A Two-Stream AMR-enhanced Model for Document-level Event Argument Extraction

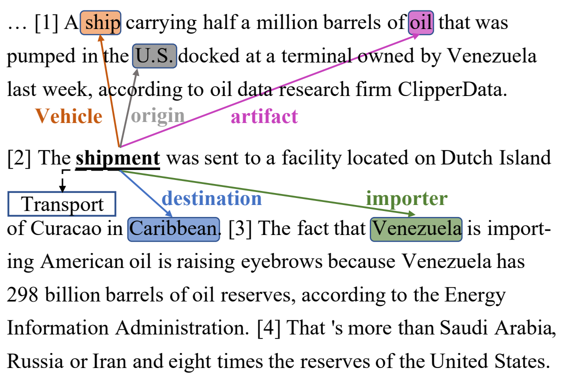
Anonymous ACL submission

Abstract

Most previous studies aim at extracting events from a single sentence, while document-level event extraction still remains under-explored. In this paper, we focus on extracting event arguments from an entire document, which mainly faces two critical problems: a) the long-distance dependency between trigger and arguments over sentences; b) the distracting context towards an event in the document. To address these issues, we propose a **T**wo-**S**tream **A**bstract meaning **R**epresentation enhanced extraction model (**TSAR**). TSAR encodes the document from different perspectives by a two-stream encoding module, to utilize local and global information and lower the impact of distracting context. Besides, TSAR introduces an AMR-guided interaction module to capture both intra-sentential and inter-sentential features, based on the locally and globally constructed AMR semantic graphs. An auxiliary boundary loss is introduced to enhance the boundary information for text spans explicitly. Extensive experiments illustrate that TSAR outperforms previous state-of-the-art by a large margin, with 2.54 F1 and 5.13 F1 performance gain on the public RAMS and WikiEvents datasets respectively, showing the superiority in the cross-sentence arguments extraction. We will release our code upon acceptance.

1 Introduction

Event Argument Extraction (EAE) aims at identifying the entities that serve as event arguments, and predicting the roles they play in the event, which is a key step for Event Extraction (EE). It helps to transform the unstructured text into structured event knowledge that can be further utilized in recommendation systems (Li et al., 2020), dialogue systems (Zhang et al., 2020a), and so on. Most previous studies assume that the events are only expressed by a single sentence and hence focus on sentence-level extraction (Chen et al., 2015; Liu et al., 2018; Zhou et al., 2021). However, in


… [1] A ship carrying half a million barrels of oil that was pumped in the U.S. docked at a terminal owned by Venezuela last week, according to oil data research firm ClipperData.

[2] The **shipment** was sent to a facility located on Dutch Island of Curacao in Caribbean. [3] The fact that Venezuela is importing American oil is raising eyebrows because Venezuela has 298 billion barrels of oil reserves, according to the Energy Information Administration. [4] That 's more than Saudi Arabia, Russia or Iran and eight times the reserves of the United States.

Figure 1: A document from RAMS dataset (Ebner et al., 2020). A *transport* event is triggered by **shipment**, with five event arguments scattering across the document.

real-life scenarios, the events are often described through a whole document consisting of multiple sentences (e.g., a news article or a financial report), which still remains under-explored.

Figure 1 illustrates an example of document-level EAE, in which a *Transport* event is triggered by *shipment*. Different from sentence-level EAE, extracting arguments out of the entire document faces two critical challenges. (1) **Long-distance dependency** among trigger and arguments. The arguments are usually located in different sentences from the trigger word and their distance can be quite far away. In Figure 1, while the trigger *shipment* is in Sentence 2, the *vehicle*, *origin*, *artifact*, and *importer* arguments are located in Sentence 1 or 3, which highly increases the extraction difficulty. To accommodate the long-range extraction, not only intra-sentential but also inter-sentential semantics should be well modeled. (2) **Distracting context**. While a document naturally encompasses more context than a single sentence, some distracting context can mislead the argument extraction. As shown in Figure 1, the *origin* argument *U.S.* can be identified more easily without Sentence 4, which does not offer useful information for the event, but contains many place entities that can be distracting, like *Saudi Arabia* and *Russia or Iran*. It remains challenging to pinpoint the useful context while discarding the distracting information.

Recently, Du and Cardie (2020a) use a tagging-based method, which fails to handle nested arguments. Instead, span-based methods predict argument roles for candidate spans (Ebner et al., 2020; Zhang et al., 2020b). Some studies directly generate arguments based on sequence-to-sequence model (Li et al., 2021). However, how to model long-distance dependency among trigger and arguments, and how to handle distracting context explicitly, remain largely under-explored.

In this paper, to tackle the aforementioned two problems, we propose a **T**wo-**S**tream **AMR**-enhanced extraction model (**TSAR**). In order to take advantage of the essential context in the document, and avoid being misled by distractions, we introduce a two-stream encoding module. It consists of a global encoder that encodes global semantics with as much context as possible to gather adequate context information, and a local encoder that focuses on the most essential information and prudently takes in extra context. In this way, TSAR can leverage complementary advantages of different encoding perspectives, and therefore make better use of the feasible context to benefit the extraction. Besides, to model the long-distance dependency, we introduce an AMR-guided interaction module. Abstract Meaning Representation (AMR) (Banarescu et al., 2013) graph contains rich hierarchical semantic relations among different concepts, which makes it favorable for complex event extraction. From such a linguistic-driven angle, we turn the linear structure of the document into both global and local graph structures, followed by a graph neural network to enhance the interactions, especially for those non-local elements. Finally, as TSAR extracts arguments in span level, where the span boundary may be ambiguous, we introduce an auxiliary boundary loss to enhance span representation with calibrated boundary.

To summarize, our contributions are three-fold. 1) We propose a two-stream encoding module for document-level EAE, which encodes the document through two different perspectives to better utilize the context. 2) We introduce an AMR-guided interaction module to facilitate the semantic interactions within the document, so that long-distance dependency can be better captured. 3) Our experiments show that TSAR outperforms the previous start-of-the-art model by large margins, with 2.54 F1 and 5.13 F1 improvements on public RAMS and WikiEvents datasets respectively, especially on cross-sentence event arguments extraction.

2 Related Work

2.1 Sentence-level Event Extraction

Previous studies mainly focus on sentence-level event extraction. Li et al. (2014) and Judea and Strube (2016) use handcrafted features to extract events from the sentence. Chen et al. (2015) firstly propose a neural pipeline model to extract events, while Nguyen et al. (2016) utilize a joint model to mitigate error propagation. To better model the interactions among words, Liu et al. (2018); Yan et al. (2019); Ma et al. (2020) make use of the dependency tree, and Wadden et al. (2019) enumerates all possible spans and propagate information in the span graph. Data augmentation is also considered (Yang et al., 2019). Moreover, some works try to reformulate the event extraction task as other tasks. For example, Du and Cardie (2020b) and Zhou et al. (2021) cast event extraction as question answering, and Xiangyu et al. (2021) model it as a sequence-to-sequence task. However, all of these models can only extract events from a single sentence. Thus, they fail to handle the much more common cases, where event arguments usually spread over multiple sentences within the document.

2.2 Document-level Event Extraction

In order to extract events from a whole piece of article with multiple sentences, document-level event extraction has attracted more and more attention recently. Yang and Mitchell (2016) utilize well-defined features to extract arguments across sentences, while most recent methods are based on neural networks. Some studies first identify entities in the document, followed by assigning these entities as specific argument roles (Yang et al., 2018; Zheng et al., 2019; Xu et al., 2021). Differently, some studies try to jointly extract entities and argument roles simultaneously, which can be further divided into tagging-based and span-based methods. Tagging-based methods directly conduct sequence labeling for each token in the document with BIO-schema (Du and Cardie, 2020a; Veyseh et al., 2021), while span-based methods predict the argument role for candidate text spans which usually have a maximum length limitation (Ebner et al.,

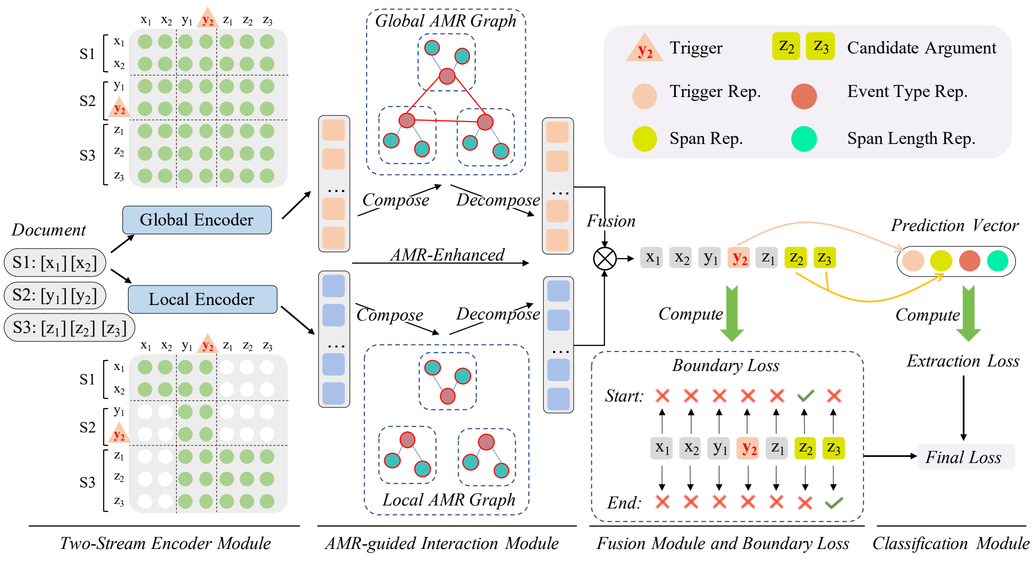


Figure 2: **Overview of our TSAR**. Firstly, taking an entire document as input, TSAR first encodes the document by the two-stream encoding module, where the global and local encoders with different attention reception fields are used to capture the context in different scopes. Then the AMR-guided interaction module constructs global AMR graphs and local ones to stimulate the interactions among concepts in the document, especially those far away from each other, based on graph neural network. Next, the information fusion module fuses the two-stream representations, and also strengthens the boundary information through a boundary loss. Finally, the classification module makes predictions for candidate spans. For conciseness, we assume the document has three sentences, S_1, S_2, S_3, and the event is triggered by y_2 with $[z_2, z_3]$ being a candidate argument span.

2020; Zhang et al., 2020b). Another line of studies reformulate the task as a sequence-to-sequence task (Du et al., 2021a,b; Li et al., 2021), or machine reading comprehension task (Wei et al., 2021).

As a span-based method, TSAR is different from prior methods that simply encode it as a long sentence. Instead, TSAR introduces a two-stream encoding module and AMR-guided interactions module to model intra-sentential and inter-sentential semantics, along with an auxiliary boundary loss to enhance span boundary information.

3 Task Formulation

Following Ebner et al. (2020), we formulate doc-level event argument extraction as follows. We define that a document $\mathcal{D}$ consists of N sentences, and a sentence is comprised of a sequence of words, i.e., $\mathcal{D} = \{w_1, w_2, \ldots, w_{|\mathcal{D}|}\}$, and $\mathrm{SEN}(w_i) \in [1, N]$ refers to the sentence that w_i belongs to. We also define the event types set $\mathcal{E}$ and the corresponding argument roles set $\mathcal{R}_e$ for each event type $e \in \mathcal{E}$. Then, given a document $\mathcal{D}$ and the trigger $t \in \mathcal{D}$ triggering the event type $e \in \mathcal{E}$, the task aims to detect all (r, s) pairs for the event, where $r \in \mathcal{R}_e$ is an argument role for the event type e, and $s \subseteq \mathcal{D}$ is a contiguous text span in the document.

4 Methodology

Figure 2 shows the overall architecture of our model TSAR. The document is fed into the two-stream encoding module, followed by the AMR-guided interaction module to derive both global and local contextualized representations. The information fusion module fuses these two-stream representations, and the classification module finally predicts argument roles for candidate spans.

4.1 Two-Stream Encoding Module

Although more context is provided by the document, it also inevitably introduces irrelevant and distracting information towards the event. These noise signals can be harmful to the argument extraction as shown in Figure 1. To capture useful information and filter distracting one, we propose a two-stream encoding module, consisting of a global encoder that is aware of all context, and a local encoder that only prudently focuses on the most essential information. Therefore, we can leverage their complementary advantages to make better use of the context information.

Specifically, the global and local encoders share the same Transformer-based pre-trained language model such as BERT. By controlling the reception

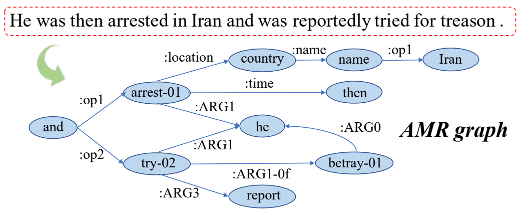


Figure 3: The AMR graph provides abstract and logical semantic information, where the nodes denote the concepts and the edges refer to different relation types. The corresponding text spans for nodes are omitted.

field of the words in the self-attention module , we can encode the document from different perspectives. In the global encoder, the attention technique is the same as the traditional Transformer:

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

where Q, K, V refers to query, key, and value matrix, and d_m is the model dimension. However, in the local encoder, we introduce a mask matrix M, such that tokens can only attend to the sentence itself and the sentence where the trigger locates, to avoid redundant distracting information:

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

$$M_{ij}=\begin{cases}0, & \text{SEN}\left(w_j\right)\in\left\{\text{SEN}\left(w_i\right),\text{SEN}\left(t\right)\right\}\\ -\infty, & Otherwise\end{cases}$$

where $\text{SEN}\left(w_i\right)$ is the sentence that the word w_i belongs to, and t refers to the trigger of the event.

Hence, we encode the document with two different streams, a global encoder Encoder^{G} and a local encoder Encoder^{L}, finally deriving two representations, Z^G and Z^L:

$$\begin{aligned}Z^{G}&=\left[z_1^{G},z_2^{G},\ldots,z_{|\mathcal{D}|}^{G}\right]=\text{Encoder}^{G}\left(\left[w_1,w_2,\ldots,w_{|\mathcal{D}|}\right]\right)\\ Z^{L}&=\left[z_1^{L},z_2^{L},\ldots,z_{|\mathcal{D}|}^{L}\right]=\text{Encoder}^{L}\left(\left[w_1,w_2,\ldots,w_{|\mathcal{D}|}\right]\right)\end{aligned}$$

4.2 AMR-Guided Interaction Module

One key challenge to extract arguments from the document is to capture the intra-sentential and inter-sentential features. Therefore, we propose an AMR-guided interaction module that adopts Abstract Meaning Representation (AMR, Banarescu et al., 2013) graph to provide rich semantic structure to facilitate the interactions among concepts, which also offers logical meanings of the document from a linguistic-driven perspective to benefit the language understanding.

AMR semantic graph models the meaning representations of a sentence as a rooted, directed, labeled graph. Concretely, with an AMR parser, a natural sentence can be parsed into an AMR graph $G=(V,E)$. The node $v=(a,b)\in V$ represents a concept that corresponds to the span ranging from w_a to w_b in the origin sentence, while the edge represents a specific AMR relation (detail in Appendix A). Thus, AMR focuses on semantic relations rather than syntactic ones, which is more high-level and beneficial to event understanding, and the structures are more close to the event trigger-arguments structures. For example, Figure 3 demonstrates how a sentence is parsed into an AMR semantic graph. As event arguments play essential roles in the text, most of them would be involved, if not all, in the AMR graphs (90% and 88% arguments in RAMS and WikiEvents datasets).

The AMR-guided interaction module is attached after the global and local encoders as shown in Figure 2. We use the AMR graphs as skeletons for information interactions, under a *composition, interaction, and decomposition* paradigm.

From the local perspective, we construct AMR graphs for each sentence in the document, and they are isolated from each other. For initialization, the vector representation of node $u=(a_u,b_u)$ is ***composed*** by averaging the local representations of its corresponding text span:

$$h_u^0=\frac{1}{|b_u-a_u+1|}\sum_{i=a_u}^{b_u}z_i^L$$

We then use L-layer stacked Graph Convolution Network (Kipf and Welling, 2017) to model the ***interactions*** among different concept nodes through edges with different relation types. Given node u at the l-th layer, the information interaction and aggregation operation is defined as follows:

$$h_u^{(l+1)}=\text{ReLU}\left(\sum_{k\in\mathcal{K}}\sum_{v\in\mathcal{N}_k(u)\bigcup\{u\}}\frac{1}{c_{u,k}}W_k^{(l)}h_v^{(l)}\right)$$

where $\mathcal{K}$ denotes different relation types, $\mathcal{N}_k(u)$ denotes the neighbors for u connected with k-th relation types and $c_{u,k}$ is a normalization constant. Besides, $W_k^{(l)}\in\mathbb{R}^{d_m\times d_m}$ is a trainable parameter.

Finally, we concatenate vectors in all layers and derive the final node representation by $h_u=W_1[h_u^0;h_u^1;\ldots;h_u^L]\in\mathbb{R}^{d_m}$. Then h_u is ***decomposed*** into the local representations of corresponding words, followed by token-wise aggregation, where $\mathbb{I}(\cdot)$ refers to the indication function:

$$\widetilde{h}_i^L = z_i^L + \frac{\sum_u \mathbb{I}(a_u <= i \wedge b_u >= i) h_u}{\sum_u \mathbb{I}(a_u <= i \wedge b_u >= i)}$$

From the global perspective, we first construct the global AMR graphs by fully connecting the root nodes of AMR graphs of different sentences, since the root nodes contain the core semantics according to the AMR core-semantic principle (Cai and Lam, 2019) [1]. Then similar graph-based interaction methods are used to obtain the AMR-enhanced global representations $\widetilde{h}_i^G$, but based on global AMR graphs instead. In this way, the inter-sentential information can flow through the sentence boundaries, and therefore long-distance dependency can also be better captured.

4.3 Information Fusion Module

In the information fusion module, we fuse the global representations $\widetilde{H}^G = \left[\widetilde{h}_1^G, \widetilde{h}_2^G, \ldots, \widetilde{h}_{|\mathcal{D}|}^G\right]$ and local representations $\widetilde{H}^L = \left[\widetilde{h}_1^L, \widetilde{h}_2^L, \ldots, \widetilde{h}_{|\mathcal{D}|}^L\right]$, to construct the final vector representations for the candidate spans.

In detail, we use a gated fusion to control how much information is incorporated from the two-stream representations. Given $\widetilde{h}_i^G$ and $\widetilde{h}_i^L$, we calculate the gate vector g_i with trainable parameters W_2 and W_3, $g_i = \text{sigmoid}(W_2 \widetilde{h}_i^G + W_3 \widetilde{h}_i^L + b)$. Then we derive the fused representations $\widetilde{h}_i$:

$$\widetilde{h}_i = g_i \odot \widetilde{h}_i^G + (1 - g_i) \odot \widetilde{h}_i^L$$

For a candidate text span ranging from w_i to w_j, its fused representation consists of the start representation $\widetilde{h}_i^{start}$, the end representation $\widetilde{h}_j^{end}$ and the average pooling of the hidden state of the span with $W_{span} \in \mathbb{R}^{d_m \times (3 \times d_m)}$:

$$s_{i:j} = W_{span} \left[\widetilde{h}_i^{start}; \widetilde{h}_i^{end}; \frac{1}{j - i + 1} \sum_{k=i}^{j} \widetilde{h}_k \right]$$

where $\widetilde{h}_i^{start} = W_s \widetilde{h}_i$ and $\widetilde{h}_i^{end} = W_e \widetilde{h}_i$.

Since we extract arguments in span level, whose boundary may be ambiguous, we introduce an auxiliary boundary loss to enhance boundary information for the $\widetilde{h}_i^{start}$ and $\widetilde{h}_i^{end}$. In detail, we predict whether the word w_i is the first or last word of a golden argument span with token-wise classifiers. We use a linear transformation followed by a sigmoid function, to derive the probability of the word w_i being the first or last word of a golden argument span, i.e., P_i^s and P_i^e.

$$P_i^s = \text{sigmoid}\left(W_4 \widetilde{h}_i^{start}\right), P_i^e = \text{sigmoid}\left(W_5 \widetilde{h}_i^{end}\right)$$

Finally, the boundary loss is defined as the following cross-entropy losses of detecting the start and end position.

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

where, y_i^s and y_i^e denote the golden labels. In this way, we introduce an explicit supervision signal to inject boundary information of the start and end representation of an span, which is shown to be necessary and important to the extraction in our exploring experiments.

4.4 Classification Module

In the classification module, we predict what argument role the candidate span plays, or it does not belong to any specific argument roles. Besides the span representation $s_{i:j}$, we also consider the trigger, event type, and the length of the span. Specifically, we concatenate the following representations to obtain the final prediction vector $I_{i:j}$: 1) the trigger representation $\widetilde{h}_t$, and the span representation $s_{i:j}$, with their absolute difference $\left|\widetilde{h}_t - s_{i:j}\right|$, and element-wise multiplication, $\widetilde{h}_t \odot s_{i:j}$; 2) the embedding of the event type E_{type}. 3) the embedding of the span length E_{len};

$$I_{i:j} = \left[\widetilde{h}_t; s_{i:j}; \left|\widetilde{h}_t - s_{i:j}\right|; \widetilde{h}_t \odot s_{i:j}; \mathrm{E}_{type}; \mathrm{E}_{len}\right]$$

We then use the cross entropy $\mathcal{L}_c$ as loss function:

$$\mathcal{L}_c = -\sum_{i=1}^{|\mathcal{D}|} \sum_{j=i}^{|\mathcal{D}|} y_{i:j} \log P\left(r_{i:j} = y_{i:j}\right) \tag{2}$$

where $y_{i:j}$ is the golden argument role, and $P\left(r_{i:j}\right)$ is derived by a feed-forward network based on $I_{i:j}$.

Finally, we train the model in an end-to-end way with the final loss function $\mathcal{L} = \mathcal{L}_c + \lambda \mathcal{L}_b$ with hyperparameter λ.

[1] We find more elaborate methods yield no further improvements, so we adopt this simple connection paradigm.

Method	Dev		Test	
	Span F1	Head F1	Span F1	Head F1
BERT-CRF	38.1	45.7	39.3	47.1
BERF-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
FEAE	-	-	47.40	-
TSAR$_{\text{base}}$ (Ours)	**45.23**	**51.70**	**48.06**	**55.04**
BART-Gen	-	-	48.64	57.32
TSAR$_{\text{large}}$ (Ours)	**49.23**	**56.76**	**51.18**	**58.53**

Table 1: **Comparison between TSAR and other methods on RAMS dataset**. Models above the double line are based on BERT$_{\text{base}}$. TSAR consistently outperforms others on Span F1 and Head F1. Compared with BART-Gen, TSAR improves 2.54 Span F1 in the test set.

Method	Arg Identification		Arg Classification	
	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$_{\text{base}}$ (Ours)	**75.52**	**73.17**	**68.11**	**66.31**
BART-Gen	71.75	72.29	64.57	65.11
TSAR$_{\text{large}}$ (Ours)	**76.62**	**75.52**	**69.70**	**68.79**

Table 2: **Comparison between TSAR and other methods on WikiEvents dataset**. Models above the double line are based on BERT$_{\text{base}}$. TSAR yields evident improvements in argument identification and classification sub-tasks. Compared with BART-Gen, TSAR improves Head F1 in argument classification by 5.13 score.

5 Experiments

5.1 Datasets

We evaluate our model on two public document-level event argument extraction datasets, RAMS v1.0 (Ebner et al., 2020) and WikiEvents (Li et al., 2021). RAMS contains 9,124 human-annotated examples, with 139 event types and 65 kinds of argument roles, and more than 21k arguments. WikiEvents is another human-annotated dataset, with 50 event types and 59 event argument roles, and more than 3.9k events. We follow the official train/dev/test split for RAMS and WikiEvents datasets, and use the evaluation script provided by Ebner et al. (2020) to evaluate the performance. The detailed statistics are provided in Appendix B.

5.2 Experiment Setups and Metrics

In our implementation, we use BERT$_{\text{base}}$ (Devlin et al., 2019) and RoBERTa$_{\text{large}}$ (Liu et al., 2019) as our backbone encoder for TSAR, with global and local encoders sharing parameters. Detailed hyperparameters are listed in Appendix C.

Following Zhang et al. (2020b), we report the Span F1 and Head F1 for RAMS dataset. Span F1 requires the predicted argument spans to fully match the golden ones, while Head F1 relaxes the constraint and evaluates solely on the head word of the argument span. The head word of a span is defined as the word that has the smallest arc distance to the root in the dependency tree. In addition, following Li et al. (2021), we report the Head F1 and Coref F1 scores for WikiEvents dataset. The model is given full credit in Coref F1 if the extracted argument is coreferential with the golden argument as used by Ji and Grishman (2008).

5.3 Main Results

We compare TSAR with the following baselines. 1) **BERT-CRF** (Shi and Lin, 2019) is a tagging-based method, which adopts a BERT-based BIO-styled sequence labeling model. 2) **Two-Step** (Zhang et al., 2020b) is a span-based method, which first identifies the head word of possible argument span, and then extends to the full span. **BERT-CRF$_{\text{TCD}}$** and **Two-Step$_{\text{TCD}}$** refers to adopting Type-Constraint Decoding mechanism as used in (Ebner et al., 2020). 3) **FEAE** (Wei et al., 2021), Frame-aware Event Argument Extraction, is a concurrent work based on question answering. 4) **BERT-QA** (Du and Cardie, 2020c) is also a QA-based model. BERT-QA and BERT-QA-Doc extract run on sentence-level and document-level, respectively. 5) **BART-Gen** (Li et al., 2021) formulate the task as a sequence-to-sequence task and uses BART$_{\text{large}}$ (Lewis et al., 2020) to generate corresponding arguments in a predefined format.

Table 1 illustrates the results in both dev and test set on RAMS dataset. As is shown, among models based on BERT$_{\text{base}}$, TSAR outperforms other previous methods. For example, TSAR yields an improvement of 4.93 $\sim$ 7.13 Span F1 and 3.70 $\sim$ 6.00 Head F1 compared with the previous method in the dev set, and up to 8.76 Span F1 in the test set. Besides, among models based on large pre-trained language models, TSAR outperforms BART-Gen by 2.54 Span F1 and 1.21 Head F1[2]. These results suggest that encoding the document in a two-stream way, and introducing AMR graphs to facilitate interactions, is beneficial to capturing

[2]We use TSAR$_{\text{large}}$ based on RoBERTa$_{\text{large}}$ to compare with BART-Gen based on BART$_{\text{large}}$, as they are pre-trained on the same corpus with the same batch size and training steps.

Method	d=-2	d=-1	d=0	d=1	d=2
BERT-CRF	14.0	14.0	41.2	15.7	4.2
Two-Step	15.6	15.3	43.4	17.8	8.5
FEAE	23.7	19.3	49.2	**25.0**	5.4
TSAR$_{\text{base}}$ (Ours)	**24.3**	**21.9**	**49.6**	24.6	**11.9**
BART-Gen	24.3	28.1	52.4	24.8	20.8
TSAR$_{\text{large}}$ (Ours)	**28.6**	**30.6**	**53.1**	**27.1**	**22.3**

Table 3: **Span F1 in RAMS dataset with different sentence distance between trigger and arguments**. Most improvements by TSAR come from cross-sentence ($d \neq 0$) arguments extraction.

Method	Dev		Test	
	Span F1	Head F1	Span F1	Head F1
TSAR$_{\text{large}}$	49.23	56.76	51.18	58.53
- Global Encoder	46.71	54.26	48.21	55.49
- Local Encoder	48.43	55.44	48.69	56.82
- AMR-guided Graph	48.63	55.24	49.21	56.70
- Boundary Loss	47.93	55.14	50.47	57.75

Table 4: **Ablation study on RAMS for TSAR$_{\text{large}}$**. The score would decrease without any kind of module.

intra-sentential and inter-sentential features, and thus improves the performance.

Moreover, we follow Li et al. (2021) to evaluate both argument identification and argument classification, and report the Head F1 and Coref F1. Identification requires the model to correctly detect the argument span boundary, while classification has to further correctly predict its argument role. As illustrated in Table 2, TSAR consistently outperforms others in both tasks. Compared with BART-Gen, TSAR improves up to 4.87/3.23 Head/Coref F1 for argument identification, and 5.13/3.68 Head/Coref F1 for argument classification. Similar results also appear among models based on BERT$_{\text{base}}$, with $5.69 \sim 36.37$ and $11.95 \sim 33.34$ Head F1 improvement for identification and classification. These results show that TSAR is superior to other methods in not only detecting the boundary of argument spans, but also predicting their roles.

6 Analysis

6.1 Cross-sentence Argument Extraction

Since there are multiple sentences in the document, some event arguments are located far away from the trigger, which highly increases the difficulty of extraction. To explore the effect of handling such cross-sentence arguments of our TSAR, we divide the event arguments in RAMS dataset into five bins according to the sentence distance between arguments and trigger, i.e., $d = \{-2, -1, 0, 1, 2\}$. We report the Span F1 on the RAMS dev set for different methods. As shown in Table 3, the Span F1 for cross-sentence arguments ($d \neq 0$) is much lower than local arguments ($d = 0$), suggesting the huge challenge to capture long-distance dependency between triggers and cross-sentence arguments. However, TSAR still surpasses other strong baselines. In detail, TSAR$_{\text{base}}$ improves 0.4 and TSAR$_{\text{large}}$ improves 0.7 F1 compared with the previous state-of-the-art, respectively. More importantly, when extracting cross-sentence arguments, TSAR$_{\text{base}}$ and TSAR$_{\text{large}}$ yield an improvement of up to 2.3 and 2.7 on average. The results support our claims that TSAR is good at capturing both intra-sentential and inter-sentential features, especially the long-distance between trigger and arguments.

6.2 Ablation Study

We conduct an ablation study to explore the effectiveness of different modules in TSAR. Table 4 show the results on RAMS datasets for TSAR$_{\text{large}}$. We also provide results for TSAR$_{\text{base}}$, and those on WikiEvents datasets in Appendix D.

Firstly, we remove the global or local encoder in the two-stream encoding module. As shown in Table 4, the removal causes drop in performance, e.g., 3.04 and 1.71 Head F1 drop on the test set without global and local encoder. It suggests the global and local encoders are complementary to each other, and both of them are necessary for TSAR.

Secondly, once we remove the AMR-guided interaction module, the Head F1 would decrease by 1.83 on the test set. It shows the semantic structure provided by AMR graphs is helpful to the arguments extraction of the document.

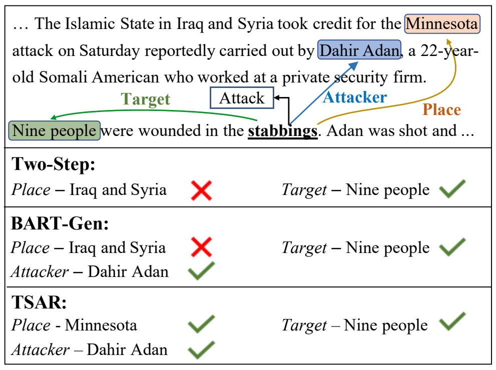
Finally, the removal of boundary loss causes the boundary information lost in span representations, which also leads to 1.62 and 0.78 Head F1 decrease on dev and test set.

6.3 Case Study

In this section, we show a specific case of the extraction results among different methods. As shown in Figure 5, *stabbings* triggers an *Attack* event with three arguments in color. Since *Nine people* is located near the trigger, all the methods correctly predict it as the *target*. However, extracting *Minnesota* and *Dahir Adan* asks for capturing long-distance dependency. Although Two-Step and BART-Gen wrongly predict the *place* as *Iraq and Syria*, and

Category	Examples	Errors	
		Two-step	TSAR
Wrong Span	It was Bush’s administration$_{participant}$, not [Obama]$_{participant}$ ’s, that **negotiated**$_{meet}$ the 2009 agreement from Iraq by Dec. 31, 2011.	86	81
Over-extract	The information minister alleged that oil **smuggled**$_{smuggle}$ into Turkey was bought by [the Turkish president ’s son]$_{transporter}$, who owns an oil company …	64	48
Partial	[280 victims]$_{victim}$,including women, children and old people $_{victim}$ … The **massacre**$_{die}$ is considered as one of the worst mass killings committed in Syria …	47	57
Overlap	Richard, the man accused of **punching**$_{injury}$ a [69-year-old protester]$_{victim}$ $_{victim}$ outside a Donald Trump rally in Asheville, NC on Monday, is vigorously …	32	28
Wrong Role	The investigation found 100 people were linked to the **transport**$_{disperseseparate}$ and … It found the missile had been driven from [Russia]$_{destination}$ $_{origin}$ into an …	46	19

Figure 4: **Error analysis on RAMS dataset**. The triggers are in **bold** with corresponding event types in green. The underlined spans refer to golden arguments, with their roles in blue. The [bracketed] spans denote the predicted arguments, with their roles noted in red. We illustrate the number of different kinds of errors for Two-step and our TSAR, which has 275 and 233 errors in total, respectively. Compared with Two-step, TSAR decreases errors in most error categories, especially for *Wrong Role* and *Over-extract*.



… The Islamic State in Iraq and Syria took credit for the Minnesota attack on Saturday reportedly carried out by Dahir Adan, a 22-year-old Somali American who worked at a private security firm.
Nine people were wounded in the **stabbings**. Adan was shot and …

Two-Step:
Place – Iraq and Syria ✗ *Target* – Nine people ✓

BART-Gen:
Place – Iraq and Syria ✗ *Target* – Nine people ✓
Attacker – Dahir Adan ✓

TSAR:
Place - Minnesota ✓ *Target* – Nine people ✓
Attacker – Dahir Adan ✓

Figure 5: **An extraction case**, where an *Attack* event is triggered by **stabbings** with three arguments. TSAR manages to extract the cross-sentence argument *Minnesota* far from the trigger, while other methods fail.

Two-Step even fails to extract the *Attacker*, TSAR manage to extract the cross-sentence arguments. It can be attributed to that our AMR-enhanced module catches *Minnesota* is the *place* of *attack* that is highly related to the trigger *stabbings* in semantics.

6.4 Error Analysis

To further explore the errors made by different models and analyze the reasons in detail, we randomly choose 200 examples from the RAMS test set and compare the predictions with golden annotations manually. We divide the errors into five categories, which is shown in Figure 4. *Wrong Span* refers to assigning a specific role to a wrong span non-overlapped with the golden one. We find it is usually due to the negative words like *not*, and the coreference spans for the golden one. *Over-extract* denotes the model predicts an argument role while it does not exist in the document. Some extracted spans are the sub-strings of the golden spans (*Partial*), or have some overlaps with them (*Overlap*). These two kinds of errors are usually attributed to the annotation inconsistency in the dataset, such as whether the adjective, quantifier, and article (e.g., *a* and *the*) before the noun should belong to the golden argument. Besides, the *Partial* error also usually occurs in cases where there is punctuation like a comma in the golden span as shown in Figure 4. Finally, though the model succeeds to identify the golden span, it can still assign wrong argument role to the span (*Wrong Role*). We compare the errors of Two-step$_{\text{TCD}}$ and TSAR$_{\text{base}}$. We observe TSAR decrease the number of errors from 275 to 233, especially for *Wrong Role* and *Over-extract*, with 27 and 16 errors reduction, respectively.

7 Conclusion

It is challenging to extract event arguments from a whole document, owing to the long-distance dependency between trigger and arguments over sentences and the distracting context. To tackle these problems, we propose **T**wo-**S**tream **A**MR-enhanced extraction model (**TSAR**). TSAR uses two-stream encoders to encode the document from different perspectives, followed by an AMR-guided interaction module to facilitate the document-level semantic interactions. An auxiliary boundary loss is introduced to enhance the boundary information for spans. Experiments on RAMS and WikiEvents datasets demonstrate that TSAR outperform previous state-of-the-art methods by a large margin, with 2.51 and 5.13 F1 improvements respectively, especially for cross-sentence argument extraction.

References

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*.

Deng Cai and Wai Lam. 2019. Core semantic first: A top-down approach for AMR parsing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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 (ACL)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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 (NAACL)*.

Xinya Du and Claire Cardie. 2020a. Document-level event role filler extraction using multi-granularity contextualized encoding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

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)*.

Xinya Du and Claire Cardie. 2020c. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Xinya Du, Alexander Rush, and Claire Cardie. 2021a. GRIT: Generative role-filler transformers for document-level event entity extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Xinya Du, Alexander Rush, and Claire Cardie. 2021b. Template filling with generative transformers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

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)*.

Ramón Fernandez Astudillo, Miguel Ballesteros, Tahira Naseem, Austin Blodgett, and Radu Florian. 2020. Transition-based parsing with stack-transformers. In *Findings of the Association for Computational Linguistics (EMNLP)*.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Alex Judea and Michael Strube. 2016. Incremental global event extraction. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations (ICLR)*.

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)*.

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 *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations (ACL)*.

Qi Li, Heng Ji, Yu Hong, and Sujian Li. 2014. Constructing information networks using one single model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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 (NAACL)*.

Xiao Liu, Zhunchen Luo, and Heyan Huang. 2018. Jointly multiple events extraction via attention-based graph information aggregation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (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. *arXiv preprint*, arXiv:1907.11692.

Jie Ma, Shuai Wang, Rishita Anubhai, Miguel Ballesteros, and Yaser Al-Onaizan. 2020. Resource-enhanced neural model for event argument extraction. In *Findings of the Association for Computational Linguistics (EMNLP)*.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Peng Shi and Jimmy Lin. 2019. Simple BERT models for relation extraction and semantic role labeling. *arXiv preprint*, arXiv:1904.05255.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Quan Hung Tran, Varun Manjunatha, Lidan Wang, Rajiv Jain, Doo Soon Kim, Walter Chang, and Thien Huu Nguyen. 2021. Inducing rich interaction structures between words for document-level event argument extraction. In *Advances in Knowledge Discovery and Data Mining - 25th Pacific-Asia Conference (PAKDD)*.

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 (EMNLP)*.

Minjie Wang, Lingfan Yu, Da Zheng, Quan Gan, Yu Gai, Zihao Ye, Mufei Li, Jinjing Zhou, Qi Huang, Chao Ma, Ziyue Huang, Qipeng Guo, Hao Zhang, Haibin Lin, Junbo Zhao, Jinyang Li, Alexander J. Smola, and Zheng Zhang. 2019. Deep graph library: Towards efficient and scalable deep learning on graphs. *arXiv preprint*, arXiv:1909.01315.

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 *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP)*.

Xi Xiangyu, Wei Ye, Shikun Zhang, Quanxiu Wang, Huixing Jiang, and Wei Wu. 2021. Capturing event argument interaction via a bi-directional entity-level recurrent decoder. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Runxin Xu, Tianyu Liu, Lei Li, and Baobao Chang. 2021. Document-level event extraction via heterogeneous graph-based interaction model with a tracker. In *The Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Haoran Yan, Xiaolong Jin, Xiangbin Meng, Jiafeng Guo, and Xueqi Cheng. 2019. Event detection with multi-order graph convolution and aggregated attention. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Hang Yang, Yubo Chen, Kang Liu, Yang Xiao, and Jun Zhao. 2018. DCFEE: A document-level Chinese financial event extraction system based on automatically labeled training data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019. Exploring pre-trained language models for event extraction and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Tianran Zhang, Muhao Chen, and Alex A. T. Bui. 2020a. 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)*.

Zhisong Zhang, Xiang Kong, Zhengzhong Liu, Xuezhe Ma, and Eduard Hovy. 2020b. A two-step approach for implicit event argument detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zixuan Zhang and Heng Ji. 2021. Abstract Meaning Representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Shun Zheng, Wei Cao, Wei Xu, and Jiang Bian. 2019. Doc2EDAG: An end-to-end document-level framework for Chinese financial event extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yang Zhou, Yubo Chen, Jun Zhao, Yin Wu, Jiexin Xu, and JinLong Li. 2021. What the role is vs. what plays the role: Semi-supervised event argument extraction via dual question answering. In *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*.

A Abstract Meaning Representation (AMR) Graph

To obtain AMR semantic graphs, we use the AMR parser proposed by Fernandez Astudillo et al. (2020), which is a state-of-the-art AMR parser and can achieve satisfactory results for downstream application (up to 81.3 Smatch on AMR2.0 data). As the number of AMR relation types is large, which results in too many demanded parameters, we follow Zhang and Ji (2021) to cluster the relation types into main categories as shown in Table 5.

Categories	Relation Types
Spatial	location, destination, path
Temporal	year, time, duration, decade, weekday
Means	instrument, manner, topic, medium
Modifiers	mod, poss
Operators	op-X
Prepositions	prep-X
Sentence	snt
Core Roles	ARG0, ARG1, ARG2, ARG3, ARG4
Others	other relation types

Table 5: Similar AMR relation types are clustered into the same relation category. The exception is that ARGx is still treated as an individual relation type.

B Statistics of Datasets

The detailed data statistics of RAMS and WikiEvents datasets are shown in Table 6.

Dataset	Split	# Doc.	# Event	# Argument
RAMS	Train	3,194	7,329	17,026
	Dev	399	924	2,188
	Test	400	871	2,023
WikiEvents	Train	206	3,241	4,542
	Dev	20	345	428
	Test	20	365	566

Table 6: Statistics of RAMS and WikiEvents datasets.

C Hyperparameters Setting

We set the dropout rate to 0.1, batch size to 8, and train TSAR using Adam (Kingma and Ba, 2015) as optimizer with 3e-5 learning rate. We train TSAR for 50 epochs for RAMS dataset and 100 epochs for WikiEvents dataset. We search the boundary loss weight λ from $\{0.1, 0.3, 0.5\}$, and L from $\{3, 4\}$, and select the best model using dev set. The maximum length of spans is limited to 20 at most. Our code is based on Transformers (Wolf et al., 2020) and DGL libraries (Wang et al., 2019). We conduct experiments in a GTX-3090 GPU.

D Ablation Study

In the main body of the paper, we illustrate the results of the ablation study for TSAR$_{\text{large}}$ on RAMS dataset. To thoroughly show the effect of different modules of TSAR, we also provide the results of the ablation study for TSAR$_{\text{base}}$ on RAMS dataset. Table 7 shows the results on RAMS dataset, from which we can observe removing different modules would cause $1.34 \sim 2.77$ Span F1 on test set.

Besides, we do ablation study on WikiEvents. As shown in Table 8, the Head F1 decreases by $0.70 \sim 2.02$ and $0.88 \sim 2.96$ for Arg Identification and Arg Classification sub-tasks respectively, once different modules are removed from TSAR$_{\text{base}}$. Similar conclusions can be drawn from the results of TSAR$_{\text{large}}$, which is shown in Table 9.

Method	Dev		Test	
	Span F1	Head F1	Span F1	Head F1
TSAR$_{\text{base}}$	45.23	51.70	48.06	55.04
- Global Encoder	43.05	50.90	45.29	53.62
- Local Encoder	44.63	51.34	46.50	53.26
- AMR-guided Graph	43.57	50.80	45.97	52.85
- Boundary Loss	44.42	51.08	46.72	53.91

Table 7: **Ablation study on RAMS for TSAR$_{\text{base}}$.** The score would decrease without any kind of module.

Method	Arg Identification		Arg Classification	
	Head F1	Coref F1	Head F1	Coref F1
TSAR$_{\text{base}}$	75.52	73.17	68.11	66.31
- Global Encoder	73.50	72.23	65.15	64.07
- Local Encoder	74.40	72.62	67.11	65.41
- AMR-guided Graph	73.88	72.45	65.83	64.94
- Boundary Loss	74.82	72.50	67.23	65.95

Table 8: **Ablation study on WikiEvents for TSAR$_{\text{base}}$.** The performance of identification and classification would decrease without any kind of module.

Method	Arg Identification		Arg Classification	
	Head F1	Coref F1	Head F1	Coref F1
TSAR$_{\text{large}}$	76.62	75.52	69.70	68.79
- Global Encoder	74.12	72.80	67.54	66.41
- Local Encoder	74.60	73.32	68.08	66.88
- AMR-guided Graph	74.52	73.82	67.67	66.54
- Boundary Loss	75.50	74.05	68.60	67.33

Table 9: **Ablation study on WikiEvents for TSAR$_{\text{large}}$.** The performance of identification and classification would decrease without any kind of module.