CEAN: Contrastive Event Aggregation Network with LLM-based Augmentation for Event Extraction

Zihao Meng, Tao Liu, Heng Zhang*, Kai Feng, Peng Zhao
Alibaba Group, China
{mzh267783,shengshu.lt,heng.zhangh1,fk284389,zp247381}@alibaba-inc.com

Abstract

Event Extraction is a crucial yet arduous task in natural language processing (NLP), as its performance is significantly hindered by laborious data annotation. Given this challenge, recent research has predominantly focused on two approaches: pretraining task-oriented models for event extraction and employing data augmentation techniques. These methods involve integrating external knowledge, semantic structures, or artificially generated samples using large language models (LLMs). However, their performances can be compromised due to two fundamental issues. Firstly, the alignment between the introduced knowledge and event extraction knowledge is crucial. Secondly, the introduction of data noise during the augmentation is unavoidable and can mislead the model's convergence. To address these issues, we propose a Contrastive Event Aggregation Network with LLM-based Augmentation to promote low-resource learning and reduce data noise for event extraction. Different from the existing methods introducing linguistic knowledge into data augmentation, an event aggregation network is established to introduce event knowledge into supervised learning by constructing adaptively-updated semantic representation for trigger and argument. For LLM-based augmentation, we design a new scheme including a multi-pattern rephrasing paradigm and a data-free composing paradigm. Instead of directly using augmentation samples in the supervised task, we introduce span-level contrastive learning to reduce data noise. Experiments on the ACE2005 and ERE-EN demonstrate that our proposed approach achieves new state-of-the-art results on both of the two datasets.

1 Introduction

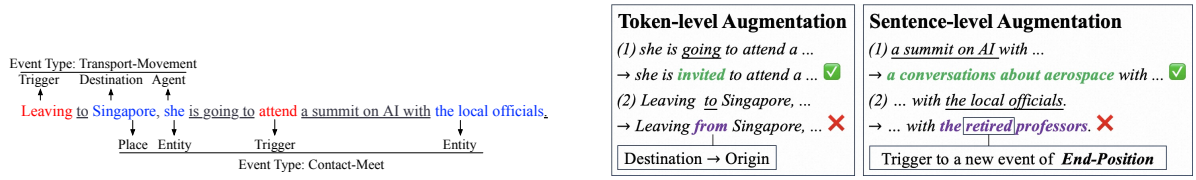
Event Extraction, as a fundamental task of information extraction, aims at acquiring structured information about periodical incidents from plain text (Li et al., 2021b; Gao et al., 2023a; Liu et al., 2022a). As shown in Figure 1a, it is usually accomplished by trigger extraction and argument extraction. Most of the methods are based on supervised learning, which require adequate high-quality labeled data. However, data annotation is usually arduously expensive, which means only insufficient data can be used to train a supervised model. In such low-resource scenarios, event extraction models often suffer from poor performances, especially when facing the hard samples with rare patterns.

To alleviate data sparsity, some methods leverage large-scale unsupervised data with pretraining tasks such as semantic structure analysis (Wang et al., 2021; Fan et al., 2022). For such methods, a crucial and challenging issue is the alignment between rich knowledge lying in unsupervised data and event knowledge. Alternatively, data augmentation methods are proposed by rephrasing the event-related fragments and adjunct fragments, which are lexically different but semantically consistent. As shown in Figure 1b, token-level augmentations are based on linguistic knowledge such as synonym replacement, which results in limited diversity improvement. Sentence-level augmentations use text generation techniques such as back translation to produce more diverse samples. With recent success on text generation, LLMs can be introduced for more flexible and diverse augmentation. Meanwhile, data noise is unavoidably expanded. Subtle lexical changes in token-level possibly result in event structure deviations. New events can be unexpectedly introduced by sentence-level augmentation without annotated. In supervised learning, model's convergence can be misled by data noise in the training data. However, few existing studies of event extraction focus on the data noise alleviation.

This paper proposes a Contrastive Event Aggregation Network with LLM-based Augmentation (CEAN). Firstly, we establish an event aggregation network with a knowledge bank by construct-

* Corresponding author

Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 321–333



(a) Event Extraction

(b) Data Augmentation

Figure 1: Examples of event extraction and data augmentation. In (a), the event-related fragments (the triggers and arguments) are marked in red and blue. The adjunct fragments are underlined. In (b), token-level and sentence-level augmentation are demonstrated where the rephrased fragments are underlined. Cleaned and noisy augmentation are marked in green and purple. By replacing the *"to"* with *"from"*, the role of *"Singapore"* deviates from *"Destination"* to *"Origin"*. By using *"retired"* in the rephrasing, a new event of *"End-Postion"* is unexpectedly introduced.

ing adaptively-updated semantic representation for trigger and argument. The knowledge bank is to store and aggregate the event-related knowledge extracted from both the original and augmented samples. The event aggregation module of the network activates the aggregated knowledge and aligns it with the event extraction task. Secondly, we propose an LLM-based augmentation method with a rephrasing paradigm by paraphrase generation and a composing paradigm relying on event schema, which enables precise control of semantic and lexical diversity metrics. Thirdly, instead of using the generated samples to train a supervised model directly, we introduce a span-level contrastive learning loss function which transforms the supervised learning process into a similarity measurement on triggers and arguments to reduce data noise. ACE2005 and ERE-EN, two benchmarks of event extraction, are used to validate our approach which reaches F1-score of 82.5%, 61.5% and 67.7%, 55.4% in trigger classification and argument role classification on the two datasets.

We summarize our contributions as belows:

- An event aggregation network is established to mitigate the data sparsity by event knowledge aggregation.
- A new LLM-based augmentation method is proposed including a multi-pattern rephrasing paradigm and a data-free composing paradigm. A span-level contrastive learning strategy is proposed to alleviate data noise.
- New state-of-the-art results are achieved on ACE2005 and ERE-EN. Each contribution is validated through the ablation study.

2 Related Work

Traditional extraction studies are based on elaborate feature engineering (Ji and Grishman, 2008; Liao and Grishman, 2010), which are replaced by deep learning-based methods with Convolutional Neural Network (Nguyen and Grishman, 2015, 2016), Recurrent Neural Network (Nguyen et al., 2016; Sha et al., 2018), Transformer (Ren et al., 2021; Li et al., 2021a; Wadden et al., 2019; Lin et al., 2020; Wang et al., 2021; Lu et al., 2021; Fan et al., 2022; Shi et al., 2023) etc. According to the procedure of trigger and argument extraction, existing studies can be classified into pipeline-based and joint-based methods, which arrange the two extraction subtasks either in a serial or parallel paradigm, respectively (Li et al., 2022, 2021b). Mostly, the extraction task can be solved in classification manner or generative manner (Li et al., 2021b). For classification manner, the typical studies employ transformer-based encoder with modules such as feed forward network (Yang et al., 2019a) or global pointer (Su et al., 2022; Zhang et al., 2023; Cao et al., 2022; Ning et al., 2023) to conduct token classification or sequence labeling. For generative manner, many approachs with transformer decoder architecture are recently proposed to solve the extraction tasks by machine reading comprehension or sequence-to-structure generation (Lu et al., 2021; Liu et al., 2022b; Shi et al., 2023). With the surge in popularity of LLMs especially ChatGPT (Ouyang et al., 2022), some research has attempted to conduct zero-shot or few-shot event extraction by harnessing the powerful capabilities of LLMs, only achieving unsatisfactory performances (Gao et al., 2023b; Wei et al., 2023; Li et al., 2023).

To tackle the data sparsity, unsupervised pretraining on tasks such as abstract meaning representation are explored (Wang et al., 2021; Fan et al., 2022). Additionally, data augmentations are conducted by external knowledge introduction (Chen et al., 2017; Liu et al., 2016; Yang et al., 2019b), mask token prediction (Yang et al., 2019a), back-translation (Xie et al., 2020), blank infilling (Gao

et al., 2023a) etc. Recenet studies prove that token-level augmentation can only bring poor diversity improvement (Gao et al., 2023a), and possibly results in event information deviation (Yang et al., 2019a). Sentence-level augmentation can better improve diversity, but also unavoidably bring data noise (Gao et al., 2023a). Recent studies introduce LLMs in sentence-level augmentation and receive performances improvement (Bonifacio et al., 2022; Whitehouse et al., 2023; Dai et al., 2023). Retrieval-based methods are also proposed for low-resource learning of information extraction (Huang et al., 2023; Chen et al., 2022).

3 Contrastive Event Aggregation Network

This section describes our proposed model CEAN for closed-domain event extraction. Detailed architecture is introduced as two parts, event aggregation network and span-level contrastive learning. The proposed approach is in pipeline-based paradigm, with only minor differences in model architecture and training methods for trigger and argument extraction. Therefore, we use the notations with a superscript of $*$ to represent the layers, tensors or functions that occur in both trigger and argument extraction. The notations marked with a superscript of T or A are specifically used for trigger extraction or argument extraction, respectively. Data-related notations are summarized in Table 1.

3.1 Event Aggregation Network

As illustrated in Figure 2, our proposed event aggregation network is comprised of an encoder for contextual representation, an event aggregation module for knowledge introduction and a global pointer module for event-related fragment extraction. The event-related fragment to extract is the trigger span and argument span in trigger and argument extraction, respectively.

3.1.1 Text Encoder for contextual representation

A transformer-based deep encoder is used for contextual embedding. Given s, the encoder transforms it into $H_s^* \in \mathbb{R}^{n\times v}$, which is shown in Eq.1.

$$H_s^* = \{h_1, ..., h_n\} = Enc^*(s) \tag{1}$$

where Enc^* is the text encoder; $h_i \in \mathbb{R}^v$ is the representation vector of the ith token and v is the hidden dimension of the encoder.

3.1.2 Event Aggregation Module for knowledge introduction

It has been proven that low-resource learning can be promoted by the introduction of external knowledge (Chen et al., 2017; Liu et al., 2016). Few of studies focus on sample-wise knowledge, which means knowledge is not built by a carefully selected knowledge base but is discovered from the interactions among samples. An event aggregation module is proposed to better represent the event knowledge in sample-wise with contextual information. It includes a span expanding matrix to represent every possible span with contextual information, a knowledge bank to derive the centroid of each event type or argument role, and an event consistency layer to evaluate the semantic distance between a span and each centroid. We first introduce the event aggregation module for trigger extraction, then the argument extraction.

Trigger Extraction Given s, a span expanding matrix $J^T \in \mathbb{R}^{n\times n\times 2v}$ is built to generate representations for all possible $\frac{1}{2}\binom{n}{2}$ spans, which means only the upper triangular of J^T is valid. For each element in J^T, $J^T[x, y]$ is defined as the text encoder representation for $s[x : y]$, which is derived by Eq 2.

$$J^T[x, y] = (h_x \oplus h_y), x \leq y \tag{2}$$

where $J^T[x, y]$ is the concatenation of h_x and h_y.

A knowledge bank $K_{EA}^T \in \mathbb{R}^{m\times 2v}$ is built for all event types as event knowledge aggregation, which is shown in Eq.3.

$$K_{EA}^T = \{K_1, ...K_m\} \tag{3}$$

where each element $K_e \in \mathbb{R}^{2v}$ denotes the representation centroid of all triggers with same event type e, derived by Eq.4.

$$K_e = \frac{1}{||T_e||}\sum_{t\in T_e}(h_{head(t)} \oplus h_{tail(t)}) \tag{4}$$

where $h_{head(t)}$ and $h_{tail(t)}$ denote the text encoder's output of the head and tail tokens of trigger t.

A consistency layer W_{EA}^T is leveraged to weight the consistency between each span and each event type, which is a linear layer sharing the same shape with K_{EA}^T in this paper. W_{EA}^T is initialized by K_{EA}^T and updated in a momentum manner, which is introduced in § 3.3. The consistency between a span $s[x : y]$ and an event type e is scored by Eq.5.

$$c^T = W_{EA}^T[e]J^T[x, y] \tag{5}$$

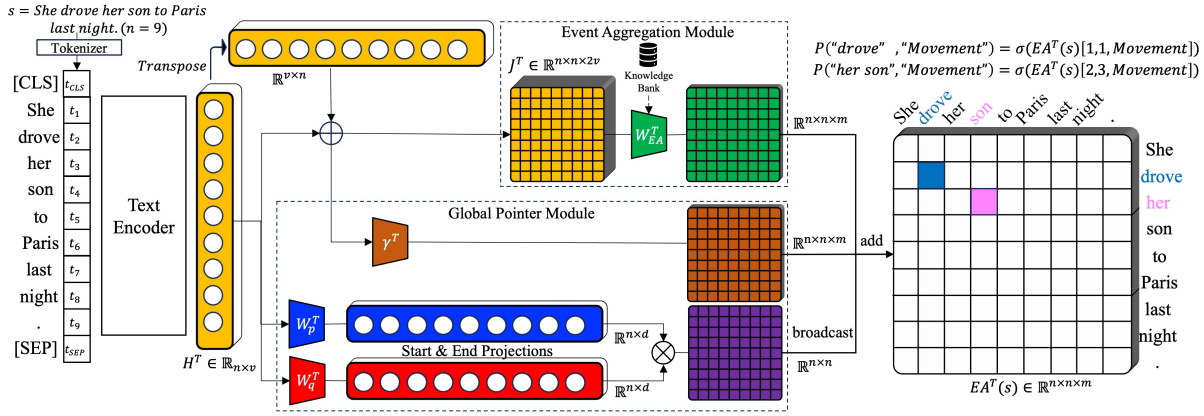


Figure 2: The architecture of Event Aggregation Network, including a text encoder, an event aggregation module and a Global Pointer module. In this figure, the workflow of trigger extraction is illustrated, where the input format of text encoder is "[CLS] + Sentence + [SEP]". In argument extraction, the architecture is slightly different which is introduced in § 3.1.2, while the input format of text encoder is "[CLS] + Sentence + [SEP] + EventType + [SEP]".

Symbol	description
$s = \{t_1, ..., t_n\}$	A sequence with n tokens.
$s[x:y] = \{t_x, ..., t_y\}$	A span of s with a start index x and an end index y.
E, R	The universial sets of event type and role in event extraction schema.
m, l	Numbers of event types and argument roles in event extraction schema
T_e, A_r	All the triggers and arguments labeled with event type e and role r in dataset.

Table 1: Notation Table

Argument Extraction The event aggregation module of argument extraction is basically similar with trigger extraction with following differences. For span expanding matrix J^A of argument extraction, an element $J^A[x, y]$ denotes the representation difference between $s[x:y]$ and the trigger t in the context, as shown in Eq. 6.

$$J^A[x, y] = h_{head(t)} \oplus h_{tail(t)} - h_x \oplus h_y \quad (6)$$

For knowledge bank $K^A_{EA} \in \mathbb{R}^{l \times 2v}$ of argument extraction, each element $K_r \in \mathbb{R}^{2v}$ denotes the representation difference between all the arguments with role r and their triggers, derived by Eq. 7.

$$K_r = \frac{\sum_{a \in A_r} (h_{head(t)} \oplus h_{tail(t)} - h_{head(a)} \oplus h_{tail(a)})}{||A_r||} \quad (7)$$

where t denotes the trigger in the context of a; $h_{head(a)}$ and $h_{tail(a)}$ denote the text encoder's output of the head and tail of argument a.

For consistency layer $W^A_{EA} \in \mathbb{R}^{l \times 2v}$ of argument extraction, we also introduce a linear layer to weight the consistency between each span and each role. The K^A_{EA} is also used to initialize and update the W^A_{EA} in argument extraction. In addition, an extra term is introduced to weight the relevance between a span and a trigger by dot product between their representations. The consistency between $s[x:y]$ and role r with the trigger t is scored by Eq.8.

$$c^A = W^A_{EA}[r] J^A[x, y] + W^A_b[r][h_{head(t)} \oplus h_{tail(t)}] \cdot [h_x \oplus h_y] \quad (8)$$

where $W^A_b \in \mathbb{R}^l$ is a liner layer to balance the two terms.

3.1.3 Global Pointer Module for span extraction

We take trigger extraction and argument extraction as sequence labeling-based tasks, and adopt Global Pointer (GP) (Su et al., 2022), a widely-used model for sequence labeling, in CEAN. Given s, GP scores if $s[x:y]$ can be extracted as an event-related fragment of o by Eq. 9.

$$GP^*(s)[x, y, o] = (W^*_p h_x)^\top (W^*_q h_y) + \gamma^*[o](h_x \oplus h_y) \quad (9)$$

where o denotes an event type in event extraction or an argument role in argument extraction. $GP^*(s)$ is the global pointer module's output matrix for the given input s, which is $GP^T(s) \in \mathbb{R}^{n \times n \times m}$ in trigger extraction and $GP^A(s)^{n \times n \times l}$ in argument extraction. W_p and W_q are the parameters for the start and end projections, which are $W^T_p, W^T_q \in \mathbb{R}^{v \times d}$ in trigger extraction and $W^A_p, W^A_q \in \mathbb{R}^{v \times d}$

in argument extraction. Here, d is the projection dimension. γ is the classification weight, which is $\gamma^T \in \mathbb{R}^{m\times 2v}$ in trigger extraction and $\gamma^A \in \mathbb{R}^{l\times 2v}$ in argument extraction.

Our proposed event aggregation network works by the linear combination of Event Aggregation Module and Global Pointer Module. Given s, Event Aggregation Network uses Eq. 10 to score if $s[x:y]$ can be extracted as an event-related fragment of o.

$$EA^*(s)[x,y,o] = GP^*(s)[x,y,o] + c^* \quad (10)$$

where EA^* is the output of the event aggregation network, which is EA^T in trigger extraction and EA^A in argument extraction. Here, EA^T is derived by the sum of GP^T and c^T, while EA^A is derived by the sum of GP^A and c^A.

With $EA^*(s)$, we use L^*_{SL} to denote the supervised loss function in trigger and argument extraction, which is Multilabel Categorical Cross Entropy (Su et al., 2022) in Eq. 11.

$$\begin{aligned} L^*_{SL} = &\frac{1}{||O||}\sum_{o\in O}[\log(\sum_{i\in\Omega_o^{pos}} e^{(-EA^*(s)[x_i,y_i,o])}+1) \\ &+\log(\sum_{j\in\Omega_o^{neg}} e^{(EA^*(s)[x_j,y_j,o])}+1)] \end{aligned} \quad (11)$$

where O is E in trigger extraction and R in argument extraction. $\Omega_o^{pos}, \Omega_o^{neg}$ are all the spans that labeled and not labeled as an event-related fragment of o, with $||\Omega_o^{pos}|| + ||\Omega_o^{neg}|| = \frac{1}{2}\binom{n}{2}$;

3.2 Span-level Contrastive Learning

Directly using augmented samples with noise as training data for supervised learning can mislead the model convergence. Thus, we propose a span-level contrastive learning loss function which transforms the supervised learning process on the whole augmented sentences into a similarity measurement on triggers and arguments. This can help the training process to be better shielded from augmented noise such as incorrect or incomplete mislabelings. Data augmentation is introduced in § 4.

Given a sample s from the original dataset, s^+ is the sequence of a positive sample derived from augmentation; S^- are sequences of negative samples, chosen from the same minibatch, which are labeled with event types or argument roles different from those of s. We use $s[x_o:y_o]$ and $s^+[x_o^+, y_o^+]$ to denote the event-related fragment of o in the original and augmented positive samples, and use $s^-[x_o^-, y_o^-]$ to denote the spans not labeled as event-related fragment of o from the negative samples. We define L^*_{CL} as the contrastive loss function which is shown in Eq.12.

$$L^*_{CL} = \frac{KL(D^*(s,x_o,y_o,o), D^*(s^+,x_o^+,y_o^+,o))}{[\frac{\sum_{s^-\in S^-} KL(D^*(s,x_o,y_o,o), D^*(s^-,x_o^-,y_o^-,o))}{||S^-||}]} \quad (12)$$

where KL denotes the Kullback-Leibler divergence. $D^*(s,x,y,o)$ is the probability distribution that $s[x:y]$ can be extracted as an event-related fragment of o, which is described as Bernoulli distributions shown in Eq. 13.

$$D^*(s,x,y,o) \sim BN(\sigma(EA^*(s)[x,y,o])) \quad (13)$$

where the σ denotes the sigmoid function and the BN denotes the Bernoulli distribution.

Algorithm 1 Training Process of CEAN.

Input: Original training dataset S and augmentation dataset S^+; A pre-trained text encoder θ_0; Nums of the training epoch N and batch size bs; Hyperparameters α, β;

Output: θ^*_N,$W^*_{GP} = \{W^*_p, W^*_q, \gamma^*\}$,$W^*_{EA}$ (and W^A_b);

```
Randomly intialize the parameter of W*_GP;
for i = 1 → N do
    Calculate K*_EA with θ*_{i-1}, S, S+;
    if i = 1 then
        W*_EA ← K*_EA
    else
        W*_EA ← βW*_EA + (1 − β)K*_EA
    end if
    for j = 1 → (N//bs) do
        Get the original samples S_j;
        Select positive samples S+_j from S+;
        Select negative samples S−_j from S_j;
        Calculate the output of EAN to S_j and S+_j with θ*_{i-1},W*_GP,W*_EA (and W^A_b);
        Calculate L*_SL by S_j;
        Calculate L*_CL by S_j, S+_j and S−_j;
        Calculate L*_sum by L*_SL and L*_CL;
        Use L*_sum to update the parameters θ*_i,W*_GP,W*_EA (and W*_b) by back propogation;
    end for
end for
```

3.3 Training Contrastive Event Aggregation Network

Finally, the CEAN is trained with the weighted sum loss between the L^*_{SL} in Eq. 11 and the L^*_{CL}

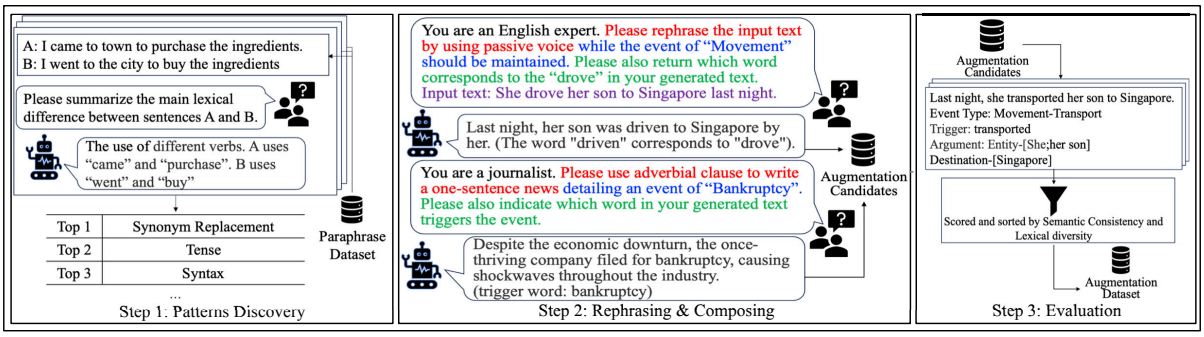


Figure 3: The flowchart of LLM-based data augmentation with three steps. The p_h, p_v, p_c, p_r and x in prompts are marked in black, red, blue, green and purple respectively. Each time we construct a prompt, we firstly sample a pattern for p_v, according to the occurance frequency in step 1.

in Eq. 12, which is shown in Eq.14.

$$L^*_{sum} = L^*_{SL} + \alpha L^*_{CL} \tag{14}$$

where the α is a hyperparameter to balance them. The entire training process can be clearly illustrated by Algorithm 1.

During training process, the W^*_{EA} is basically updated by back propogation algorithm along with the text encoder and the Global Pointer module. Since the parameters of the encoder is updated, the K^*_{EA} accordingly changes. Thus, the W^*_{EA} is additionally updated in a momentum manner before each epoch, which can be illustrated by Eq.15.

$$W^*_{EA} \leftarrow \beta W^*_{EA} + (1-\beta) K^*_{EA} \tag{15}$$

where K^*_{EA} denotes the knowledge bank output by the text encoder after the $(i-1)$ epoch, $\beta \in [0, 1]$ is the momentum coefficient hyperparameter.

4 LLM-based Data Augmentation

This section describes LLM-based data augmentation shown in Figure 3. The paraphrase patterns and the prompt engineering for two paradigms are introduced, followed by evaluation metric.

4.1 Paraphrase Patterns

Lexical diversity can be improved by various patterns in word-level or sentence-level (Gao et al., 2023a). We firstly leverage an LLM to automatically discover paraphrase patterns on Parabank (Hu et al., 2019), a large-scale paraphrase dataset. According to the frequency, main patterns include synonym replacement and transformations on tense, part of speech, voice and sentence structure, from token-level to sentence-level. Prompts are accordingly designed to instruct an LLM to rephrase or compose samples based on main patterns.

Fragment	Function
x	An original sentence for rephrasing or an event structure for composing.
p_v	Instruct the LLM to generate text with a given pattern discovered in § 4.1.
p_c	Inform the LLM of the event structure to preserve after rephrasing.
p_r	Instruct the LLM to annotate triggers and arguments on the generated sentence.
p_h	Role prompting fragment instructing an LLM to impersonate an expert.

Table 2: Prompt engineering for LLM-based data augmentation. The p_c is only necessary in the rephrasing paradigm while the others are shared by two paradigms.

4.2 Prompt Engineering for Two Paradigms

The proposed data augmentation scheme includes a rephrasing paradigm and a composing paradigm. The former one makes word-level or sentence-level modifications on original samples while the latter one composes entirely new sample with given event description. They are conducted by prompt engineering with the fragments of x, p_v, p_c, p_r, p_h listed in Table 2, whose examples are provided by Table 5 in § A. Augmented samples from the two paradigms are collected as augmented candidates.

4.3 Evaluation Metrics

For the augmentation candidates in § 4.2, we evaluate them from semantic consistency and lexical diversity. For semantic consistency evaluation, we introduce the pre-trained text encoder in Eq. 1, using "`[CLS]`" tokens as sentence representations. For lexical diversity evaluation, word-level Levenshtein Distance is used. Given candidate s' with event type e, the semantic consistency and lexical diversity are calculated by Eq. 16 and 17.

$$Con(s', e) = \min_{s \in D_e} \left(\frac{cosine(H_{s'}[\texttt{CLS}], H_s[\texttt{CLS}]) + 1}{2} \right) \quad (16)$$

$$DIV(s', e) = \min_{s \in D_e} \left(\frac{LD(s', s)}{||s'||} \right) \quad (17)$$

where D_e denote all the original samples labeled with e; the $LD(\cdot)$ denotes the word-level Levenshtein Distance. Samples with high lexical diversity and high semantic consistency are desired, so the evaluation function is defined as Eq. 18.

$$\phi(s', e) = Con(s', e) + \lambda DIV(s', e) \quad (18)$$

where λ is a coefficient to balance them. Eq 18 is used to select samples from the candidates of rephrasing and composing paradigms separately. We use ρ to denote the number ratio of the rephrasing samples and the composing samples.

5 Experiment

5.1 Dataset and Evaluation Criteria

To validate our contributions, experiments are performed on the ACE05-E$^+$ and ERE-EN datasets, both of which contain multi-token event triggers and pronoun roles. For a fair comparison, both dataset split and evaluation criteria align with the previous work (Lin et al., 2020). The ACE05-E$^+$ dataset, with a schema of 33 event types and 22 roles, is split into a training set, a validation set and a test set with 4419, 468 and 424 events. The ERE-EN dataset, with a schema of 38 event types and 21 roles, is split into a training set, a validation set and a test set with 6208, 525 and 551 events. Other information of the datasets is listed in Table 6 in § A. In the experiments, the result of trigger classification(Tri-C) are used as input features in argument role classification(Arg-C). The metrics of Precision(P), Recall(R) and F1-score($F1$) are calculated based on the following criteria while the $F1$ of each task is the pivital metric for comparison.

- A trigger is correctly classified if its offset and event type match the golden label.
- An argument is correctly classified if its offset, event type and role match the golden label.

5.2 Expermental Setup

Computational facilities and software environment used for the experiments is listed by Table 7 in § A. Hyperparameter selections are listed by Table 8 in § A. A pre-trained MPNet-BASE (Song et al., 2020) is used as the text encoder. The Adam optimizer (Kingma and Ba, 2017) is used for model training. To derive the augmentation dataset, we firstly generate a candidate dataset which is 4 times size of the original dataset D_0 by using text-davinci-003 (Ouyang et al., 2022) in our augmentation scheme. Then, the candidate samples are sorted by Eq 18 while only the high-scoring samples are retained. We set the retaining proportion to 12.5%, 25% and 50%. Based on the performances on validation set, we select 25% to produce an augmentation dataset D_A, sharing the same size of D_0. To further demonstrate our LLM-based data augmentation scheme, two extra augmentation datasets D_{SR} and D_{MTP}, sharing the same size with D_A, are produced by synonym replacement and masked token prediction which are listed by Table 9 in § A. For reproducibility, experiments are performed under 3 random seeds and the medium result is chosen for overall performance comparison.

5.3 Overall Performance

The studies recently published for Tri-C and Arg-C on ACE05-E$^+$ and ERE-EN are introduced for comparisons. Overall performances of the comparison are listed in Table 3, where the CEAN denotes an Event Aggregation Network trained with original and augmentation dataset by the loss in Eq. 14.

Compared to the existing methods in Table 3, our approach achieves best $F1$ on both datasets. In Tri-C and Arg-C of ACE05-E$^+$, our approach improves the $F1$ by 2.7% and 0.4%. In Tri-C and Arg-C of ERE-EN, our approach improves the $F1$ by 0.8% and 0.3%. Our approach manages to improve P and keep a competitive R.

In conclusion, our approach outperforms all the other studies on each dataset, becoming the state-of-the-art of ACE05-E$^+$ and ERE-EN.

5.4 Ablation Study

To explore how the performances are affected by event aggregation module, data augmentation and contrastive learning, ablation study is conducted on ACE05-E$^+$. Starting by a text encoder and a GP module, we implement each of the three contributions. Detailed description and performances of the ablation study are listed in Table 4.

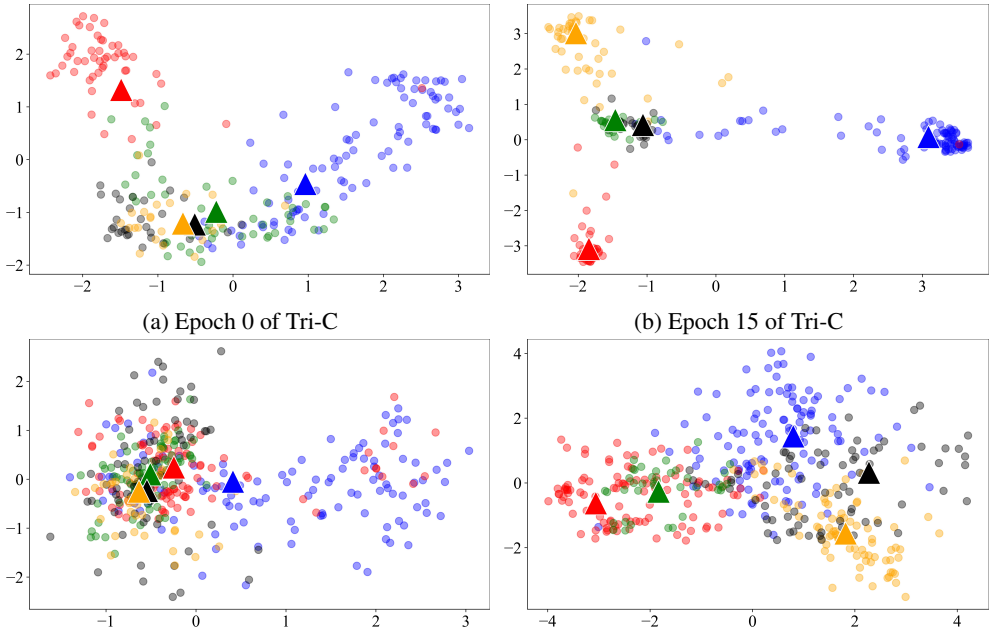
Experiments of G1 are used to validate the event aggregation network, including event aggregation module and knowledge bank. In G1-2, event aggregation module without knowledge bank is evaluated, which brings -0.6% and $+0.3\%$ on the $F1$ of Tri-C and Arg-C to G1-1. In G1-3, knowledge bank is evaluated based on event aggregation module,

Methods	ACE05-E$^+$						ERE-EN					
	Tri-C			**Arg-C**			**Tri-C**			**Arg-C**		
	***F*1**	P	R	***F*1**	P	R	***F*1**	P	R	***F*1**	P	R
ONEIE (Lin et al., 2020)	72.8	72.1	73.6	54.8	55.4	54.3	59.1	58.4	59.9	50.5	51.8	49.2
CLEVE (Wang et al., 2021)	79.8	78.1	81.5	61.1	55.4	**68.0**	-	-	-	-	-	-
InterIE (Fan et al., 2022)	75.3	-	-	60.1	-	-	-	-	-	-	-	-
Text2Event (Lu et al., 2021)	71.8	71.2	72.5	54.4	54.0	54.8	59.4	59.2	59.6	48.3	49.4	47.2
GTEEDP (Liu et al., 2022b)	74.3	67.3	**83.0**	54.7	49.8	60.7	66.9	61.9	**72.8**	55.1	51.9	**58.8**
HDGSE (Shi et al., 2023)	77.2	75.5	79.0	57.7	57.6	57.8	66.1	64.5	67.9	53.5	**54.5**	52.6
CEAN	**82.5**	**82.9**	82.1	**61.5**	**60.9**	62.2	**67.7**	**69.4**	66.1	**55.4**	54.1	56.8

Table 3: Overall performance comparisons between existing methods and CEAN on ACE05-E$^+$ and ERE-EN.

Index	Model	W^*_{EA}	Loss	Data	**Tri-C**			**Arg-C**		
					***F*1**	P	R	***F*1**	P	R
G1-1	GP	-	L^*_{SL}	D_0	77.1	80.3	74.1	59.4	55.0	64.7
G1-2	EAN	Random Initialization	L^*_{SL}	D_0	76.5	78.8	74.3	59.7	56.0	63.9
G1-3	EAN	Knowledge Bank	L^*_{SL}	D_0	81.2	84.1	78.5	60.5	59.4	61.6
G2-1	EAN	Knowledge Bank	L^*_{SL}	D_0+D_A	81.5	82.3	80.8	60.6	56.6	65.3
G2-2(a)	CEAN	Knowledge Bank	L^*_{sum}	D_0+D_{SR}	81.5	82.0	81.0	60.5	59.6	61.4
G2-2(b)	CEAN	Knowledge Bank	L^*_{sum}	D_0+D_{MTP}	81.6	82.1	81.1	60.3	59.1	61.5
G2-2(c)	CEAN	Knowledge Bank	L^*_{sum}	D_0+D_A	82.5	82.9	82.1	61.5	60.9	62.2

Table 4: Ablation study on ACE05-E$^+$. Event Aggregation Network (EAN) is validated in G1 by a GP model and an EAN with different settings. Data augmentation and span-level contrastive learning are verified in G2 with original dataset D_0 and augmented datasets $\{D_{SR}, D_{MTP}, D_A\}$ as training data, and L^*_{SL} or L^*_{sum} as loss functions.

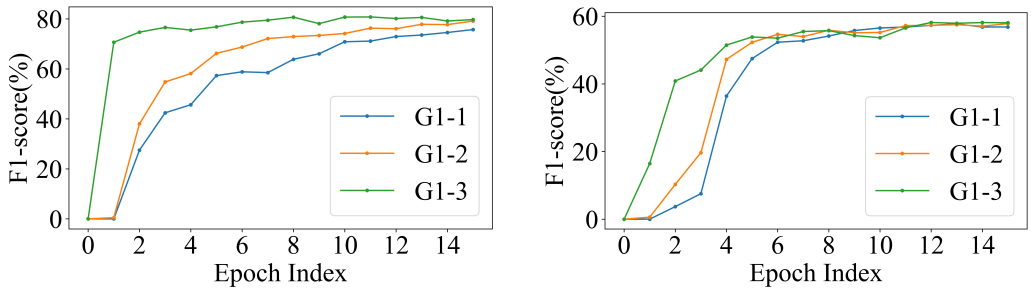


(a) Epoch 0 of Tri-C (b) Epoch 15 of Tri-C

(c) Epoch 0 of Arg-C (d) Epoch 15 of Arg-C

Figure 4: Visualization of event aggregation by PCA on ACE05-E$^+$. The top 5 event types and roles of ACE05-E$^+$'s test set are chosen for demonstration, which are *Attack*, *Meet*, *Transport*, *Transfer-Ownership*, *End-Position* and *Entity*, *Place*, *Person*, *Artifact*, *Destination*. Their points are marked in blue, red, green, black and orange respectively. Their trigger and argument spans in test set are selected, whose slices are extracted from J^* and are reduced into 2-dim vectors, which are plotted as the dot points. The K_e and K_r of these top event types and roles are also reduced into 2-dim vectors, which are plotted as the triangle points. A dot point is correctly classified in this figure if it shares the same color with its nearest triangle point.

which brings 4.1% and 1.1% improvement on $F1$ of the two tasks to G1-1, and $+4.7\%$ and $+0.8\%$ to G1-2. For a better demonstration, we provide the visualization of event aggregation in Figure 4 by Principal Component Analysis (PCA). After 15 training epochs, the event aggregation accuracy,



(a) Trigger Classfication

(b) Argument Classfication

Figure 5: $F1$ on test set in the first 15 epochs of G1-3. The W_{EA} is initialized by knowledge bank. The $F1$ on test set is quickly improved from the first epoch.

defined by the ratio of the correctly classified dot points to all the dot points, is improved from 63.4% and 41.4% to 81.9% and 62.7% in Tri-C and Arg-C. In addition to performance improvement, event aggregation module can help the model to obtain a faster convergence, proven by Figure 5.

Experiments of G2 are used to verify the LLM-based augmentation scheme and contrastive learning strategy. In G2-1, using the augmentation scheme without the span-level contrastive learning loss L_{sum} brings $+0.3\%$ and $+0.1\%$ on the $F1$ of Tri-C and Arg-C to G1-3. In G2-2(c), using the augmentation scheme with L_{sum} brings $+1.3\%$ and $+1.0\%$ on the $F1$ of Tri-C and Arg-C to G1-3. Also, in G2-2(c), our augmentation scheme brings $+1.0\%$, $+1.0\%$ and $+0.9\%$, $+1.2\%$ on the $F1$ of Tri-C and Arg-C to G2-2(a) and G2-2(b).

In conclusion, all proposed contributions are proven effective for event extraction, and the combination of them reaches the best performance.

6 Conclusion

We propose a Contrastive Event Aggregation Network with LLM-based Augmentation to promote low-resource learning and reduce data noise for event extraction. CEAN introduces event knowledge into supervised learning by establishing knowledge bank for triggers and arguments. We design an LLM-based augmentation scheme including a multi-pattern rephrasing paradigm and a data-free composing paradigm to improve lexical diversity. We introduce span-level contrastive learning to reduce data noise unavoidably originated in the augmentation. Experiments on the ACE2005 and ERE-EN datasets demonstrate that our proposed approach achieves new state-of-the-art results.

Limitations

Similar with many pipeline-based methods, the performance of CEAN on Arg-C is limited by the error propogation from trigger classification. That is, the incorrectly extracted triggers in Tri-C are used as the input features of Arg-C, resulting in amplification of errors. We take the golden triggers as the input of G1-3 and obtain a $F1 = 70.3\%$ on Arg-C. That means the error propogation from Tri-C leads to a 9.8% decreasement on the $F1$ of Arg-C. Future work should explore transforming CEAN into a joint-based method to alleviate effect of error propogation.

Ethics Statement

We are aware of and fully agree with the ACL Ethics Policy. Large language model is used for data augmentation in event extraction. There is a low possibility that the generate contents include biased, toxic, counterfactual or harmful texts. Thus, it is important to evaluate all potential issues when the model is deployed in real event extraction task.

References

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Data augmentation for information retrieval using large language models.

Hu Cao, Jingye Li, Fangfang Su, Fei Li, Hao Fei, Shengqiong Wu, Bobo Li, Liang Zhao, and Donghong Ji. 2022. Oneee: A one-stage framework for fast overlapping and nested event extraction.

Xiang Chen, Lei Li, Ningyu Zhang, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022. Relation extraction as open-book examination: Retrieval-enhanced prompt tuning. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2443–2448.

Yubo Chen, Shulin Liu, Xiang Zhang, Kang Liu, and Jun Zhao. 2017. Automatically labeled data generation for large scale event extraction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 409–419.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, Hongmin Cai, Lichao Sun, Quanzheng Li, Dinggang Shen, Tianming Liu, and Xiang Li. 2023. Auggpt: Leveraging chatgpt for text data augmentation.

Siqi Fan, Yequan Wang, Jing Li, Zheng Zhang, Shuo Shang, and Peng Han. 2022. Interactive information extraction by semantic information graph. IJCAI.

Jun Gao, Changlong Yu, Wei Wang, Huan Zhao, and Ruifeng Xu. 2023a. Mask-then-fill: A flexible and effective data augmentation framework for event extraction. *CoRR*, abs/2301.02427.

Jun Gao, Huan Zhao, Changlong Yu, and Ruifeng Xu. 2023b. Exploring the feasibility of chatgpt for event extraction.

J. Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. 2019. Parabank: Monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation.

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*, 35(10):9946–9958.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-08: Hlt*, pages 254–262.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization.

Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023. Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness.

Jingye Li, Hao Fei, Jiang Liu, Shengqiong Wu, Meishan Zhang, Chong Teng, Donghong Ji, and Fei Li. 2021a. Unified named entity recognition as word-word relation classification.

Qian Li, Jianxin Li, Jiawei Sheng, Shiyao Cui, Jia Wu, Yiming Hei, Hao Peng, Shu Guo, Lihong Wang, Amin Beheshti, and Philip S. Yu. 2022. A survey on deep learning event extraction: Approaches and applications. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–21.

Qian Li, Jianxin Li, Jiawei Sheng, Shiyao Cui, Jia Wu, Yiming Hei, Hao Peng, Shu Guo, Lihong Wang, Amin Beheshti, et al. 2021b. A compact survey on event extraction: Approaches and applications. *IEEE Transactions on Industrial Informatics*, 14(9):1–21.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 789–797.

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.

Shulin Liu, Yubo Chen, Shizhu He, Kang Liu, and Jun Zhao. 2016. Leveraging framenet to improve automatic event detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2134–2143.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022a. Dynamic prefix-tuning for generative template-based event extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5216–5228. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022b. Dynamic prefix-tuning for generative template-based event extraction. *arXiv preprint arXiv:2205.06166*.

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.

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.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with 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 2: Short Papers)*, pages 365–371.

Thien Huu Nguyen and Ralph Grishman. 2016. Modeling skip-grams for event detection with convolutional neural networks. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 886–891.

Jinzhong Ning, Zhihao Yang, Zhizheng Wang, Yuanyuan Sun, and Hongfei Lin. 2023. Odee: A one-stage object detection framework for overlapping and nested event extraction. In *Proceedings of*

the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23, pages 5170–5178. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Feiliang Ren, Longhui Zhang, Shujuan Yin, Xiaofeng Zhao, Shilei Liu, Bochao Li, and Yaduo Liu. 2021. A novel global feature-oriented relational triple extraction model based on table filling.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Ge Shi, Yunyue Su, Yongliang Ma, and Ming Zhou. 2023. A hybrid detection and generation framework with separate encoders for event extraction. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3155–3172.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867.

Jianlin Su, Ahmed Murtadha, Shengfeng Pan, Jing Hou, Jun Sun, Wanwei Huang, Bo Wen, and Yunfeng Liu. 2022. Global pointer: Novel efficient span-based approach for named entity recognition.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. *arXiv preprint arXiv:1909.03546*.

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juanzi Li, and Jie Zhou. 2021. Cleve: Contrastive pre-training for event extraction. *arXiv preprint arXiv:2105.14485*.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, Yong Jiang, and Wenjuan Han. 2023. Zero-shot information extraction via chatting with chatgpt.

Chenxi Whitehouse, Monojit Choudhury, and Alham Fikri Aji. 2023. Llm-powered data augmentation for enhanced crosslingual performance.

Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in neural information processing systems*, 33:6256–6268.

Sen Yang, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li. 2019a. 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.

Yiying Yang, Zhongyu Wei, Qin Chen, and Libo Wu. 2019b. Using external knowledge for financial event prediction based on graph neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2161–2164.

Tianlong Zhang, Zhe Xue, Xu Shao, Yawen Li, Junjiang Wu, and Yunfei Long. 2023. Cppfee: A cascade pointer prediction framework for financial event extraction. In *2023 5th International Conference on Data-driven Optimization of Complex Systems (DOCS)*, pages 1–8.

A Appendix

Paradigm	Prompt	Augmentation Result
Rephrasing	You are an English expert. Please rephrase the input text by using passive voice while the event of "transfer-ownership" should be maintained. Please also return which word corresponds to the buyer of this event in your generated text. Input text: *He sold his cinema to Disney for over $1 billion.*	*The cinema was sold to Disney for over $1 billion by him. The buyer in this event is Disney.*
Rephrasing	You are an English expert. Please rephrase the input text by using adverbial clause while the event of "beborn" should be maintained. Please also return which word in your generated text corresponds to the word "birth". Input text: *The movie star announced the birth of her second daughter on social media.*	*On social media, the movie star announced that her second daughter was born. The word "birth" corresponds to "born".*
Rephrasing	You are an English expert. Please rephrase the input text by modifying the syntax while the event of "Start-Position" should be maintained. Please also return which word in your generated text triggers this event. Input text: *In her last year at university, she began interning in government departments.*	*She started interning in government departments during her last year at university. (Trigger word: started)*
Rephrasing	You are an English expert. Please rephrase the input text by replacing the word "leaving" with its synonym while the event of "End-Position" should be maintained. Please also return which word in your generated text is the synonym. Input text: *After leaving the company where he worked for 5 years, he returned to teach at the university.*	*After departing from the company where he worked for 5 years, he returned to teach at the university. (Synonym: departing)*
Composing	You are a journalist. Please use exclamative clause to write an one-sentence news report detailing an event of "Marriage". Please also indicate which word in your generated text triggers the event.	*"Congratulations to the newlyweds who just exchanged rings and said 'I do'!" - Trigger words: exchanged rings, said 'I do'.*
Composing	You are a journalist. Please use a gerund to write an one-sentence news report detailing an event of "Conflict:Convict". Please also indicate which word in your generated text triggers the event.	*The victim's family demands justice for the brutal murder, convicting the suspect being their only hope to end the conflict. (The word "convicting" triggers the event.)*
Composing	You are a journalist. Please use imperative clause to write an one-sentence news report detailing an event of "Movement:Transport". Please also indicate which word in your generated text is the destination of the movement.	*Evacuate the passengers immediately as a train carrying hazardous chemicals derailed en route to the industrial zone. (destination: industrial zone)*
Composing	You are a journalist. Please use perfect tense to write an one-sentence news report detailing an event of "Life:Injure". Please also indicate which word in your generated text is the victim of the event.	*The athlete has been rushed to the hospital after having sustained a severe injury while competing in the championship. (victim: athlete)*

Table 5: Examples of LLM-based data augmentation scheme with rephrasing and composing paradigms.

Dataset	LDC Catalog Number	License
ACE2005	LDC2006T06	LDC User Agreement for Non-Members
ERE-EN	LDC2015E29/68/78	LDC User Agreement for Non-Members

Table 6: Information of the used datasets.

Issue	Information
Computing Infrastructure	Nvidia Tesla T4
Total Computational Budget	6 GPU hours
Software Environment	Python 3.6.3 & Pytorch 1.6.0

Table 7: Computational facilities and software information of the experiments.

Hyperparameters	Value
Sequence max input length n	128
Batch size bs	8
Training epoch number in trigger extraction N	25
Training epoch number in argument extraction N	50
Learning rate of the GP module	$2e-4$
Learning rate of the other modules	$2e-5$
β_1 of Adam optimizer	0.9
β_2 of Adam optimizer	0.999
ϵ of Adam optimizer	$1e-8$
Weight decay of Adam optimizer	0
Hidden dimension of the text encoder v	768
Projection dimension in trigger extraction d	64
Projection dimension in argument extraction d	512
Coefficient of loss function α	$1e-2$
Coefficient of event aggregation module in trigger extraction β	0.8
Coefficient of event aggregation module in argument extraction β	0.7
Size ratio between the augmentation candidates and the original dataset	4
Size ratio between the augmentation dataset and the original dataset	1
Coefficient of augmentation evaluation λ	1
Number ratio between the rephrasing samples and the composing samples ρ	1

Table 8: Hyperparameter selections for the experiments.

Augmentation Method	Dataset	Descrition
Synonym Replacement	D_{SR}	Given an original sample, we randomly sampled 15% of tokens and replace them with their synonym words retrieved from WordNet, a lexical database widely used in natural language processing researches, to generate the augmented sample.
Masked Token Prediction	D_{MTP}	Given an original sample, we randomly sampled 15% of tokens and replaced each of them with a `[MASK]` token. Then, the augmented sample was derived by using a pre-trained model to make masked token predictions on these positions.
LLM-based Data Augmentation	D_A	First, we generate a candidate dataset which is 4 times size of the original dataset by the first two steps of our augmentation scheme. Then, the candidate samples are sorted by Eq 18 while only the top 25% are retained.

Table 9: Information of the augmentation datasets produced for experiment G2-2. Each of the three datasets shares the same size with the original dataset.