word arguments increase the difficulty of extraction and lead to poor performance (dropping about 30 Arg-C F1). However, our method significantly alleviates the performance drop, especially in the most difficult scenario where the span length is not less than four, achieving 12.0 Arg-C F1 gains. This reveals that our method can help the model reason about the information of complex arguments in an easy-to-hard way, and improve the ability to extract complex arguments.

Model	Trigger-Argument Word Distance			
	$[0, 10)_{66\%}$	$[10, 20)_{16\%}$	$[20, 50)_{14\%}$	$[50, +\infty)_{4\%}$
PAIE	58.2	35.3	26.9	25.9
SREA	**59.2**	**36.5**	**29.3**	**27.3**

Table 3. Results of different trigger-argument distances on RAMS.

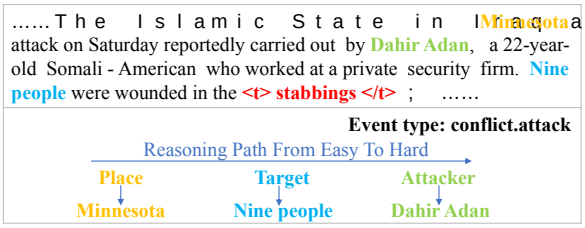

Fig. 4. Case Study.

Analysis of Trigger-Argument Distance. In the document-level EAE task, some event arguments are far away from the trigger word, which significantly increases the difficulty of extraction. Table 3 explores the performance in this scenario. It can be found that our method significantly improves the ability to extract arguments in all distance scenarios (1.0, 1.2, 2.4, and 1.4 Arg-C F1 gains), especially for long distances. This demonstrates that our method can use existing information to help reason about long-distance arguments, which is convenient for application in complex real-world scenarios.

Case Study. We show an example from RAMS dataset to illustrate the capability of our method. The example is presented in Fig. 4, the arguments of the event are scattered in different sentences and have different span lengths. Our method first accurately plans the reasoning path from easy to hard, then leverages the reasoning history information of simple arguments to help complete the extraction of complex arguments according to the reasoning path, and accurately extract all arguments in the event.

4. CONCLUSION

In this paper, we propose a novel easy-to-hard reasoning framework for the document-level EAE task, including three core modules: sorting, reasoning, and extraction. Extensive experiments on two datasets demonstrate the effectiveness and generalization ability of our proposed method in different scenarios. In the future, we will further apply our framework to other information extraction tasks.

5. ACKNOWLEDGEMENT

This work is supported by the National Key Research and Development Program of China (NO.2022YFB3102200) and Strategic Priority Research Program of the Chinese Academy of Sciences with No. XDC02030400.

6. REFERENCES

[1] Zhisong Zhang, Xiang Kong, Zhengzhong Liu, Xuezhe Ma, and Eduard Hovy, "A two-step approach for implicit event argument detection," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020, pp. 7479–7485, Association for Computational Linguistics.

[2] Yusheng Huang and Weijia Jia, "Exploring sentence community for document-level event extraction," in *Findings of the Association for Computational Linguistics: EMNLP 2021*, Punta Cana, Dominican Republic, Nov. 2021, pp. 340–351, Association for Computational Linguistics.

[3] Kung-Hsiang Huang and Nanyun Peng, "Document-level event extraction with efficient end-to-end learning of cross-event dependencies," in *Proceedings of the Third Workshop on Narrative Understanding*, Virtual, June 2021, pp. 36–47, Association for Computational Linguistics.

[4] Runxin Xu, Peiyi Wang, Tianyu Liu, Shuang Zeng, Baobao Chang, and Zhifang Sui, "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*, Seattle, United States, July 2022, pp. 5025–5036, Association for Computational Linguistics.

[5] Yubing Ren, Yanan Cao, Fang Fang, Ping Guo, Zheng Lin, Wei Ma, and Yi Liu, "CLIO: Role-interactive multi-event head attention network for document-level event extraction," in *Proceedings of the 29th International Conference on Computational Linguistics*, Gyeongju, Republic of Korea, Oct. 2022, pp. 2504–2514, International Committee on Computational Linguistics.

[6] Hao Li, Yanan Cao, Yubing Ren, Fang Fang, Lanxue Zhang, Yingjie Li, and Shi Wang, "Intra-event and inter-event dependency-aware graph network for event argument extraction," in *Findings of the Association for Computational Linguistics: EMNLP 2023*, Houda Bouamor, Juan Pino, and Kalika Bali, Eds., Singapore, Dec. 2023, pp. 6362–6372, Association for Computational Linguistics.

[7] Kaiwen Wei, Xian Sun, Zequn Zhang, Jingyuan Zhang, Guo Zhi, and Li Jin, "Trigger is not sufficient: Exploiting frame-aware knowledge for implicit event argument extraction," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online, Aug. 2021, pp. 4672–4682, Association for Computational Linguistics.

[8] Jian Liu, Yufeng Chen, and Jinan Xu, "Machine reading comprehension as data augmentation: A case study on implicit event argument extraction," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic, Nov. 2021, pp. 2716–2725, Association for Computational Linguistics.

[9] Sha Li, Heng Ji, and Jiawei Han, "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*, Online, June 2021, pp. 894–908, Association for Computational Linguistics.

[10] Yubing Ren, Yanan Cao, Ping Guo, Fang Fang, Wei Ma, and Zheng Lin, "Retrieve-and-sample: Document-level event argument extraction via hybrid retrieval augmentation," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, Eds., Toronto, Canada, July 2023, pp. 293–306, Association for Computational Linguistics.

[11] Jiaju Lin, Qin Chen, Jie Zhou, Jian Jin, and Liang He, "Cup: Curriculum learning based prompt tuning for implicit event argument extraction," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, Lud De Raedt, Ed. 7 2022, pp. 4245–4251, International Joint Conferences on Artificial Intelligence Organization, Main Track.

[12] Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao, "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)*, Dublin, Ireland, May 2022, pp. 6759–6774, Association for Computational Linguistics.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June 2019, pp. 4171–4186, Association for Computational Linguistics.

[14] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer, "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*, Online, July 2020, pp. 7871–7880, Association for Computational Linguistics.

[15] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li, "Listwise approach to learning to rank: theory and algorithm," in *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, William W. Cohen, Andrew McCallum, and Sam T. Roweis, Eds. 2008, vol. 307 of *ACM International Conference Proceeding Series*, pp. 1192–1199, ACM.

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

[17] Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme, "Multi-sentence argument linking," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020, pp. 8057–8077, Association for Computational Linguistics.

[18] Xinya Du and Claire Cardie, "Event extraction by answering (almost) natural questions," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online, Nov. 2020, pp. 671–683, Association for Computational Linguistics.