Revisiting Event Argument Extraction: Can EAE Models Learn Better When Being Aware of Event Co-occurrences?

Yuxin He[1] and **Jingyue Hu**[1] and **Buzhou Tang**[1,2,†]
[1]Department of Computer Science, Harbin Institute of Technology, Shenzhen, China
[2]Peng Cheng Laboratory, Shenzhen, China
21S051047@stu.hit.edu.cn
tangbuzhou@gmail.com

Abstract

Event co-occurrences have been proved effective for event extraction (EE) in previous studies, but have not been considered for event argument extraction (EAE) recently. In this paper, we try to fill this gap between EE research and EAE research, by highlighting the question that *"Can EAE models learn better when being aware of event co-occurrences?"*. To answer this question, we reformulate EAE as a problem of table generation and extend a SOTA prompt-based EAE model into a non-autoregressive generation framework, called TabEAE, which is able to extract the arguments of multiple events in parallel. Under this framework, we experiment with 3 different training-inference schemes on 4 datasets (ACE05, RAMS, WikiEvents and MLEE) and discover that via training the model to extract all events in parallel, it can better distinguish the semantic boundary of each event and its ability to extract single event gets substantially improved. Experimental results show that our method achieves new state-of-the-art performance on the 4 datasets. Our code is avilable at https://github.com/Stardust-hyx/TabEAE.

1 Introduction

Event argument extraction (EAE) is an essential subtask of event extraction (EE). Given an input text and trigger(s) of target event(s), the EAE task aims to extract all argument(s) of each target event. Recently, substantial progress has been reported on EAE, thanks to the success of pre-trained language models (PLMs).

Previous studies on EE commonly take event co-occurrences into account. However, recent works on EAE (Ebner et al., 2020; Zhang et al., 2020; Xu et al., 2022; Du and Cardie, 2020; Wei et al., 2021; Liu et al., 2021; Li et al., 2021; Du et al., 2021; Lu et al., 2021; Ma et al., 2022) only consider one event at a time and ignore event co-occurrences (as illustrated in Figure 1). In fact, event co-occurrences always exisit in text and they are useful in revealing event correlation and contrasting the semantic structures of different events. For the instance in Figure 1, there exist two events in the same context. The two events are triggered by "leaving", "become" respectively, and share the same subject "Davies". It is clear that there exists a strong causal correlation between the two events. However, mainstream works on EAE split the instance into two samples, which conceals this correlation.

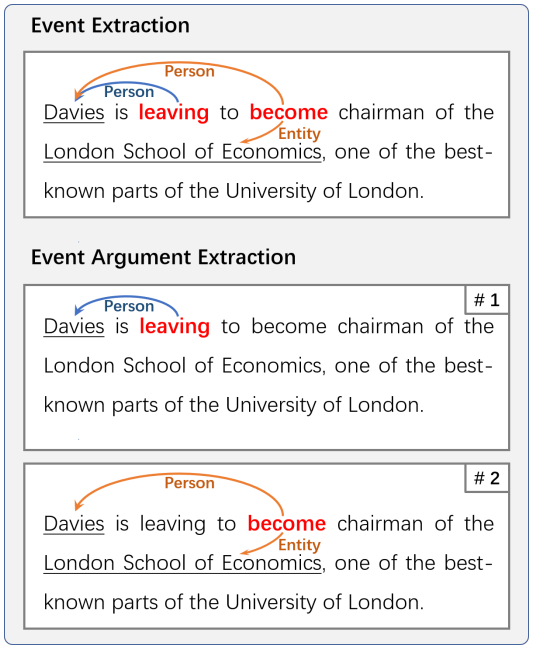


Figure 1: An illustration of EE and EAE. The triggers are in red and the arguments are underlined. EE models aim at extracting all events concurrently, whereas mainstream EAE models are trained to extract the arguments for one event trigger at a time.

In this paper, we try to resolve this divergence between EE research and EAE research, by highlighting the question that *"Can EAE models learn better*

[†]Corresponding Author.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 12542–12556
July 9-14, 2023

when being aware of event co-occurrences?". To address this question, we reformulate EAE as a problem of table generation and extend the SOTA prompt-based EAE model, PAIE (Ma et al., 2022), into a non-autoregressive generation framework to extract the arguments of multiple events concurrently. Our framework, called TabEAE, inherits the encoding, prompt construction and span selection modules from PAIE, but employs a novel non-autoregressive decoder for table generation.

Under this framework, we explore three kinds of training-inference schemes: (1) *Single-Single*, training model to extract single event at a time and infer in the same way; (2) *Multi-Multi*, training model to extract all events in parallel and infer in the same way; (3) *Multi-Single*, training model to extract all events in parallel and let the model extract single event at a time during inference. According to our experiments, the Multi-Single scheme works the best on 3 benchmarks (ACE, RAMS and WikiEvents) and the Multi-Multi scheme works the best on the MLEE benchmark, where the phenomenon of nested events extensively exists. Besides, in-depth analysis reveals that via training TabEAE to extract all events in parallel, it can better capture the semantic boundary of each event and its ability to extract single event at a time gets substantially improved.

To sum up, our contributions include:

- We observe the divergence between EE research and EAE research in terms of the phenomenon of event co-occurrence. To resolve this divergence, we extend the SOTA prompt-based EAE model PAIE into a text-to-table framework, TabEAE, which is able to extract the arguments of multiple events concurrently.
- Under the TabEAE framework, we explore three training-inference schemes, i.e. Single-Single, Multi-Multi, Multi-Single, and verify the significance of event co-occurrence for EAE.
- The proposed method outperforms SOTA EAE methods by 1.1, 0.4, 0.7 and 2.7 in Arg-C F1 respectively on the 4 benchmarks ACE05, RAMS, WikiEvents and MLEE.

2 Related Work

2.1 Event Argument Extraction

As a crucial subtask of EE, EAE has long been studied. In the early stages, EAE is only treated as a component of EE systems (Chen et al., 2015; Nguyen et al., 2016; Yang et al., 2018; Zheng et al., 2019; Lin et al., 2020), where the phenomenon of event co-occurrence is always taken into account.

Recently, more and more works study EAE as a stand-alone problem. We summarize these recent works on EAE into 4 categories: (1) span-based methods that identify candidate spans and predict the roles of them (Ebner et al., 2020; Zhang et al., 2020; Xu et al., 2022); (2) QA-based methods that query arguments using questions constructed with predefined templates (Du and Cardie, 2020; Wei et al., 2021; Liu et al., 2021); (3) sequence-to-sequence methods that leveraging generative PLMs, e.g. BART (Lewis et al., 2020) and T5 (Raffel et al., 2020), to sequentially generate all arguments of the target event (Li et al., 2021; Du et al., 2021; Lu et al., 2021); (4) a prompt-based method by Ma et al. (2022) that leverages slotted prompts to extract arguments in a generative slot-filling manner.

Among them, the prompt-based method, PAIE (Ma et al., 2022), demonstrates SOTA performance. However, all of them only consider one event at a time, diverging from EE research. In this work, we adapt PAIE into a non-autoregressive table generation framework, which is able to extract the arguments of multiple events in parallel.

2.2 Text-to-Table

Although table-to-text (Bao et al., 2018; Chen et al., 2020) is a well-studied problem in the area of controllable natural language generation, Text-to-Table, the inverse problem of table-to-text, is just newly introduced by Wu et al. (2022). In Wu et al. (2022), text-to-table is solved with a sequence-to-sequence model enhanced with table constraint and table relation embeddings. In contrast, our table generation framework constructs the slotted table input based on given trigger(s) and predefined prompt(s), and generate in a non-autoregressive manner.

3 Methodology

In this section, we will first give an formal definition of EAE and then introduce TabEAE, our solution to the task in detail.

3.1 Task Definition

An instance of EAE has the general form of $(\mathbf{x}, \{t_i\}_{i=1}^{N}, \{e_i\}_{i=1}^{N}, \{R^{e_i}\}_{i=1}^{N}, \{A_i\}_{i=1}^{N})$, where $\mathbf{x}$ is the text (a sentence or a document), N is the

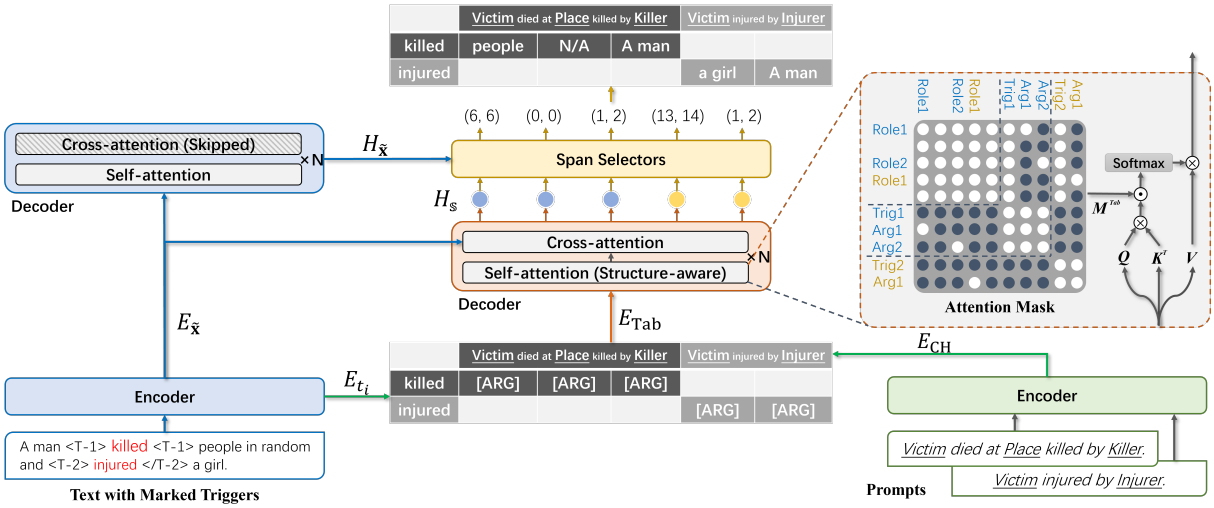


Figure 2: An overview of TabEAE, our non-autoregressive table generation framework. It includes four components: (1) Trigger-aware context encoding (in blue); (2) Slotted table construction (in green); (3) Non-autoregressive table decoding (in orange); (4) Span selection (in gold). The two encoders here share the same parameters and the two decoders here share the same parameters. The prompts here are simplified and shorten due to the space limit.

number of target events, t_i is the trigger of i-th event, e_i is the type of i-th event, R^{e_i} is the set of argument roles associated with the event type e_i, A_i is the set of arguments of the i-th event and each $a^{(r)} \in A_i$ is a textual span within $\mathbf{x}$ that represents the role $r \in R^{e_i}$. Different from the formulation by previous research on EAE that only considers one event for an input instance, this formulation takes all events co-occurring in the same context into consideration, providing a more comprehensive view of the problem.

3.2 TabEAE

Our solution to EAE is a non-autoregressive table generation framework, namely TabEAE, which is derived from the SOTA prompt-based EAE model PAIE. Figure 2 gives an overview of the framework. A detailed description of each component comes as follows.

3.2.1 Trigger-aware Context Encoding

Given an input text $\mathbf{x} = x_1, x_2, ..., x_L$ with a set of event triggers, we first mark each trigger with a pair of markers (<T-i>, </T-i>), where i counts the order of occurrence. Note that, there may be multiple events sharing the same trigger, in which case the shared trigger will only be marked once. After that, we tokenize the marked text into

$$\tilde{\mathbf{x}} = [\text{<s>}, x_1, x_2, ..., \text{<T-1>}, x_{t_1}, \text{</T-1>}, \quad (1)$$

$$..., \text{<T-}i\text{>}, x_{t_i}, \text{</T-}i\text{>}, ..., x_L, \text{</s>}] \quad (2)$$

where x_{t_i} is the text fragment of the i-th trigger.

By feeding $\tilde{\mathbf{x}}$ into a transformer-based encoder, we can get the encoding of the text:

$$E_{\tilde{\mathbf{x}}} = \text{Encoder}(\tilde{\mathbf{x}}) \quad (3)$$

We follow PAIE (Ma et al., 2022), to further decodes $E_{\tilde{\mathbf{x}}}$ with a decoder to obtain the event-oriented context representation:

$$H_{\tilde{\mathbf{x}}} = \text{Decoder}(E_{\tilde{\mathbf{x}}}) \quad (4)$$

3.2.2 Slotted Table Construction

The decoder input is constructed as a slotted table, where the column header is the concatenation of event-schema prompt(s) proposed by PAIE. Considering the example in Figure 2, there are a Life-die event with trigger "kills" and a Life-injure event with trigger "injured". Then the column header is *"Victim (and Victim) died at Place (and Place) killed by Killer (and Killer). Victim (and Victim) injured by Injurer (and Injurer)."*, where the first sentence is the prompt for Life-die event, the second sentence is the prompt for Life-injure event, and each underlined part is named after a argument role, acting as the head of a column. There are multiple columns sharing the same argument role for the extraction of multiple arguments playing the same role in an event.

We initialize the representation of column header by feeding each prompt into the encoder in parallel and concatenating the encoding outputs:

$$E_{\text{PR}_j} = \text{Encoder}(\text{PR}_j) \quad (5)$$

$$E_{\text{CH}} = [E_{\text{PR}_1} : ... : E_{\text{PR}_j} : ... : E_{\text{PR}_M}] \quad (6)$$

where PR_j is the j-th prompt, M is the number of event type(s).

The i-th row of the table starts with the i-th trigger, followed by a sequence of argument slots S_i. The initial representation of the i-th trigger, E_{t_i}, is copied from the encoding of the marked text. And the initial representation of argument slots, E_{S_i}, is the average of the encoding of corresponding argument roles (in the column header) and the encoding of corresponding trigger markers. We denote all the argument slots as $\mathbb{S} = \{S_i\}_{i=1}^{N}$.

The initial representations of table components are row-wise concatenated to obtain the initial representation of the table:

$$E_{\text{Tab}} = [E_{\text{CH}} : E_{t_1} : E_{S_1} : ... : E_{t_N} : E_{S_N}] \quad (7)$$

3.2.3 Non-autoregressive Table Decoding

The non-autoregressive decoder iteratively updates the representation of input table via structure-aware self-attention inner the table as well as cross-attention between the table and the encoder output.

Structure-aware Self-attention We devise a structure-aware self-attention mask, M^{Tab}, so that each element of the table can only attend to the region related to it. Our design is as follows:

- All tokens within the column header attend to each other.
- All tokens within the column header attend to the event trigger(s).
- Each role along with corresponding argument slot(s) attend to each other.
- Each event trigger along with corresponding argument slot(s) attend to each other.

Note that this attention mask is only used for the decoding of slotted table. When computing $H_{\tilde{\mathbf{x}}}$ (Equation 4), we employ normal self-attention.

The cross-attention mechanism is the same as the one in Transformer (Vaswani et al., 2017) and it is only employed to decode the table. When computing $H_{\tilde{\mathbf{x}}}$ (Equation 4), it is skipped.

3.2.4 Span Selection

With the output of table decoding, H_{Tab}, we can obtain the final representation of the argument slots, $H_{\mathbb{S}} \subset H_{Tab}$. We transform each representation vector $\mathbf{h}_{s_k} \in H_{\mathbb{S}}$ into a span selector $\{\Phi_{s_k}^{\text{start}}, \Phi_{s_k}^{\text{end}}\}$ (Du and Cardie, 2020; Ma et al., 2022):

$$\Phi_{s_k}^{\text{start}} = \mathbf{h}_{s_k} \odot \mathbf{w}^{\text{start}} \quad (8)$$

$$\Phi_{s_k}^{\text{end}} = \mathbf{h}_{s_k} \odot \mathbf{w}^{\text{end}} \quad (9)$$

where $\mathbf{w}^{\text{start}}$ and $\mathbf{w}^{\text{end}}$ are learnable weights, and $\odot$ represents element-wise multiplication.

The span selector $\{\Phi_{s_k}^{\text{start}}, \Phi_{s_k}^{\text{end}}\}$ is responsible for selecting a span $(\hat{start}_k, \hat{end}_k)$ from the text to fill in the argument slot s_k:

$$\text{logit}_k^{\text{start}} = H_{\tilde{\mathbf{x}}} \Phi_{s_k}^{\text{start}} \in \mathbb{R}^L \quad (10)$$

$$\text{logit}_k^{\text{end}} = H_{\tilde{\mathbf{x}}} \Phi_{s_k}^{\text{end}} \in \mathbb{R}^L \quad (11)$$

$$\text{score}_k(l, m) = \text{logit}_k^{\text{start}}[l] + \text{logit}_k^{\text{end}}[m] \quad (12)$$

$$(\hat{start}_k, \hat{end}_k) = \underset{(l,m):\ 0<m-l<L}{\arg\max} \text{score}_k(l, m)$$

where l or m represents the index of arbitrary token within the text.

Note that, there can be more than one argument playing the same role in an event, requiring further consideration for the assignment of golden argument spans during training. Hence, we follow (Carion et al., 2020; Yang et al., 2021; Ma et al., 2022) to fine tune our model with the Bipartite Matching Loss. The loss for an training instance is defined as

$$P_k^{\text{start}} = \text{Softmax}(\text{logit}_k^{\text{start}}) \quad (13)$$

$$P_k^{\text{end}} = \text{Softmax}(\text{logit}_k^{\text{end}}) \quad (14)$$

$$\mathcal{L} = -\sum_{i=1}^{N} \sum_{(start_k, end_k) \in \delta(A_i)} (\log P_k^{\text{start}}[start_k] + \log P_k^{\text{end}}[end_k]) \quad (15)$$

where $\delta(\cdot)$ represents the optimal assignment calculated with Hungarian algorithm (Kuhn, 1955) according to the assignment cost devised by (Ma et al., 2022), and $(start_k, end_k)$ is the golden span optimally assigned to the k-th argument slot. For an argument slot relating to no argument, it is assigned with the empty span $(0, 0)$.

3.3 Three Training-Inference Schemes

Under the TabEAE framework, there exist three possible training-inference schemes: (1) *Single-Single*, train TabEAE to extract single event at a time and infer in the same way; (2) *Multi-Multi*, train TabEAE to extract all events in parallel and infer in the same way; (3) *Multi-Single*, train TabEAE to extract all events in parallel and let it extract single event at a time during inference. For the *Single* mode, only one trigger is marked in the input text; for the *Multi* mode, all the triggers are marked in the text. Note that, when trained to extract all events in parallel, TabEAE also learn to extract single event, since a great portion of training instances has only one event.

Scheme	Model	PLM	ACE05		RAMS		WikiEvents		MLEE	
			Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C	Arg-I	Arg-C
Single-Single	EEQA (2020)	BERT	70.5	68.9	48.7	46.7	56.9	54.5	68.4	66.7
	EEQA (2020)*	RoBERTa	72.1	70.4	51.9	47.5	60.4	57.2	70.3	68.7
	BART-Gen (2021)	BART	69.9	66.7	51.2	47.1	66.8	62.4	71.0	69.8
	TSAR (2022)	BERT	-	-	56.1	51.2	70.8	65.5	72.3	71.3
	TSAR (2022)*	RoBERTa	-	-	57.0	52.1	71.1	65.8	72.6	71.5
	PAIE (2022)	BART	75.7	72.7	56.8	52.2	70.5	65.3	72.1	70.8
	PAIE (2022)*	RoBERTa	76.1	73.0	57.1	52.3	70.9	65.5	72.5	71.4
	DEGREE (2022)	BART	76.0	73.5	-	-	-	-	-	-
	DEGREE (2022)*	RoBERTa	76.6	73.9	-	-	-	-	-	-
	TabEAE (Ours)	RoBERTa	75.5	72.6	57.0	52.5	70.8	65.4	71.9	71.0
Multi-Multi	TabEAE (Ours)	RoBERTa	75.9	73.4	56.7	51.8	71.1	66.0	**75.1**	**74.2**
Multi-Single	TabEAE (Ours)	RoBERTa	**77.2**	**75.0**	**57.3**	**52.7**	**71.4**	**66.5**	72.0	71.3

Table 1: Main results on four benchmarks. Both RoBERTa and BART here are of large-scale (with 24 Transformer layers). * means we replace the original PLM with RoBERTa and rerun their code (hyperparameter tuning is conducted when necessary). The highest scores are in bold font and the second-highest scores are underlined.

4 Experiments

4.1 Implementation Details

We implement TabEAE with Pytorch and run the experiments with a Nvidia Tesla A100 GPU. We instantiate the encoder with the first 17 layers of RoBERTa-large (Liu et al., 2019).[1] The weight of the self-attention layers and feedforward layers of the decoder is initialized with the weight of the remaining 7 layers of RoBERTa-large. The setting of 17-layer encoder + 7-layer decoder is found to be optimal by our experiment (See Appendix C). Note that the cross-attention part of the decoder is newly initialized in random and we set its learning rate to be 1.5 times the learning rate of other parameters. We leverage the AdamW optimizer (Loshchilov and Hutter, 2017) equipped with a linear learning rate scheduler to tune our model. See Appendix B for details of hyperparameter tuning.

4.2 Experiment Setups

Datasets We experiment with 4 datasets, including ACE05 (Doddington et al., 2004), RAMS (Ebner et al., 2020), WikiEvents (Li et al., 2021) and MLEE (Pyysalo et al., 2012). ACE05 is a sentence-level dataset, while the others are in document-level. The corpora of ACE05, RAMS and WikiEvents mainly consist of news, while the corpus of MLEE lies in the biomedical domain. Besides, the phenomenon of nested event is commonly observed in MLEE, but rare in the other 3 datasets. See Appendix A for a detailed description of the datasets.

Evaluation Metrics Following previous works (Li et al., 2021; Ma et al., 2022), we measure the performance with two metrics: (1) strict argument identification F1 (Arg-I), where a predicted argument of an event is correct if its boundary matches any golden arguments of the event; (2) strict argument classification F1 (Arg-C), where a predicted argument of an event is correct only if its boundary and role type are both correct. All the reported results are averaged over 5 runs with different random seeds.

4.3 Compared Methods

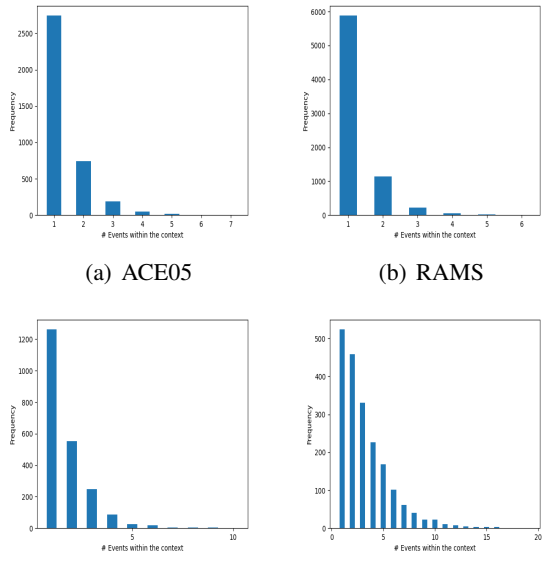
We compare TabEAE with several SOTA methods:

- **EEQA** (Du and Cardie, 2020), a QA-based EAE model that treats EAE as a machine reading comprehension problem;
- **BART-Gen** (Li et al., 2021), a seq-to-seq EAE model that generates predicted arguments conditioned on event template and context;
- **TSAR** (Xu et al., 2022), a two-stream AMR-enhanced span-based EAE model;
- **PAIE** (Ma et al., 2022), a prompt-based EAE model that leverages slotted prompts to obtain argument span selectors;
- **DEGREE** (Hsu et al., 2022), a data-efficient model that formulates EAE as a conditional generation problem.

[1] We choose RoBERTa-large for a fair comparison with EAE methods based on BART-large, as the two PLMs adopt the same tokenizer and are pre-trained on the same corpus.

Model	Scheme	ACE05		RAMS		WikiEvents		MLEE	
		# Ev = 1 [185]	# Ev > 1 [218]	# Ev = 1 [587]	# Ev > 1 [284]	# Ev = 1 [114]	# Ev > 1 [251]	# Ev = 1 [175]	# Ev > 1 [2025]
PAIE (2022)	Single-Single	70.97	73.88	52.72	52.14	65.31	65.37	78.91	70.11
TabEAE	Single-Single	71.21	73.83	52.82	51.61	65.27	65.46	79.26	70.32
	Multi-Multi	**73.38**	73.45	**52.87**	50.82	**67.30**	65.32	**81.13**	**73.60**
	Multi-Single		**76.13**		**52.49**		**66.19**		69.97

Table 2: Comparison of EAE models with different training-inference schemes in terms of their performance on instances with different numbers of events. The number in brackets is the number of supporting events in the test set. Unless otherwise specified, we only measure the Arg-C F1 for the experiments in the Analysis section.



(a) ACE05 (b) RAMS

(c) WikiEvents (d) MLEE

Figure 3: Distributions of the number of events per instance on the four datasets.

4.4 Main Results

The overall performances of compared baselines and TabEAE are illustrated in Table 1. We find that TabEAE (Single-Single) is competitive to previous SOTA models (TSAR, DEGREE and PAIE) on the four benchmarks. This is expected since these models follow the same training-inference scheme and leverage PLMs of the same scale.

In the mean time, TabEAE (Multi-Single) outperforms the SOTA model by 0.6 Arg-I F1 and 1.1 Arg-C F1 on ACE05, by 0.2 Arg-I F1 and 0.4 Arg-C F1 on RAMS, by 0.3 Arg-I F1 and 0.7 Arg-C F1 WikiEvents.

As for the MLEE dataset, TabEAE (Multi-Multi) performs better than TabEAE (Multi-Single) and yields 2.5 Arg-I F1 gain, 2.7 Arg-C F1 gain compared to SOTA models.

We analyze the reason behind the results in §5.1.

5 Analysis

5.1 The Effect of Training-inference Scheme

To analyze the influence of the training-inference scheme, we measure the performances of EAE models with different training-inference schemes on handling instances with different numbers of events. The results are shown in Table 2. We can see that PAIE (Single-Single) and TabEAE (Single-single) have similar capacity in extracting stand-alone events and co-occurring events.

When trained to extract all the events in parallel, the Arg-C F1 of TabEAE on instances with single event increases by 2.17, 0.05, 2.03 and 1.87 on the 4 datasets respectively. However, by letting TabEAE extract all the events in parallel during inference, the Arg-C F1 on instances with multiple events drops by 0.38, 0.79, 0.14 on ACE, RAMS and WikiEvents respectively, while increasing by 3.28 on MLEE. We believe this phenomenon is the result of two factors:

1. The distribution of the number of events per instance. As plotted in Figure 3, there are much more instances with multiple events on WikiEvents and MLEE than on ACE05 and RAMS. Hence, the model is better trained to extract multiple events concurrently on WikiEvents and MLEE.
2. Difficulty. Generally, it is more difficult for a model to extract all the events in one pass. But it is not the case for the MLEE dataset, since there are around 32.9% of the arguments acting as triggers of other events in MLEE, and when all triggers are provided (as in the Multi-Multi scheme), it become easier for the model to extract all the arguments.

When training TabEAE to extract all events in parallel and letting it extract one event at a time during inference, the Arg-C F1 of TabEAE on instances with multiple events increases by 2.3, 0.88,

Scheme	ACE05		RAMS		WikiEvents		MLEE	
	N-O [319]	Overlap [84]	N-O [690]	Overlap [181]	N-O [296]	Overlap [69]	N-O [1460]	Overlap [734]
Single-Single	71.1	78.6	51.6	55.6	65.7	64.4	75.4	65.8
Multi-Multi	-	-	-	-	-	-	78.1 (+2.7)	69.4 (+3.6)
Multi-Single	72.8 (+1.7)	80.8 (+2.2)	51.7 (+0.1)	56.1 (+0.5)	66.4 (+0.7)	66.9 (+2.5)	-	-

Table 3: Comparison of TabEAE with different training-inference schemes in terms of their capacity to extract the arguments of overlapping events (events with shared arguments). The number in square brackets is the number of supporting events in the test set. **N-O**: Non-overlapping.

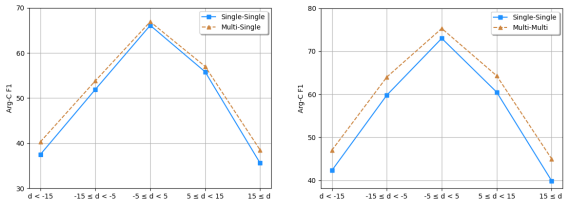
0.73 on ACE, RAMS and WikiEvents respectively. This is reasonable, since there is a large portion of instances having only one event, which means the model is also well-trained to extract one event at a time under the Multi-Single scheme.

5.2 Capturing the Event Semantic Boundary

We hypothesize that the performance gains yielded by the Multi-Multi and Multi-Single schemes rooted in the stronger ability of TabEAE to capture the event semantic boundary. To verify this, we further measure the model's ability to capture the event semantic boundary from two points of view: (1) Inter-event Semantic; (2) Inner-event Semantic.

From the view of **inter-event semantic**, we compare the performance of TabEAE with different training-inference schemes in terms of their ability to extract the arguments of overlapping events (i.e., events with shared arguments). As illustrated in Table 3, when trained to extract all events concurrently, the model's performance gains of extracting the arguments of overlapping events are much higher than that of extracting the arguments of non-overlapping events. Specifically, the differences of performance gains are 0.5 Arg-C F1 on ACE05, 0.4 Arg-C F1 on RMAS, 1.8 Arg-C F1 on WikiEvents and 0.9 Arg-C F1 on MLEE. This suggests that TabEAE can better distinguish the semantic boundary between overlapping events.

From the view of **inner-event semantic**, we compare the performance of TabEAE with different training-inference schemes in terms of their ability to extract arguments of different distances to the triggers. We define the distance here as the head word index of an argument minus the head word index of its corresponding trigger. The experiments are conducted on the document-level datasets WikiEvents and MLEE, where the distance distribution of event arguments is more disperse. The results are plotted in Figure 4. We can observe that, when equipped with the Multi-Multi/Multi-Single schemes the model's performance gains of extracting remote arguments are higher than the performance gains of extracting nearby arguments. This means TabEAE gets better at extracting arguments around the event boundary.



(a) WikiEvents (b) MLEE

Figure 4: Comparison of TabEAE with different training-inference schemes in terms of their ability to extract arguments of different distances to the triggers. d is the word distance from a trigger to an argument (the value is negative when the argument is on the left side of the trigger).

Model	ACE05	RAMS	WikiEvents	MLEE
TabEAE	75.0	52.7	66.5	74.2
w/o SAAM	73.1	51.2	65.4	72.7
w/o PET	70.8	49.3	61.9	69.9
w/o Prompts	72.5	50.9	64.8	71.3
BERT → BART	72.7	51.0	65.4	72.4

Table 4: Results of ablation study. For ACE05, RAMS and WikiEvents, we experiment with TabEAE (Multi-Single); for MLEE, we experiment with TabEAE (Multi-Multi). SAAM: Structure-aware Attention Mask. PET: Pre-computed Encodings of the input Table.

5.3 Ablation Study

To verify the effectiveness of different components of TabEAE, we conduct ablation study on the 4 datasets. The results are illustrated in Table 4.

After removing the **structure-aware attention mask**, the Arg-C F1 scores drop by 1.9, 1.5, 1.1, 1.5 on ACE05, RAMS, WikiEvents and MLEE respectively. This demonstrates the benefit of letting

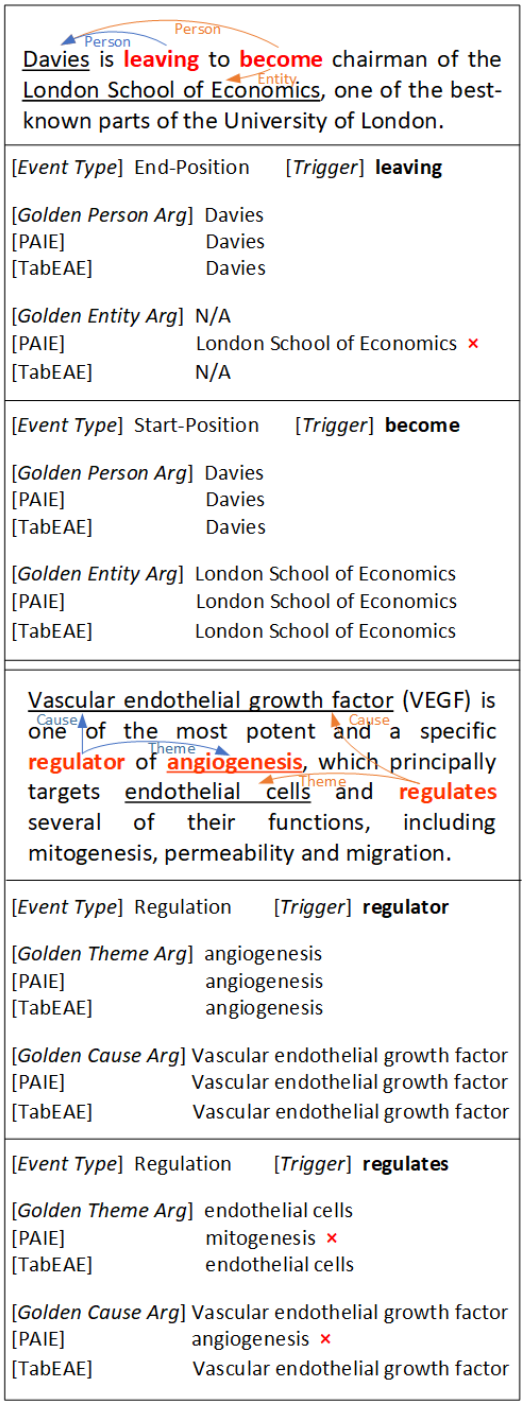


Figure 5: Two test cases from ACE05 and MLEE.

each table token only paying attention to the table region related to it.

After replacing the **pre-computed encodings of the input table** with RoBERTa token embeddings, the Arg-C F1 scores drop by 4.2, 3.4, 4.6, 3.9 on the 4 datasets. This proves the necessity of initializing the embeddings of input table with the encodings computed by the encoder.

When constructing the table column header with the concatenation of argument roles instead of **prompts**, the Arg-C F1 scores drop by 2.5, 1.8, 1.7 and 2.9 on the 4 datasets respectively. This coincides with the finding by (Ma et al., 2022) that hand-crafted prompts can be of great help to the task of EAE.

When replacing the encoder/decoder of TabEAE with **BART** encoder/decoder, the model performance degrades by 2.3, 1.7, 1.1, 1.8 on the 4 datasets respectively. The reason behind this degradation should be the uni-directional self-attention employed by BART decoder is not suitable for the decoding of table.

5.4 Case Study

Figure 5 illustrates 2 test cases from ACE05 and MLEE respectively. In the first test case, there exist 2 events triggered by "leaving" and "become", with a shared argument "Davies". PAIE incorrectly predicts "London School of Economics" as an argument of the event triggered by "leaving", which is essentially an argument of the event triggered by "become". In contrast, TabEAE is able to avoid this mistake, demonstrating a stronger capacity to capture the event semantic boundary.

In the second test case, there exist 3 events triggered by "regulator", "regulates" and "angiogenesis" respectively. Among them, the event triggered by "angiogenesis" has no argument. For the event triggered by "regulates", PAIE fails to extract the remote argument "Vascular endothelial growth factor", while TabEAE correctly extracts it by being aware of the co-occurring event that shares this argument.

6 Conclusion

In this paper, we point out that recent studies on EAE ignore event co-occurrences, resulting in a divergence from main-stream EE research. To remedy this, we highlight the question that *"Can EAE models learn better when being aware of event co-occurrences"* and explore it with a novel text-to-table framework, *TabEAE*, that can extract multiple event in parallel. By experimenting with 3 training-inference schemes on 4 datasets, we find that when trained to extract all event concurrently, TabEAE can better capture the event semantic boundary and its ability to extract single event gets greatly improved. Our work demonstrates the significance of event co-occurrence for EAE and establishes a new foundation for future EAE research.

7 Limitations

In this section, we summarize the limitations of our work as follows:

- There is still a lot more to explore in terms of event co-occurrence for EAE (e.g., iterative extraction, course learning, etc.). We are unable to cover all in this work and will explore further in the future.
- As demonstrated by our ablation study, the high performance of our model greatly relies on the manual prompts. This limits the application of our model to the scenes where high-quality prompts are unavailable and difficult to construct. To address this, we should look into the area of automatic prompt construction.
- Our work ignores the phenomenon of entity co-reference commonly existing in narrative documents. This limits the model's ability to figure out the underlying relation between entities, which is crucial for the task of EAE. And we will take entity co-references into account in our future works.

Acknowledgments

We thank the reviewers for their insightful comments and valuable suggestions. This study is partially supported by National Key R&D Program of China (2021ZD0113402), National Natural Science Foundations of China (62276082, U1813215 and 61876052), National Natural Science Foundation of Guangdong, China (2019A1515011158), Major Key Project of PCL (PCL2021A06), Strategic Emerging Industry Development Special Fund of Shenzhen (20200821174109001) and Pilot Project in 5G + Health Application of Ministry of Industry and Information Technology & National Health Commission (5G + Luohu Hospital Group: an Attempt to New Health Management Styles of Residents).

References

Junwei Bao, Duyu Tang, Nan Duan, Zhao Yan, Yuanhua Lv, Ming Zhou, and Tiejun Zhao. 2018. Table-to-text: Describing table region with natural language. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *Computer Vision – ECCV 2020*, pages 213–229, Cham. Springer International Publishing.

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020. Logical natural language generation from open-domain tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7929–7942, Online. Association for Computational Linguistics.

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

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie M. Strassel, and Ralph M. Weischedel. 2004. The automatic content extraction (ace) program tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC-2004)*.

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

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

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

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

Jungo Kasai, Nikolaos Pappas, Hao Peng, J. Cross, and Noah A. Smith. 2020. Deep encoder, shallow de-

coder: Reevaluating the speed-quality tradeoff in machine translation. *ArXiv*, abs/2006.10369.

H. W. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

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

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

Jian Liu, Yufeng Chen, and Jinan Xu. 2021. 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*, pages 2716–2725, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. Cite arxiv:1711.05101Comment: Published as a conference paper at ICLR 2019.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online. Association for Computational Linguistics.

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022. Prompt for extraction? PAIE: Prompting argument interaction for event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6759–6774, Dublin, Ireland. Association for Computational Linguistics.

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: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Sampo Pyysalo, Tomoko Ohta, Makoto Miwa, Han-Cheol Cho, Jun'ichi Tsujii, and Sophia Ananiadou. 2012. Event extraction across multiple levels of biological organization. *Bioinformatics*, 28(18):i575–i581.

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

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5937–5947, Online. Association for Computational Linguistics.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

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 and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4672–4682, Online. Association for Computational Linguistics.

Xueqing Wu, Jiacheng Zhang, and Hang Li. 2022. Text-to-table: A new way of information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2518–2533, Dublin, Ireland. Association for Computational Linguistics.

Runxin Xu, Peiyi Wang, Tianyu Liu, Shuang Zeng, Baobao Chang, and Zhifang Sui. 2022. A two-stream AMR-enhanced model for document-level event argument extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5025–5036, Seattle, United States. Association for Computational Linguistics.

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 ACL 2018, System Demonstrations*, pages 50–55, Melbourne, Australia. Association for Computational Linguistics.

Hang Yang, Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, and Taifeng Wang. 2021. Document-level event extraction via parallel prediction networks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6298–6308, Online. Association for Computational Linguistics.

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

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 and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 337–346, Hong Kong, China. Association for Computational Linguistics.

A Profile of Datasets

ACE05 (Doddington et al., 2004)[2] is an annotated information extraction corpus of newswire, broadcast news and telephone conversations. We utilize its English event annotation for sentence-level EAE. We preprocess the data in the same way as Wadden et al. (2019) do.

RAMS (Ebner et al., 2020)[3] is a document-level EAE dataset, which contains 9,124 annotated events from English online news. Since it is annotated event-wise (each event occupies one instance), we have to aggregate events occurring in the same context into one instance with multiple events. We follow the original train/dev/test data split.

WikiEvents (Li et al., 2021)[4] is a document-level EAE dataset, consisting of events recorded in English Wikipedia along with the linking news articles that mention these events. The dataset is also annotated with the co-reference links of arguments, but we only use the exact argument annotations in our experiments.

MLEE (Pyysalo et al., 2012)[5] is a document-level event extraction dataset with manually annotated abstracts of bio-medical publications written in English. We follow the preprocessing procedure of (Trieu et al., 2020). Since there is only train/test data split for the preprocessed dataset, we employ the training set as the development set.

Statistics Detailed statistics of the datasets are listed in Table 5.

B hyperparameter Settings

Most of the hyperparameters follow the same configuration of (Ma et al., 2022). We only tune a few hyperparameters manually for each dataset by trying different values of each hyperparameter within an interval and choosing the value that results in the highest Arg-C F1 on the development set. The trial-intervals and the final hyperparameter configuration are shown in Table 6.

C Number of Encoder/Decoder Layers

We have employed the bottom layers of RoBERTa-large as our encoder and the top layers of RoBERTa-large as our decoder. To find the optimal layer allocation, we have tried different settings and recorded the corresponding model performance. This experiment is conduct on ACE and MLEE. The results are plotted in Figure 6. We can observe that the overall performance on the two datasets reaches

[2]https://catalog.ldc.upenn.edu/LDC2006T06

[3]https://nlp.jhu.edu/rams/

[4]https://github.com/raspberryice/gen-arg

[5]http://www.nactem.ac.uk/MLEE/

Dataset	ACE05	RAMS	WikiEvents	MLEE
# Event types	33	139	50	23
# Args per event	1.19	2.33	1.40	1.29
# Events per text	1.35	1.25	1.78	3.32
# Events				
Train	4202	7329	3241	4442
Dev	450	924	345	-
Test	403	871	365	2200

Table 5: Dataset Statistics.

hyperparameters	Trial-Interval	ACE05	RAMS	WikiEvents	MLEE
Training Steps	-	10000	10000	10000	10000
Warmup Ratio	-	0.1	0.1	0.1	0.1
Learning Rate	-	2e-5	2e-5	2e-5	2e-5
Max Gradient Norm	-	5	5	5	5
Batch Size	[2, 16]	8	4	4	4
Context Window Size	-	250	250	250	250
Max Span Length	-	10	10	10	10
Max Encoder Seq Length	-	200	500	500	500
Max Decoder Seq Length	[200, 400]	250	200	360	360

Table 6: hyperparameter settings. The hyperparameters without trial-interval are set to be the same as in Ma et al. (2022) without tuning.

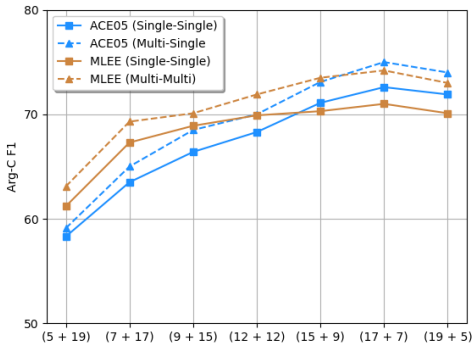


Figure 6: Comparison of TabEAE with different numbers of encoder-decoder layers. $(x + y)$ represents x-layer encoder + y-layer decoder.

the peak when there are 17 encoder layers and 7 decoder layers in the model. This observation coincides with recent findings on the areas of machine translation and spell checking that "deep encoder + shallow decoder" is superior to the conventional architecture with balanced encoder-decoder depth (Kasai et al., 2020; Sun et al., 2021).

D Prompt Construction

The prompts for ACE05, RAMS and WikiEvents are directly from (Li et al., 2021; Ma et al., 2022), which are manually constructed from the predefined ontology associated with each dataset. For MLEE, we manually construct the prompts in a similar manner, as shown in Table 7.

Event Type	Natural Language Prompt
Cell proliferation	Cell proliferate or accumulate
Development	Anatomical Entity develop or form
Blood vessel development	neovascularization or angiogenesis at Anatomical Location
Growth	growth of Anatomical Entity
Death	death of Anatomical Entity
Breakdown	Anatomical Entity degraded or damaged
Remodeling	Tissue remodeling or changes
Synthesis	synthesis of Drug/Compound
Gene expression	expression of Gene and Gene (and Gene)
Transcription	transcription of Gene
Protein processing	processing of Gene product
DNA methylation	methylation of Entity at Site
Metabolism	metabolism of Entity
Catabolism	catabolism of Entity
Phosphorylation	phosphorylation of Entity at Site
Dephosphorylation	dephosphorylation of Entity at Site
Pathway	Entity and Entity and Entity (and Entity) participate in signaling pathway or system
Localization	Entity At Location or To Location or From Location
Binding	Site of Entity bind or interact with Site of Entity (and Site of Entity)
Regulation	Something regulate Event/Entity at Site
Positive regulation	Something positively regulate Event/Entity at Site
Negative regulation	Something negatively regulate Event/Entity at Site
Planned process	Something is treated with Entity and Entity (and Entity)

Table 7: Prompts manually constructed for the MLEE dataset.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 7.

☒ A2. Did you discuss any potential risks of your work?
No potential risk is forseen.

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

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

B ☑ Did you use or create scientific artifacts?

Section 1, 3, 4.1, 4.2, Software Supplement.

☑ B1. Did you cite the creators of artifacts you used?
Section 1, 3, 4.1, 4.2.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
README (Software Supplement).

☑ 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)?
README (Software Supplement).

☒ 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?
The datasets that we used are from official and trusted sources.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 4.2, Appendix A.

☑ 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.
Appendix A.

C ☑ Did you run computational experiments?

Section 4, 5.

☒ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
The backbone of our model is RoBERTa, which is commonly used and quite familiar to NLP researchers.

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?
Section 4.1, 4.2, Appendix B.

☑ 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?
Section 4.2.

☑ 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.)?
Section 4.1, Software Supplement.

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.