Boosting Event Extraction with Denoised Structure-to-Text Augmentation

Bo Wang[1,2,3] **, Heyan Huang**[1,2,3*]**, Xiaochi Wei** [5] **, Ge Shi**[4]**, Xiao Liu**[1,2,3]**,**
Chong Feng [1,2,3] **, Tong Zhou** [4] **, Shuaiqiang Wang**[5] **, Dawei Yin**[5]
[1]School of Computer Science and Technology, Beijing Institute of Technology
[2]Key Lab of IIP&IS, Ministry of Industry and Information Technology, China
[3]Southeast Academy of Information Technology, Beijing Institute of Technology
[4]Faculty of Information Technology, Beijing University of Technology [5]Baidu Inc.
`{bwang,hhy63}@bit.edu.cn`

Abstract

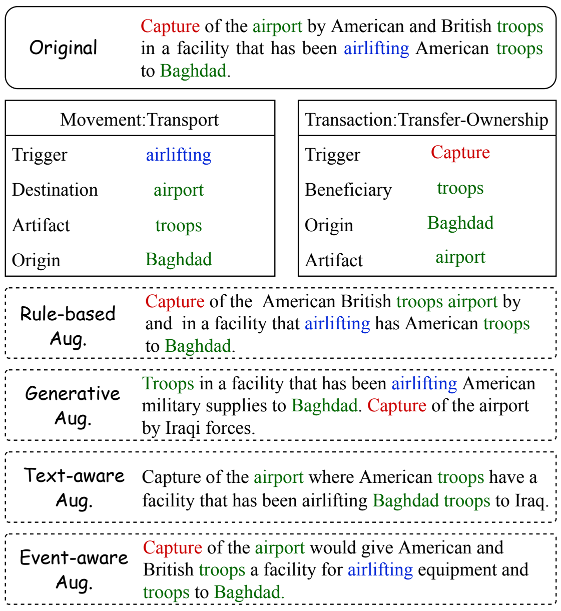
Event extraction aims to recognize pre-defined event triggers and arguments from texts, which suffer from the lack of high-quality annotations. In most NLP applications, involving a large scale of synthetic training data is a practical and effective approach to alleviate the problem of data scarcity. However, when applying to the task of event extraction, recent data augmentation methods often neglect the problem of grammatical incorrectness, structure misalignment, and semantic drifting, leading to unsatisfactory performances. In order to solve these problems, we propose a denoised structure-to-text augmentation framework for event extraction (DAEE), which generates additional training data through the knowledge-based structure-to-text generation model and selects the effective subset from the generated data iteratively with a deep reinforcement learning agent. Experimental results on several datasets demonstrate that the proposed method generates more diverse text representations for event extraction and achieves comparable results with the state-of-the-art.

1 Introduction

Event extraction is an essential yet challenging task for natural language understanding. Given a piece of text, event extraction systems discover the event mentions and then recognize event triggers and their event arguments according to pre-defined event schema (Doddington et al., 2004; Ahn, 2006). As shown in Figure 1, the sentence "*Capture of the airport by American and British troops in a facility that has been airlifting American troops to Baghdad.*" contains two events, a `Movement:Transport` event triggered by "*airlifting*" and a `Transaction:Transfer-Ownership` event triggered by "*Capture*". In the `Movement:Transport` event, three event roles are involved, i.e., `Artifact`, `Destination`, and `Origin`, and their arguments are *troops*, *airports*, and *Baghdad*, respectively. As to the `Transaction:Transfer-Ownership` event, the event roles are `Beneficiary`, `Origin`, and `Artifact`. Accordingly, the arguments are *troops*, *Baghdad*, and *airports*.

Traditional event extraction methods regard the task as a trigger classification sub-task and several arguments classification sub-tasks (Du and Cardie, 2020; Liu et al., 2020; Lin et al., 2020; Zhang and Ji, 2021; Nguyen et al., 2021, 2022a,b), while some of the recent research casting the task as a sequence generation problem (Paolini et al., 2021; Li et al., 2021; Hsu et al., 2022; Huang et al., 2023). Compared with classification-based methods, the latter line is more data-efficient and flexible. Whereas, the data containing event records are scarce, and the performance is influenced by the amount of data as the results shown in Hsu et al. (2022).

As constructing large-scale labeled data is of great challenge, data augmentation plays an important role here to alleviate the data deficient prob-



Original: Capture of the airport by American and British troops in a facility that has been airlifting American troops to Baghdad.

Movement:Transport	
Trigger	airlifting
Destination	airport
Artifact	troops
Origin	Baghdad

Transaction:Transfer-Ownership	
Trigger	Capture
Beneficiary	troops
Origin	Baghdad
Artifact	airport

Rule-based Aug.: Capture of the American British troops airport by and in a facility that airlifting has American troops to Baghdad.

Generative Aug.: Troops in a facility that has been airlifting American military supplies to Baghdad. Capture of the airport by Iraqi forces.

Text-aware Aug.: Capture of the airport where American troops have a facility that has been airlifting Baghdad troops to Iraq.

Event-aware Aug.: Capture of the airport would give American and British troops a facility for airlifting equipment and troops to Baghdad.

Figure 1: Example of text data augmentation methods.

[*]Corresponding author.

lem. There are three main augmentation methods, i.e., Rule-based augmentation method (Wei and Zou, 2019b; Dai and Adel, 2020), generative method (Wu et al., 2019; Kumar et al., 2020; Anaby-Tavor et al., 2020; Wei and Zou, 2019a; Ng et al., 2020), and text-aware method (Ding et al., 2020). However, they have different drawbacks. 1) **Grammatical Incorrectness.** Rule-based methods expand the original training data using automatic heuristic rules, such as randomly synonyms replacement, which effectively creates new training instances. As the example of *Rule-based Aug* illustrated in Figure 1, these processes may distort the text, making the generated syntactic data grammatically incorrect. 2) **Structure Misalignment.** Triggers and arguments are key components of event records, whether for both the original one and the augmented one. Nonetheless, triggers and arguments may not always exist in previous augmentation methods. As the example of *Generative Aug* illustrated in Figure 1, even though the meaning of the generated augmented sentence is quite similar to the original one, the important argument "*airport*" is missing. This may mislead the model to weaken the recognition of the DESTINATION role. 3) **Semantic Drifting.** Another important aspect of data augmentation is semantic alignment. The generated text needs to express the original event content without semantic drifting. However, this problem is commonly met in the *Text-aware Aug* method. As the example illustrated in Figure 1, the sentence completely contains all the triggers and arguments. But instead of *Baghdad*, *Iraq* is regarded as the ORIGIN in generated sentences, which may confuse the model to recognize the correct ORIGIN role.

In order to solve the aforementioned problem when applying data augmentation to event extraction, we proposed a denoised structure-to-text augmentation framework for event extraction (DAEE). For structure misalignment problems, a knowledge-based structure-to-text generation model is proposed. It is equipped with an additional argument-aware loss to generate augmentation samples that exhibit features of the target event. For the **Semantic Drift** problem, we designed a deep reinforcement learning (RL) agent. It distinguishes whether the generated text expresses the corresponding event based on the performance variation of the event extraction model. At the same time, the agent further guides the generative model to pay more attention to the samples with the **Structure Misalignment** and **Grammatical Incorrectness** problems and thus affords the *Event-aware Aug* text that both contain important elements and represent appropriate semantics. Intuitively, our agent is able to select effective samples from the combination of generated text and its event information to maximize the reward based on the event extraction model.

The key contributions of this paper are threefold:

- We proposed a denoised structure-to-text augmentation framework. It utilizes an RL agent to select the most effective subset from the augmented data to enhance the quality of the generated data.
- Under the proposed framework, a knowledge-based structure-to-text generation model is proposed to satisfy the event extraction task, which generates high-quality training data containing corresponding triggers and arguments.
- Experimental results on widely used benchmark datasets prove that the proposed method achieves superior performance over state-of-the-art event extraction methods on one dataset and comparable results on the other datasets.

2 Related Work

2.1 Event Extraction

Many existing methods use classification-based models to extract events (Nguyen et al., 2016; Wang et al., 2019; Yang et al., 2019; Wadden et al., 2019; Liu et al., 2018). And some global features are introduced to make an enhancement for joint inference (Lin et al., 2020; Li et al., 2013; Yang and Mitchell, 2016). With the large-scale use of PLMs, some of the researchers dedicated to developing generative capabilities for PLMs in event extraction, i.e., transforming into translation tasks (Paolini et al., 2021), generating with constrained decoding methods (Lu et al., 2021), and template-based conditional generation (Li et al., 2021; Hsu et al., 2022; Liu et al., 2022; Du et al., 2022). Compare with the above method directly uses a limited number of the training set, we use a denoised structure-to-text augmentation method to alleviate the problem of insufficient data.

2.2 Data Augmentation

Rather than starting from an existing example and modifying it, some model-based data augmentation approaches directly estimate a generative process produce new synthetic data by masking randomly chosen words from the training set and sample from it (Anaby-Tavor et al., 2020; Hou et al., 2018; Xia et al., 2019; Wu et al., 2019; Kumar et al., 2020). Other research design prompt (Wang et al., 2022, 2021) or use conditional generation (Ding et al., 2020) for the data augmentation. However, the above methods are mainly applied to generation tasks or comprehension tasks with simpler goals, such as text classification. When faced with complex structured extraction tasks, post-processing screening becomes a cumbersome problem. Inspired by RL, we use a policy model to automatically sift through the generated data for valid and semantically consistent samples.

3 Method

In this paper, we focus on generating the additional training set from structured event records for augmentation. Previous augmentation methods usually have **Structure Misalignment** and **Grammatical Incorrectness**, and **Semantic Drifting** problems as mentioned in the introduction. Instead, we introduce a policy-based RL strategy to select intact augmentation sentences.

3.1 Task Definition

In the generation-based event extraction task, the extraction process is divided into several subtasks according to event types $\mathcal{E}$. For each event type $e \in \mathcal{E}$, the purpose of the event extraction model is to generate $\mathcal{Y}_e$ according to the predefined prompt $\mathcal{P}_e$ and context $\mathcal{C}$, where $\mathcal{Y}_e$ is the answered prompts containing extracted event records. Except for the original data $\mathbb{T}_o$, we use a policy model as RL agent to select the effective subset $\mathbb{P}_i$ from the generated data $\mathbb{G}_i$ in the i-th epoch, thus improving the data efficiency by filtering the generated samples.

3.2 Framework

Our proposed denoised structure-to-text augmentation framework is mainly composed of the event extraction model, structure-to-text generation model, and policy model. As the policy-based RL process shown in Figure 2, the event record is first fed into the structure-to-text generation model to obtain the additional training data. Then they are filtered according to the action selected by the policy-based agent. Thus, we obtain the denoised augmentation training data for event extraction model. We use the filtered training data to retrain the event extraction model and the enhancement of the F1 score is regarded as a reward to retrain the policy model. The guidance of the event extraction model further helps the policy model select efficient samples. Finally, the generation model is retrained according to the weighted training data, and the weight is the removing action probability calculated by the retrained policy model. The retraining captain the generation model produces superior-quality sentence and consequently help the other components. The components of our proposed method will be described in the following.

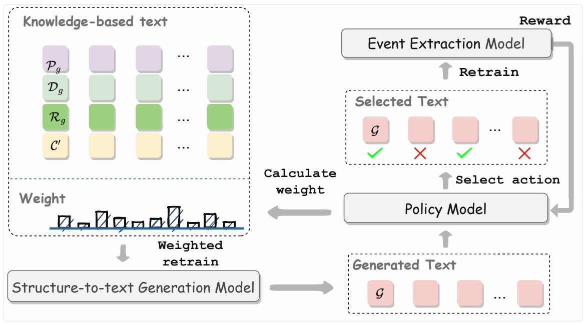


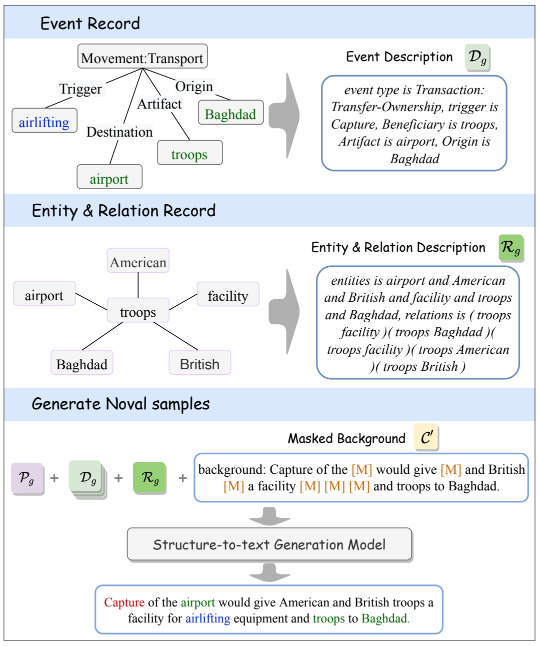
Figure 2: The proposed policy-based RL framework.

3.3 Reinforcement Learning components

The definitions of the fundamental components are introduced in the following. The **States** include the information from the current sentence and the corresponding golden event records. These two parts are both converted to the sentence vector through PLMs for the decision of action. We update states after re-generate the text guided by the previous action probability. At each iteration, the **Actions** decided by the policy model is whether to remove or retain the generated instance according to whether the sentences generated do express the corresponding event records. We use the enhancement of the F1 score as the **Rewards** for the actions decided by the policy model. Specifically, the F1 score of argument classification F_i at i-th epoch on the development set is adopted as the performance evaluation criterion. Thus, the reward $\mathcal{R}_i$ can be formulated as the difference between the adjacent epochs:

$$\mathcal{R}_i = \alpha(F_i - F_{i-1}), \tag{1}$$

where α is a scaling factor to convert the reward into a numeric result for RL agent.



Event Record
Movement:Transport
Trigger
Origin
Artifact
airlifting
Baghdad
Destination
troops
airport
Event Description $\mathcal{D}_g$
event type is Transaction: Transfer-Ownership, trigger is Capture, Beneficiary is troops, Artifact is airport, Origin is Baghdad
Entity & Relation Record
Entity & Relation Description $\mathcal{R}_g$
American
airport
facility
troops
Baghdad
British
entities is airport and American and British and facility and troops and Baghdad, relations is (troops facility)(troops Baghdad)(troops facility)(troops American)(troops British)
Generate Noval samples
Masked Background $\mathcal{C}'$
$\mathcal{P}_g$ + $\mathcal{D}_g$ + $\mathcal{R}_g$ +
background: Capture of the [M] would give [M] and British [M] a facility [M] [M] [M] and troops to Baghdad.
Structure-to-text Generation Model
Capture of the airport would give American and British troops a facility for airlifting equipment and troops to Baghdad.

Figure 3: Example of structured information representations and structure-to-text generation.

3.3.1 Event Extraction Model

We use the generation-based method GTEE-BASE (Liu et al., 2022) with the trained irrelevance classifiers as the event extraction model. The event extraction model is based on BART (Lewis et al., 2020), the entire probability $p(\mathcal{Y}_e \mid \mathcal{X}_e)$ is calculated through formulated input $\mathcal{X}_e = [\mathcal{P}_e; \texttt{[SEP]}; \mathcal{C}]$, where [;] denotes the sequence concatenation operation, and [SEP] is the corresponding separate marker. Following (Li et al., 2021) to reuse the predefined argument templates, the prompt $\mathcal{P}_e$ contains the type instruction and the template, and the event records are parsed by template matching and slot mapping according to their own event description template.

3.3.2 Structure-to-text Generation Model

As to the structure-to-text generation model, T5 (Raffel et al., 2020) is used because of its outstanding generation performance. Similar to its original setting, we define the task as a sequence transformation task by adding the prefix "*translate knowledge into sentence*" at the beginning as $\mathcal{P}_g$ to guide the generation model. It is difficult to directly generate text from structured event records with limited training data, so we randomly mask the original sentence with the special token [M] to produce the masked sentence $\mathcal{C}'$, and the mask rate is λ. $\mathcal{C}'$ is used as the background in the input of the generation model $\mathcal{X}_g$. As shown in Figure 3, the structured information annotated in the training set is transformed into event description $\mathcal{D}_g$ and relation description $\mathcal{R}_g$, respectively. They are further used as background knowledge to assist in the structure-to-text generation and the original sentence $\mathcal{C}$ is regarded as the generation target $\mathcal{Y}_g$. Given the previously generated tokens $y_{<s}$ and the input $\mathcal{X}_g$. It is notable that the entire probability $p(\mathcal{Y}_g \mid \mathcal{X}_g)$ is calculated as:

$$p(\mathcal{Y}_g \mid \mathcal{X}_g) = \prod_{s=1}^{|\mathcal{Y}_g|} p\left(y_s \mid y_{<s}, \mathcal{X}_g\right). \qquad (2)$$
$$\mathcal{X}_g = [\mathcal{P}_g; \mathcal{D}_g; \mathcal{R}_g; \mathcal{C}']$$

In addition, an argument-aware loss $\mathcal{L}_a$ is added to enforce the model to help the model to pay more attention to the event arguments during the generation process. For all event arguments that have not been generated, we search for text spans in the generated text most similar to the remaining event arguments. Detailly, we aggregate the triggers and arguments which not included in the generated text. These triggers and arguments are transformed into a one-hot embedding set $\mathbb{A}$ and each element is denoted as $a_m \in A$ denote. And the probability of selecting the token at each position in the generation model is extracted for matching the optimal-related position. By setting the window size to the number of words in a_m, we divide the probability sequence into pieces using the sliding window and obtain all the candidate set $\mathbb{K}_m$ for each a_m in $\mathbb{A}$. We first calculate the $L1$ distance between a_m and each element in $\mathbb{K}_m$ as the distance score between them. Then, all distance scores are mixed together in the back of completely traversing $\mathbb{A}$. in the case of avoiding the conflict of matching positions, greedy search is finally utilized to check each element in $\mathbb{A}$ to the position with the lowest distance score. Together with the original language model loss function $\mathcal{L}_{lm}$, the loss function of the generation model $\mathcal{L}_g$ is defined as:

$$\mathcal{L}_{lm} = \sum_{s=1}^{|\mathcal{Y}_g|} y_s log\, p(y_s \mid y_{<s}, \mathcal{X}_g)$$
$$\mathcal{L}_a = \sum_{t=1}^{\mathrm{T}} \sum_{k=k_t}^{k_t'} y_k log\, p(y_k \mid y_{<k}, \mathcal{X}_g) \qquad (3)$$
$$\mathcal{L}_g = -\frac{1}{\mathrm{N}} \sum_{n=1}^{\mathrm{N}} (\beta \mathcal{L}_{lm} + \gamma \mathcal{L}_a)$$

where N is the number of instances, T is the number of elements contained in the current unmatched set,

k_t and k'_t denote the start and end position of t-th unmatched element in the original sentence, and y_k is the k-th corresponding trigger or argument word.

3.3.3 Policy Model

For each input sentence, our policy model is required to determine whether it expresses the target event records. Thus, the policy model makes a removal action if it is irrelevant to the target event records and it is analogous to a binary classifier. For each generated sentence $\mathcal{G} \in \mathbb{G}_i$, the input of the policy model $\mathcal{X}_p$ consists of $\mathcal{G}$ and corresponding event description $\mathcal{D}_g$. The symbolic representation of input is formulated as $\mathcal{X}_p = [\mathcal{D}_g; \texttt{[SEP]}; \mathcal{G}]$ with the separate marker [SEP]. We fine-tune the BERT model by feeding the [CLS] vector into the MLP layer. And then a softmax function is utilized to calculate the decision probability for retaining the sample $\mathcal{G}$. A binary cross-entropy loss function is introduced for this classifier,

$$\mathcal{L}_p = -\frac{1}{\mathrm{N}} \sum_{n=1}^{\mathrm{N}} y_n \log p(y_n \mid \mathcal{X}_p), \tag{4}$$

where y_n is the golden action for n-th sample, and N is the number of instances.

3.4 Training Strategy

3.4.1 Pre-training

The three components, i.e., event extraction model, structure-to-text generation model, and policy model, are pre-trained with different strategies. Since the policy model has no task-specific information at the very beginning, the generation model is trained for several epochs at first to establish the training set for the policy model. We stop training the generation model until more than 70% of the trigger and arguments could be generated. The generated sentences containing their corresponding triggers and arguments are considered positive samples for the policy model, while the others are treated as negative samples. To get a balance between positive and negative samples, we randomly select some event descriptions and sentences irrelevant to the event descriptions as negative samples as well. We early stop training the policy model when the precision reaches $80\% \sim 90\%$. This can preserve the information entropy of the result predicted by the policy model, and extend the exploration space. Then we continue to pre-train the generation model and the event extraction model with the original training set for fixed epochs. These two pre-trained models are used as our initialized generation model and extraction model in the retraining process, respectively.

3.4.2 Retraining with Rewards

For i-th epoch in retraining the agent, the policy model selects actions for each element in generated dataset $\mathbb{G}_i$. According to the actions, $\mathbb{G}_i$ is divided into negative samples $\mathbb{N}_i$ and positive samples set $\mathbb{P}_i$. Then we sample a subset from the original training data, and $\mathbb{T}_o$ is mixed with $\mathbb{P}_i$ as the reconstructed training set $\mathbb{T}_i$ and used to retrain the event extraction model. Except for the improvement of argument F1 score, the growth on trigger F1 is also beneficial for the model. Therefore, we updated the checkpoint while either the trigger or argument F1 score improved to avoid falling into a local optimum. Following (Qin et al., 2018), we employ two sets for training the policy model,

$$\begin{aligned} \mathbb{D}_{i-1} &= \mathbb{N}_{i-1} - (\mathbb{N}_{i-1} \cap \mathbb{N}_i) \\ \mathbb{D}_i &= \mathbb{N}_i - (\mathbb{N}_{i-1} \cap \mathbb{N}_i) \end{aligned} . \tag{5}$$

Since we can't explore all directions to get the maximum reward for a single step, we select a constant number of samples from $\mathbb{D}_{i-1}$ and $\mathbb{D}_i$ for training, respectively, named $\mathbb{D}'_{i-1}$ and $\mathbb{D}'_i$. Referring to Equation (6), the retraining loss function of our policy model $\mathcal{L}'_p$ is defined as:

$$\begin{aligned} \mathcal{L}'_p = &\sum^{\mathbb{D}'_i} y_n \log p(y_n \mid \mathcal{X}_p) \mathcal{R}_i + \\ &\sum^{\mathbb{D}'_{i-1}} y_n \log p(y_n \mid \mathcal{X}_p)(-\mathcal{R}_i). \end{aligned} \tag{6}$$

The probability of being considered an invalid sample is taken as the weight for retraining the corresponding instance in the generation model. So we use the probability of removing the sample $w_n = 1 - \log p(y_n \mid \mathcal{X}_p)$ as the sample weight and retrain the generation model with the following retraining loss function $\mathcal{L}'_g$ referring to Equation (3):

$$\mathcal{L}'_g = -\frac{1}{\mathrm{N}} \sum_{n=1}^{\mathrm{N}} (\beta w_n \mathcal{L}^n_{lm} + \gamma w_n \mathcal{L}^n_a) \tag{7}$$

where $\mathcal{L}^n_{lm}$ and $\mathcal{L}^n_a$ are the language model loss and argument-aware loss for n-th sample, respectively. The detail of the retraining algorithm is shown in Appendix A.

Model	Trg-C			Arg-C		
	P	R	F1	P	R	F1
OneIE	$\underline{72.1}$	73.6	72.8	$\underline{55.4}$	54.3	54.8
Text2Event	71.2	72.5	71.8	54.0	$\underline{54.8}$	54.4
DEGREE-e2e	-	-	72.7	-	-	$\underline{55.0}$
GTEE-dynpref	67.3	**83.0**	$\underline{74.3}$	49.8	**60.7**	54.7
DAEE	**78.8**$^{\pm 0.4}$	$\underline{75.1}^{\pm 5.0}$	**76.9**$^{\pm 0.4}$	**58.5**$^{\pm 1.5}$	$54.4^{\pm 0.4}$	**56.3**$^{\pm 0.2}$

Table 1: Results on ACE05-E^{+}. We reported the average result of eight runs with different random seeds, our results are like "$a^{\pm b}$", where "a" and "b" represents the mean and the variance, respectively. We bold the highest scores and underline the second highest scores.

Model	Trg-C			Arg-C		
	P	R	F1	P	R	F1
OneIE	58.4	59.9	59.1	$\underline{51.8}$	$\underline{49.2}$	50.5
Text2Event	59.2	59.6	59.4	49.4	47.2	48.3
DEGREE-e2e	-	-	57.1	-	-	49.6
GTEE-dynpref	$\underline{61.9}$	**72.8**	**66.9**	51.9	**58.8**	**55.1**
DAEE	**68.7**$^{\pm 0.8}$	$\underline{61.6}^{\pm 0.5}$	$\underline{65.0}^{\pm 0.4}$	**57.7**$^{\pm 0.8}$	$46.7^{\pm 0.4}$	$\underline{51.6}^{\pm 0.3}$

Table 2: Results on ERE-EN.

4 Experiments

4.1 Experimental Settings

4.1.1 Datasets and Evaluation Metrics

Following the previous work (Zhang et al., 2019; Wadden et al., 2019; Du and Cardie, 2020; Lu et al., 2021; Hsu et al., 2021; Liu et al., 2022), We preprocess the two widely used English event extraction benchmarks, ACE 2005 (LDC2006T06) and ERE (LDC2015E29, LDC2015E68, and LDC2015E78) into ACE05-E and ERE-EN. ACE 2005 is further preprocessed into ACE05-E^{+} following (Lin et al., 2020). Statistics of the datasets are further shown in Appendix B.1.

Following previous work (Zhang et al., 2019; Wadden et al., 2019), we use precision (P), recall (R), and F1 scores to evaluate the performance. More specifically, we report the performance on both trigger classification (**Trig-C**) and argument classification (**Arg-C**). In the task of trigger classification, if the event type and the offset of the trigger are both correctly identified, the sample is denoted as correct. Similarly, correct argument classification means correctly identifying the event type, the role type, and the offset of the argument. Following (Lu et al., 2021; Liu et al., 2022), the offset of extracted triggers is decoded by string matching in the input context one by one. For the predicted argument, the nearest matched string is used as the predicted trigger for offset comparison.

Model	Trg-C			Arg-C		
	P	R	F1	P	R	F1
DyGIE++	-	-	69.7	-	-	48.8
GAIL	$\underline{74.8}$	69.4	72.0	**61.6**	45.7	52.4
OneIE	-	-	$\underline{74.7}$	-	-	**56.8**
BERT_QA	71.1	73.7	72.3	$\underline{56.8}$	50.2	53.3
MQAEE	-	-	71.7	-	-	53.4
TANL	-	-	68.5	-	-	48.5
BART-Gen	69.5	72.8	71.1	56.0	51.6	53.7
Text2Event	67.5	71.2	69.2	46.7	53.4	49.8
DEGREE-e2e	-	-	70.9	-	-	54.4
GTEE-dynpref	63.7	**84.4**	72.6	49.0	**64.8**	55.8
DAEE	**75.1**$^{\pm 1.7}$	$\underline{76.6}^{\pm 4.1}$	**75.8**$^{\pm 0.6}$	$55.9^{\pm 3.6}$	$\underline{57.2}^{\pm 1.8}$	$\underline{56.5}^{\pm 0.3}$

Table 3: Results on ACE05-E. The first group is the classification-based methods and the second group is the generation-based methods.

4.1.2 Baselines

We illustrate the event extraction results between our proposed DAEE and the baselines conducted in two categories, i.e., classification-based models and generation-based models.

The first category is **classification-based models**, DyGIE++ (Wadden et al., 2019): a joint model with contextualized span representations. GAIL (Zhang et al., 2019): an RL model jointly extracting entity and event. OneIE (Lin et al., 2020): a joint neural model for information extraction task with several global features and beam search. BERT_QA (Du and Cardie, 2020): a method using separated question-answering pairs for event extraction. MQAEE (Li et al., 2020): a question answering system with multi-turn asking.

The other category is **generation-based methods**, and our proposed DAEE belongs to this one. TANL (Paolini et al., 2021): a method that use translation tasks modeling event extraction in a trigger-argument pipeline. BART-Gen (Li et al., 2021): a document-level event extraction method through conditional generation. Text2Event (Lu et al., 2021): a method directly generates structure from the text. DEGREE-e2e (Hsu et al., 2022): a method using discrete prompts and end-to-end conditional generation to extract event. GTEE-dynpref (Liu et al., 2022): a generative template-based event extraction method using dynamic prefix-tuning.

4.2 Results and Analysis

4.2.1 Main Results

The performance comparison on dataset ACE05-E^{+} is shown in Table 1. It can be observed that DAEE achieves the SOTA F1 score on ACE05-E^{+} and obtain 1.1% and 0.7% gain of F1 scores for**Trg-C** and **Arg-C**, respectively. The improvement indicates

Model	Trg-C			Arg-C		
	P	R	F1	P	R	F1
DAEE	78.8	75.1	76.9	58.5	54.4	**56.3**
w/o AL	78.0	**75.5**	76.7	56.2	**55.6**	55.9
w/o RG	78.8	75.5	**77.1**	56.2	54.9	55.5
w/o RL	**79.0**	71.9	75.3	**56.3**	54.3	55.3

Table 4: Ablation Study on ACE05-E$^+$ for event extraction. AL denotes the argument-aware loss $\mathcal{L}_a$, RG denotes the process of retraining the generation model, and RL denotes the reinforcement learning strategy.

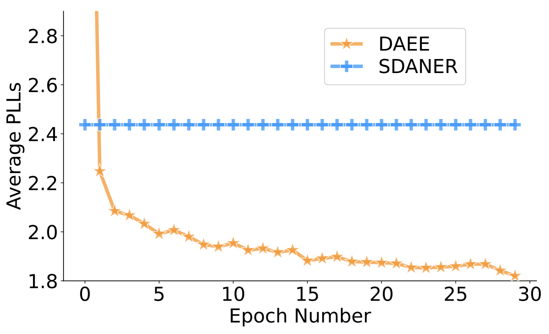


Figure 4: Results for the PLLs of DAEE and SDANER on ACE05-E$^+$.

that DAEE is able to guide the generation model to generate the text containing events and select suitable samples to improve the effectiveness of the event extraction model.

Table 2 presents the performance of baselines and DAEE on ERE-EN. The performance of DAEE decreases compared with GTEE-DYNPREF, but the performance is still higher than other methods, which may be affected that ERE-EN contains more pronoun arguments. The pronoun roles would offer less information for the generation model thus reducing the role of structured text in guiding the generation model.

Comparing the results on ACE05-E as Table 3 shows, we gain an improvement of 1.1% on **Trg-C** and a competitive F1 score on **Arg-C** with the SOTA classification-based method ONEIE, outperforming the others. This observation supports that structured information used in the knowledge-based generation model makes up for the information gap used by multi-task extraction.

4.2.2 Ablation Study

We further conducted an ablation study by removing each module at a time. The experimental results on ACE05-E$^+$ are presented in Table 4. We can see that the F1 score of **Arg-C** decreases by 0.4% and 0.8% when removing the argument-aware loss $\mathcal{L}_a$ and stopping retraining the generation model, respectively. The results indicate that the deployment of argument-aware loss and retraining strategy is conducive to the generation module in our framework. Then, we remove the RL strategy, which means that the generated samples are directly mixed with the original training samples for training the event extraction model from scratch. The F1 score of **Trg-C** and **Arg-C** decreases by 1.6% and 1.0%, respectively. This demonstrates that the RL strategy could ensure that the generated data is more suitable for downstream event extraction tasks and guide the improvement on both **Trg-C** and **Arg-C**.

4.2.3 Iterative Generation Discussion

To illustrate our framework is able to enhance the quality of generated sentences, we calculate the masked language model score *pseudo-log-likelihood scores* (PLLs)[1] following (Salazar et al., 2020) for each training epoch. The token $\boldsymbol{w}_s$ in the sentence is masked and predicted using all past and future tokens $\boldsymbol{W}_{\backslash s} := (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{s-1}, \boldsymbol{w}_{s+1}, \ldots, \boldsymbol{w}_{|\boldsymbol{W}|})$, and the PLLs for each sentence is calculated as

$$\text{PLLs}(\boldsymbol{W}) := \frac{1}{|\boldsymbol{W}|}\sum_{t=1}^{|\boldsymbol{W}|} \log P_{\text{MLM}}(\boldsymbol{w}_s \mid \boldsymbol{W}_{\backslash s}; \Theta).$$

The results for each epoch are the average of sentence scores over the entire training set as shown in Figure 4. PLLs is declining with the iterative process, which demonstrates that DAEE enhances the fluency of generated data and improves the effect of event extraction under the guidance of RL agent. Furthermore, we compare DAEE with a rule-based sequence labeling data augment method SDANER (Dai and Adel, 2020). SDANER contains four rule-based augmentation methods. Synonym replacement is selected according to its lowest average PLLs. DAEE generates sentences with lower PLLs compared with the rule-based method. The results demonstrate that DAEE generates more fluency and grammatically correct data.

4.2.4 Argument Loss Analysis

To verify the effectiveness of argument-aware loss $\mathcal{L}_a$ in reducing mismatches triggers and arguments, we alter the hyperparameter γ and explore the change of the unmatched number of arguments

[1]BERT is fine-tuned through mask language model loss using the training set for calculating PLLs.

Event type	Transaction:Transfer-Ownership
Original sentence	yes, **we** got uh **purchased** by our strategic **partner**, so um
GENERATION MODEL (w/o $\mathcal{L}_a$)	yeah , **we** bought from our **partner**, um, um
GENERATION MODEL	well , **we** **purchased** our **partner** purchased, um
DAEE	yeah, **we** got uh **purchased** by our **partner**,
Event type	Life:Die & Conflict:Attack
Original sentence	the iraqi government reports 1252 civilians have been **killed** in **the war**.
GENERATION MODEL (w/o $\mathcal{L}_a$)	the iraqi government says more than 200 **civilians** have been **killed** in this war .
GENERATION MODEL	the iraqi government **killed** **civilians** in **the war** .
DAEE	the iraqi government says more than 200 **civilians** have been **killed** **the war** .

Table 5: Efficient generated synthetic data from our proposed methods and simple generated Sentence. Text chunks in Blue and Red are the event triggers for different event type, text chunks in Green are the event arguments.

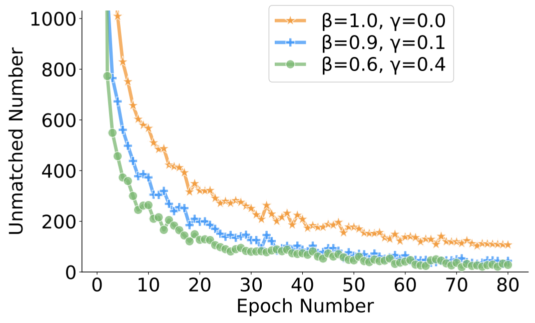


Figure 5: Unmatched arguments numbers of different training epochs.

during the training process. Three generation models are trained according to the loss function mentioned in Equation (3), and results shown in Figure 5 are observed by the change in the ratio of β and γ. Compared with setting γ to 0, the number of unmatched arguments drops rapidly under the condition of adding the $\mathcal{L}_a$ by increasing the γ. Meanwhile, the number of unmatched arguments converges around 30 after adding $\mathcal{L}_a$, while the number converges to around 120 without $\mathcal{L}_a$.

4.2.5 Diversity Analysis

Intuitively, diverse sentence description in the training set is able to enhance the model performance. We thus verify the diversity of the generated text. The degree of diversity is reported by calculating the number of distinct bigrams and trigrams in the generated text which has not appeared in the original text and the results are shown in Table 6. In the following, we use GENERATION MODEL to represent the directly trained structure-to-text generation model. Referring to the indicators proposed in (Li et al., 2016), The diversity, the argument-aware loss $\mathcal{L}_a$ helps the GENERATION MODEL to produce more diverse synthetic data, which is because the argument-aware loss makes the model focus more on retaining the triggers and arguments rather

Model	bigrams	trigrams
GENERATION MODEL	0.160	0.398
GENERATION MODEL (w/o $\mathcal{L}_a$)	0.125	0.323
DAEE	0.143	0.365

Table 6: Results of diversity analysis on ACE05-E$^+$.

than generating more similar content to the original text. The diversity is affected by the RL strategy due to the concentration on the effect of event extraction. Horizontally compared to Table 4, the experimental results demonstrate that diversified text can enable the model to obtain more information based on similar event records.

4.2.6 Synthetic Data Case Study

Table 5 shows representative examples generated by our proposed DAEE and other methods and we can see the following comparative phenomena. In the case of comparing whether to add the argument-aware loss, the GENERATION MODEL generates all the triggers and arguments in three examples, which demonstrate the generation model without $\mathcal{L}_a$ shuffles the text leaking problem. There is a misalignment in the first example for the text generated through GENERATION MODEL. The original sentence contains two roles, i.e., ARTIFACT and BUYER, and their arguments are *we* and *partner*, but the two arguments have been swapped in the synthetic text. In the second example, the *government* should play the role of AGENT in LIFE:DIE event according to the output of GENERATION MODEL, which is not appeared in the golden event record and resulting in redundancy. Neither of the above errors occurs in DAEE shown in the table, which proves the RL strategy could also be guidance for improving the effectiveness of generative models.

5 Conclusion

In this paper, we studied DAEE, the denoised structure-to-text augmentation framework for event extraction. The structure-to-text generation model with argument-aware loss is guided by the reinforcement learning agent to learn the task-specific information. Meanwhile, the reinforcement learning agent selects effective samples from generated training data that are used to reinforce the event extraction performance. Experimental results show that our model achieves competitive results with the SOTA on ACE 2005, which is also a proven and effective generative data augmentation method for complex structure extraction.

6 Limitation

This paper proposes a denoised structure-to-text augmentation framework for event extraction (DAEE), which generates and selects additional training data iteratively through RL framework. However, we still gain the following limitations.

- The framework uses reinforcement learning to select effective samples, which is a process of iterative training and predicting the generation model, policy model, and event extraction models. The iterative training framework is complicated and time-consuming compared to the standalone event extraction model.
- Even the Argument Loss decreases the number of unmatched arguments in a generated sentence, the generation model generates more fluent sentences while at the expense of the ability to ensure that all the event arguments are included completely.

7 Acknowledgement

This work was supported by the Joint Funds of the National Natural Science Foundation of China (Grant No. U19B2020). We would like to thank the anonymous reviewers for their thoughtful and constructive comments.

References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia. Association for Computational Linguistics.

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Do Not Have Enough Data? Deep Learning to the Rescue! In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 7383–7390. AAAI Press.

Xiang Dai and Heike Adel. 2020. An Analysis of Simple Data Augmentation for Named Entity Recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3861–3867, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. DAGA: Data Augmentation with a Generation Approach for Low-resource Tagging Tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6045–6057, Online. 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*. European Language Resources Association.

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*, pages 671–683, Online. Association for Computational Linguistics.

Xinya Du, Sha Li, and Heng Ji. 2022. Dynamic Global Memory for Document-level Argument Extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 5264–5275.

Yutai Hou, Yijia Liu, Wanxiang Che, and Ting Liu. 2018. Sequence-to-Sequence Data Augmentation for Dialogue Language Understanding. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1234–1245. Association for Computational Linguistics.

I-Hung Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, and Nanyun Peng. 2021. DEGREE: A Data-Efficient Generative Event Extraction Model. *CoRR*, abs/2108.12724.

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.

Heyan Huang, Xiao Liu, Ge Shi, and Qian Liu. 2023. Event extraction with dynamic prefix tuning and relevance retrieval. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–13.

Ashutosh Kumar, Kabir Ahuja, Raghuram Vadapalli, and Partha P. Talukdar. 2020. Syntax-Guided Controlled Generation of Paraphrases. *Trans. Assoc. Comput. Linguistics*, 8:330–345.

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.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020. Event Extraction as Multi-turn Question Answering. In *Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 829–838, Online. Association for Computational Linguistics.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119. The Association for Computational Linguistics.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint Event Extraction via Structured Prediction with Global Features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 73–82, Sofia, Bulgaria. 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, Yubo Chen, Kang Liu, Wei Bi, and Xiaojiang Liu. 2020. Event Extraction as Machine Reading Comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 1641–1651, Online. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022. Dynamic Prefix-Tuning for Generative Template-based Event Extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 5216–5228. Association for Computational Linguistics.

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*, pages 1247–1256, Brussels, Belgium. Association for Computational Linguistics.

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*, pages 2795–2806, Online. Association for Computational Linguistics.

Nathan Ng, Kyunghyun Cho, and Marzyeh Ghassemi. 2020. SSMBA: self-supervised manifold based data augmentation for improving out-of-domain robustness. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1268–1283. Association for Computational Linguistics.

Minh Van Nguyen, Viet Dac Lai, and Thien Huu Nguyen. 2021. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 27–38, Online. Association for Computational Linguistics.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022a. Joint extraction of entities, relations, and events via modeling inter-instance and inter-label dependencies. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4363–4374, Seattle, United States. Association for Computational Linguistics.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022b. Learning cross-task dependencies for joint extraction of entities, events, event arguments, and relations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9349–9360, Abu Dhabi, United Arab Emirates. 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.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cícero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured Prediction as Translation between Augmented Natural Languages. In *Proceedings of the 9th International Conference on Learning Representations*. OpenReview.net.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018. Robust Distant Supervision Relation Extraction via Deep Reinforcement Learning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2137–2147, Melbourne, Australia. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked Language Model Scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

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*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Xiaozhi Wang, Ziqi Wang, Xu Han, Zhiyuan Liu, Juanzi Li, Peng Li, Maosong Sun, Jie Zhou, and Xiang Ren. 2019. HMEAE: Hierarchical Modular Event Argument Extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5777–5783, Hong Kong, China. Association for Computational Linguistics.

Yufei Wang, Can Xu, Qingfeng Sun, Huang Hu, Chongyang Tao, Xiubo Geng, and Daxin Jiang. 2022. PromDA: Prompt-based Data Augmentation for Low-Resource NLU Tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 4242–4255. Association for Computational Linguistics.

Zirui Wang, Adams Wei Yu, Orhan Firat, and Yuan Cao. 2021. Towards Zero-Label Language Learning. *CoRR*, abs/2109.09193.

Jason Wei and Kai Zou. 2019a. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. 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 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Jason W. Wei and Kai Zou. 2019b. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 6381–6387. Association for Computational Linguistics.

Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Conditional BERT Contextual Augmentation. In *Proceedings of the Computational Science - ICCS 2019 - 19th International Conference*, volume 11539 of *Lecture Notes in Computer Science*, pages 84–95. Springer.

Mengzhou Xia, Xiang Kong, Antonios Anastasopoulos, and Graham Neubig. 2019. Generalized Data Augmentation for Low-Resource Translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 5786–5796. Association for Computational Linguistics.

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: Human Language Technologies*, pages 289–299, San Diego, California. Association for Computational Linguistics.

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*, pages 5284–5294, Florence, Italy. Association for Computational Linguistics.

Tongtao Zhang, Heng Ji, and Avirup Sil. 2019. Joint Entity and Event Extraction with Generative Adversarial Imitation Learning. *Data Intell.*, 1(2):99–120.

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: Human Language Technologies*, pages 39–49, Online. Association for Computational Linguistics.

Dataset	Split	#Sents	#Events	#Roles
ACE05-E	Train	17,172	4,202	4,859
	Dev	923	450	605
	Test	832	403	576
ACE05-E$^+$	Train	19,216	4,419	6,607
	Dev	901	468	759
	Test	676	424	689
ERE-EN	Train	14,736	6,208	8,924
	Dev	1,209	525	730
	Test	1,163	551	822

Table 7: Dataset statistics.

Name	EE	POLICY	GEN
learning rate (pretrain)	1e-5	1e-5	3e-5
learning rate (retrain)	1e-6	1e-6	3e-5
train batch size	32*2	32	32
epochs (pretrain)	15	-	20
epochs (retrain)	2	1	1
weight decay (pretrain)	1e-5	1e-5	1e-5
gradient clip	5.0	5.0	5.0
warm-up ratio (pretrain)	10%	-	-
optimizer	AdamW	Adam	Adam

Table 8: Hyperparameter setting for our models, EE denotes the event extraction model, POLICY denotes the policy model, GEN denotes the generation model.

A Details of Methods

The detail of the retraining algorithm is shown in Algorithm 1.

B Details of Experiments

B.1 Data Statistics

In this paper, we use the three datasets to verify our proposed method, the statistics of the datasets are shown in Table 7.

B.2 Implementation Details

All experiments were conducted with NVIDIA A100 Tensor Core GPU 40GB. For the pre-trained language model, we reuse the three English models released by Huggingface[2]. Specifically, γ and β are set to 0.1 and 0.9 in Equation (2), respectively, the RL training epoch is set to 80, the reward scale α is set to 10, the sample ratio from original event extraction training set is set to 0.5, the negative sample ratio for GTEE-BASE in training is set to 12% for event extraction, and the other hyperparameters used are shown in Table 8.

B.3 Generation Reliability Discussion

To verify the verifies the convince of the generated data, we train GTEE-BASE through the samples with event record, which is because that only the samples with event record are used for data augmentation. The results are shown in Table 9. The $F1$ score trained on DD increases by 1.1% and 2.5% compared with the results trained on OD and GD, respectively. The data generated by DAEE achieves a closer effect to original data, which thus could be utilized for training the competitive event extraction models.

Model	Trg-C			Arg-C		
	P	R	F1	P	R	F1
DD	69.3	79.7	74.1	47.6	56.5	51.7
GD	68.5	81.4	74.4	42.3	58.6	49.2
OD	66.3	80.7	72.8	43.1	61.2	50.6

Table 9: The experimental results on ACE05-E$^+$,DD denotes using the generated data though DAEE, while GD denotes the data from GENERATION MODEL without RL, DD denotes the data from the original training set.

[2]https://huggingface.co/t5-base, https://huggingface.co/bert-base-uncased, https://huggingface.co/facebook/bart-large

Algorithm 1 The process of retraining the reinforcement learning framework.

Parameter:The original event extraction training set $\mathbb{T}_o$, parameters of policy model θ_p, event extraction model θ_e, generation model θ_g, generated sentence set, n-th generated sentence $\mathcal{G}_n$, positive samples set $\mathbb{P}_i$, negative samples set $\mathbb{N}_i$

```
 1: Initialize trigger $F1$ score $F^t_{max}$ and role $F1$ score $F^a_{max}$ through $\theta_e$
 2: for epoch $i$ in $1 \to$ K do
 3:     for $\mathcal{G}_n$ in $\mathbb{G}_{i-1}$ do
 4:         Calculate $[\mathcal{D}_g; [SEP]; \mathcal{G}_n] \to \mathcal{X}_p$
 5:         Sample action according $p(y_n \mid \mathcal{X}_p, \theta_p)$
 6:         if action == 1 then
 7:             Add $\mathcal{G}_n \to \mathbb{P}_i$
 8:         else
 9:             Add $\mathcal{G}_n \to \mathbb{N}_i$
10:         end if
11:     end for
12:     Calculate $\mathbb{D}'_i$ and $\mathbb{D}'_i$ according Equation 5
13:     Sample $\mathbb{T}_{sub}$ from $\mathbb{T}_o$ and concatnate $\{\mathbb{T}_{sub}, \mathbb{P}_i\} \to \mathbb{T}_i$
14:     Retrain event extraction model through $\mathbb{T}_i$
15:     Calculate Trg-C score $F^t_i$ and Arg-C score $F^a_i$, training set for GENERATION MODEL $\mathbb{Y}_i$
16:     Calculated Reward $\alpha(F^a_i - F^a_{i-1}) \to \mathcal{R}_i$
17:     if $F^a_i > F^a_{max}$ or $F^t_i > F^a_{max}$ then
18:         Change $F^a_i \to F^t_{max}$, $F^t_i \to F^t_{max}$, and update $\theta_p$
19:     end if
20:     Retrain policy through $\mathbb{D}_i$ and $\mathbb{D}_{i-1}$ according Equation 6
21:     Update training weight $1 - \log p(\mathcal{Y}_p \mid \mathcal{X}_p) \to w_n$ for each sample in $\mathbb{Y}_g$,
22:     Retrain the generation model through weighted $\mathbb{Y}_g$ according Equation 3
23:     Update $\theta_g$ and generate $\mathbb{G}_i$
24: end for
```

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
We report it in Section 6.

☒ A2. Did you discuss any potential risks of your work?
Event extraction is a standard task in NLP and we do not see any significant ethical concerns. We evaluate the event extraction task with conventional metrics. As the extraction evaluation is our main focus, we do not anticipate the production of harmful outputs on our proposed task.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
We report it in Section 1.

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

B ☑ Did you use or create scientific artifacts?

We report it in Section 4.

☑ B1. Did you cite the creators of artifacts you used?
We report it in Section 4.

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
The first institution of our paper is the Beijing Institute of Technology, which has obtained the authorization of the LDC User Agreement for ACE 2005 and Rich-ERE data. The code (https://github.com/huggingface/trans we used is licensed under Apache License 2.0.

☒ 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)?
Our work is free for public research purposes.

☑ 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 data set used in this paper was annotated from the publicly available text when annotated by their authors.

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

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
We report it in Appendix A.

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

C ☑ Did you run computational experiments?

We report it in Section 4.

☒ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
We used a conventional pre-training model and we did not focus on model and GPU memory efficiency, so we did not report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
We report it in 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?
We report it in Section 4.2.1

☑ 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.)?
We report it in Appendix B.

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.?
Not applicable. Left blank.

☐ 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)?
Not applicable. Left blank.

☐ 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?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.