Generate-then-Revise: An Effective Synthetic Training Data Generation framework for Event Detection

Huidong Du, Hao Sun, Pengyuan Liu, Dong Yu
School of Information Science, Beijing Language and Culture University
Beijing, China
yzqtdu@gmail.com, 202221198129@stu.blcu.edu.cn
liupengyuan@pku.edu.cn, yudong_blcu@126.com

Abstract

Large language models (LLMs) struggle with event detection (ED) due to the structured and variable number of events in the output. Existing supervised approaches rely on a large amount of manually annotated corpora, facing challenges in practice when event types are diverse and the annotated data is scarce. We propose Generate-then-Revise (GtR), a framework that leverages LLMs in the opposite direction to address these challenges in ED. GtR utilizes an LLM to generate high-quality training data in three stages, including a novel data revision step to minimize noise in the synthetic data. The generated data is then used to train a smaller model for evaluation. Our approach demonstrates significant improvements on the low-resource ED. We further analyze the generated data, highlighting the potential of synthetic data generation for enhancing ED performance.

1 Introduction

Event detection (ED) aims to extract structured information from natural language text. Previous researches typically involve supervised fine-tuning on a large amount of manually annotated corpora, facing challenges in practice due to the vast number of event types in real-world scenarios and the high cost of manually annotating data. Recently, large language models (LLMs) like ChatGPT and Llama2 have made remarkable progress in many tasks in low-resource scenarios, enabling some tasks to achieve performance comparable to fully supervised learning in zero-shot or few-shot conditions. However, the success of large models in natural language understanding and reasoning tasks is difficult to directly replicate in ED (Ma et al., 2023b; Qin et al., 2023; Han et al., 2023; Li et al., 2023).

Current research indicates that directly using prompt instructions to instruct models to perform ED generally results in poor performance (Huang et al., 2024). As shown in Figure 1, researchers evaluate the average performance of several LLMs on ED in a test set containing 250 documents. Among them, Mixtral-8x7B-Instruct-v0.1 achieves the best trigger identification (TI) F1 score of 37.5 in a 64-shot setting, while for trigger classification (TC), the performance of various models is even worse, and there is no linear increase in performance with the increase in samples.The failure of LLMs in ED is related to the characteristics of the task itself. ED requires generating a structured list of events of varying lengths. Although these models are not adept at structured output tasks, recent research has found that models can generate meaningful inputs based on outputs in the opposite direction (Josifoski et al., 2023), effectively transforming the original computation from $\mathcal{P}(y_k|x)$ to $\mathcal{P}(x|y_k)$.

This suggests that we can leverage the asymmetry to synthesize high-quality data for ED. Data generation avoids the difficulties of outputting structured content while reducing reliance on manually annotated data. However applying LLMs to data generation on ED is non-trivial. Unlike classification tasks like sentiment analysis (which use sentence-level labels like positive or negative) (Ye et al., 2022a; Ye et al., 2022b), ED requires generating data with token-level labels (Gao et al., 2022), i.e, each sentence should contain both the correct triggers and event types. Simply prompting LLMs can lead to noisy data, as

This work is funded by the Humanity and Social Science Youth foundation of Ministry of Education (23YJAZH184) and the Fundamental Research Funds for the Central Universities in BLCU (No. 21PT04)

Model	TI	TC
Mixtral-8x7B-Instruct-v0.1 (2-shot)	30.4	10.2
Mixtral-8x7B-Instruct-v0.1 (6-shot)	34.4	10.6
Mixtral-8x7B-Instruct-v0.1 (16-shot)	35.4	12.1
Mixtral-8x7B-Instruct-v0.1 (32-shot)	36.7	13.8
Mixtral-8x7B-Instruct-v0.1 (64-shot)	37.5	14.6
gpt-3.5-turbo-1106 (2-shot)	33.9	11.8
gpt-3.5-turbo-1106 (16-shot)	35.2	12.3
Llama-2-70b-chat-hf (2-shot)	30.6	11.3
Llama-2-70b-chat-hf (6-shot)	32.2	12.4

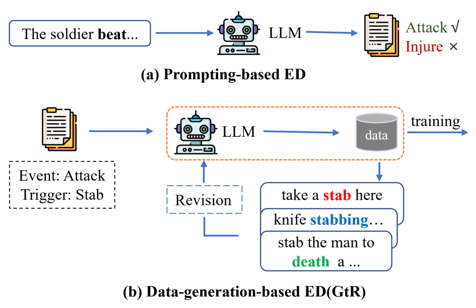

Figure 1: Table of the performance of LLms on ED and Illustrations of ED with LLMs. Compared to prompting, GtR generate synthetic data with LLMs for training. However, data generation may encounter several issues in ED. In (b), the red word represents an incorrect event, blue word suggets competing events (a same trigger for *Attack* and *Injure* event), and the green word represents an accompanying event *Death*. GtR proposes a revision step to address these problems.

shown in Figure 1, including accompanying events, competing events and incorrect events. Unlabled accompanying events emerge frequently in sythesized data and can disrupt the training process.

Existing works leveraging LLMs to generate labeled data in ED often overlook these challenges, primarily relying on sophisticated prompts to produce training data. Ma's approach (Ma et al., 2023a) leverages LLMs for data augmentation but fails to address the noise issues discussed earlier. While existing research offers noise mitigation techniques for classification tasks (e.g., noise-robust loss functions) (Ye et al., 2022b; Gao et al., 2023), these methods are ineffective against the specific problem of accompanying events in ED. To address this gap, we identify three key challenges to overcome when mitigating noise in labeled data generation: (1) Aligned Generation: The LLM must generate data that strictly adheres to the specified triggers and event types. (2) Internal Consistency: The triggers and event types within each generated sample need to be consistent and mutually supportive. (3) Accompanying Event Handling: We need a mechanism to identify and label accompanying events during generation.

To address these issues, we propose a sentence-level event data generation framework called GtR. GtR synthesizes high-quality automatically annotated data in three stages, which can be used in zero-shot scenarios without manual annotation and can also enhance existing data in few-shot settings. The synthesized data also includes adversarial examples, which can significantly improve the robustness of the model. In addition, considering that synthesized data inevitably contains some noise, we introduce a dynamic weighting module to score the synthesized data, reducing the interference of noisy data and improving the performance of models trained on synthesized data.

2 Related works

Zero-Shot Event Detection Zero-shot event detection (ED) aims to identify triggers and event types without accessing training data (Du and Cardie, 2020; Lu et al., 2022; Hsu et al., 2022). Previous approaches for zero-shot ED rely on external tools such as AMR and FrameNet to learn event features from seen events, and apply them to unseen events (Huang et al., 2018; Zhang et al., 2022b). These methods assume that seen and unseen events have similarities, which may not always hold true in real-world scenarios.

Recently, applying LLMs to zero-shot tasks has gained popularity . Compared to fully supervised learning, LLMs alleviate the trouble of limited data. However, when it comes to ED, the prompt-based method still yields relatively unsatisfactory performance (Ma et al., 2023b; Qin et al., 2023; Han et al., 2023; Li et al., 2023).

Dataset-generation-based Zero-shot Learning Some researches have explored an alternative approach leveraging the knowledge of LLMs for zero-shot learning. These methods prompt the LLMs to generate samples for downstream tasks based on the given task descriptions and labels (Schick and Schütze, 2021). With the generated samples, a smaller model with faster speed and lower computational requirements can be trained. These methods have shown significant results in classification tasks. Ye et al. (Ye et al., 2022a) indicates that the synthetic data generated by LLMs is of high-quality in sentiment classification and natural language inference, even surpassing the performance achieved by fine-tuning on original data on some datasets.

Some studies recently explore data generation with LLMs in the realm of information extraction. Josifoski (2023) generates high-quality structured data using LLMs in relation extraction. Ma (2023a) applies data generation in event extraction but fails to address the issues of accompanying and incorrect events. In contrast to these methods, GtR generates samples from scratch, and proposes a revision stage to address the noise, obtaining data that can be directly used for training.

3 Data Generation Framework

In this section, we introduce how GtR generates high-quality labeled training data for ED, i.e. $\mathcal{S}_{syn} = \{\mathcal{X}_{sen}, \mathcal{Y}_{event}\}$.

3.1 Event Template

We define an event template formatted as a JSON array to specify key information about events to guide the following stages. The event template $\mathcal{E} \sim \{e_1, e_2..., e_k\}$, where k is the number of events, consists of two items for every elements.

- **Description**. *Description* is the concrete definition of an event type. For example, the *Injure* event can be defined as *a person experiences physical harm*. *Description* should be precise to avoid ambiguous results when generating triggers and sentences.

- **Examples**. *Examples* provide demonstrations and CoT rationales for generating triggers and sentences, facilitating in-context learning and enhancing their overall quality. Note that we **do not** use examples in zero-shot setting for fair comparison.

3.2 Trigger Expansion

With the guide of an event template, we iteratively leverage LLMs to produce triggers for every event. Formally, given the model $\mathcal{M}$ and a function $\mathcal{F}_{tri} : \mathcal{E} \rightarrow \mathcal{P}_{tri}$, where $\mathcal{P}_{tri}$ denotes the prompt, we obtain generated triggers $\mathcal{T}_{tri} \sim \mathcal{M}(\mathcal{F}_{tri}(\mathcal{E}))$. $\mathcal{F}_{tri}$ retrieves event description and examples from the template, and combines them with an instruction to form the prompt fed to LLMs. The input prompt for *Injure* would be: *list 10 words which mean that a person experiences physical harm.*

3.3 Data Generation

Both triggers and sentences can be generated simultaneously, but in order to get better control over the type and diversity of sentences, we split them into two stages. With the function $\mathcal{F}_{sen} : (\mathcal{E}, \mathcal{T}_{tri}) \rightarrow \mathcal{P}_{sen}$, we gain positive and negative samples $(\mathcal{X}_{sen}, \mathcal{Y'}_{event}) \sim \mathcal{M}(\mathcal{F}_{sen}(\mathcal{E}, \mathcal{T}_{tri}))$, where $\mathcal{P}_{sen}$ denotes the prompt for generating sentences, $\mathcal{X}_{sen}$ denotes the generated sentences and $\mathcal{Y'}_{event}$ represents the events. For every trigger in T_{tri}, $\mathcal{F}_{sen}$ combines description and examples from the template with an instruction to prompt LLMs to output sentences.

The complete prompt for positive and negative samples are nearly identical. For example, given the trigger **kill**, we build the following prompt: *make 5 sentences with the word kill. The sentences should have different backgrounds. The length and structure should vary from each other. The word should* ***mean/not mean*** *causing the death of a person*. LLMs are capable of effectively generating sentences that meet the requirements of the prompts.

3.4 Data Revision

Data obtained from the *Data Generation* step are highly likely noisy, containing incorrect events and accompanying events, e.g., a sentence for *Attack* may also include the *Injure* or *Die* event. Hence, we propose the *Data Revision* step to solve these problems. With $\mathcal{F}_{rev} : (\mathcal{E}, \mathcal{T}_{tri}, \mathcal{X}_{sen}, \mathcal{Y'}_{event}) \rightarrow (\mathcal{X}_{sen}, \mathcal{Y}_{event})$, we get the final training dataset.

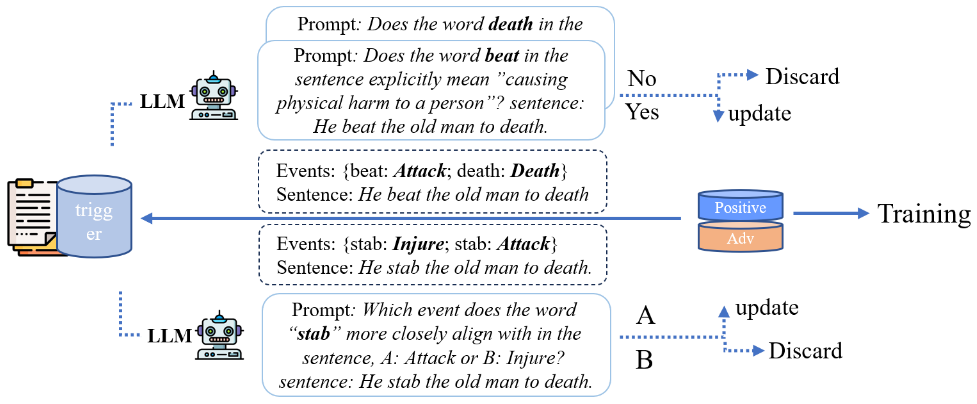
Prompt: Does the word death in the
Prompt: Does the word beat in the sentence explicitly mean "causing physical harm to a person"? sentence: He beat the old man to death.
LLM
No
Yes
Discard
update
trigger
Events: {beat: Attack; death: Death}
Sentence: He beat the old man to death
Events: {stab: Injure; stab: Attack}
Sentence: He stab the old man to death.
Positive
Adv
Training
LLM
Prompt: Which event does the word "stab" more closely align with in the sentence, A: Attack or B: Injure? sentence: He stab the old man to death.
A
B
update
Discard

Figure 2: Data Revision: we address incorrect events, accompanying events and competing events based on the inference of LLM. As depicted in the figure, the LLM determines that event A is more appropriate than event B within current sentence.

Incorrect and Accompanying Events $\mathcal{F}_{rev}$ identifies events within generated text from $\mathcal{X}_{sen}$ using triggers in $\mathcal{T}_{tri}$. It first labels all words or phrases from $\mathcal{T}_{tri}$ as potential events within the sentence. Then $\mathcal{F}_{rev}$ constructs prompts leveraging a LLM to answer Yes/No questions about each potential event. For example, consider the trigger **beat** associated with the *Attack* event and the sentence *He beat the old man to death*. As shown in Figure 2, $\mathcal{F}_{rev}$ labels the word **death** as a potential *Death* event and then builds prompts for *Attack* and *Death* successively. The complete prompt is like *sentence: He beat the old man to death. Does the word beat in the sentence explicitly mean "causing physical harm to a person"?* Events receiving a **Yes** answer are retained, while those receiving a **No** are discarded. This process helps resolve incorrect and accompanying events within the generated text.

Competing Events We need to settle competing events in the samples. In some cases, LLMs mistakenly assign the same trigger in a sentence to different similar events, leading to race conditions. Given the trigger **stab**, LLMs may consider it both as an *Injure* and *Attack* event. To tackle this, we request LLM to explain the difference between these two events. Provided with the explanation as context, LLM determines which event is more accurate within the current sentence. Finally, we modify samples based on the answers.

After *Data Revision*, we gain the final synthetic data $\mathcal{S}_{syn}$ for training.

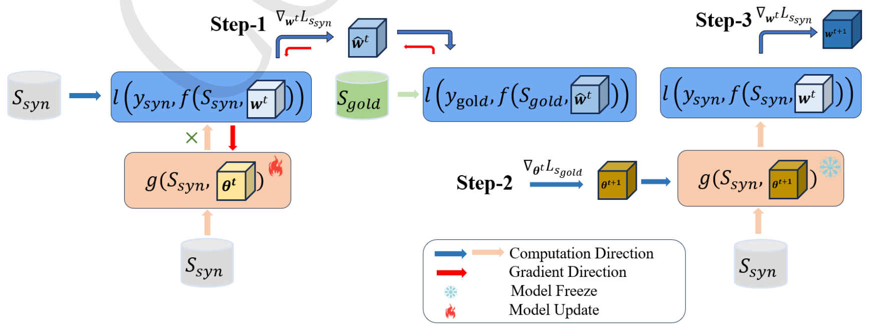
Step-1
Step-2
Step-3
$\nabla_{w^t} L_{s_{syn}}$
$\nabla_{\theta^t} L_{s_{gold}}$
S_{syn}
S_{gold}
$l(y_{syn}, f(S_{syn}, w^t))$
$l(y_{gold}, f(S_{gold}, \hat{w}^t))$
$g(S_{syn}, \theta^t)$
$g(S_{syn}, \theta^{t+1})$
Computation Direction
Gradient Direction
Model Freeze
Model Update

Figure 3: Illustration of the process of weighting.

4 Dynamic Weighting

Data Revision can mitigate the aforementioned noise issues. As the scale of generated data increases, residual noise will inevitably continue to grow, constraining further improvements in model performance. Therefore, we also propose a dynamic weighting module $g(x, \boldsymbol{\theta})$ that assigns weight coefficients to samples in $\mathcal{S}_{syn}$.

4.1 Weighting Model

We first utilize pre-trained language models such as BERT to encode the synthesized data. Given a sample sentence $W = \{w_1, w_2, ..., w_n\}$ of length n, we obtain its encoded feature representation $\hat{W} = \{\hat{w_1}, \hat{w_2}, ..., \hat{w_n}\}$. The sentence contains a list of events, with each event recording a specific trigger w_i and event type e_{w_i}. Next, we initialize learnable event type vectors $E = \{e_1, e_2, ..., e_k\}$, where k is the number of event types. We score the similarity between w_i and e_{w_i} using $g(x, \boldsymbol{\theta})$ as follows:

$$g(w_i) = Norm(Similarity(MLP(\hat{w_i}), e_{w_i})) \tag{1}$$

Here, the similarity is computed using cosine similarity, and the similarity scores are normalized. $g(x, \boldsymbol{\theta})$ outputs scores ranging from 0 to 1 for each trigger in the event list, determining the likelihood of triggering the corresponding event

4.2 Model Optimization

In this subsection, we will detail the optimization process of $g(x, \boldsymbol{\theta})$. We follow Shu (2019) to optimize the model based on $\mathcal{S}_{syn}$ (Support Set) and a small set of gold data $\mathcal{S}_{gold}$ (Query Set) obtained from examples. A classification network $f(x, \boldsymbol{\omega})$ is first trained based on the Support set, where $M \in \mathcal{R}^{H \times k+1}$, $k+1$ equals the total number of event types and Null event.

$$f(x) = SoftMax(Encoder(x)M) \tag{2}$$

Next, we use the following loss function to obtain the updated parameters of the classification network $\hat{\boldsymbol{\omega}^t}$ at the time of t:

$$L_{s_{syn}} = \frac{1}{m}\sum_{i=1}^{m} Score(w_i) L^i_{s_{syn}} \tag{3}$$

$$Score(w_i) = \begin{cases} 1, & if\ \exists\ e_{w_i} \\ g(w_i), & if\ \nexists\ e_{w_i} \end{cases} \tag{4}$$

$$L^i_{s_{syn}} = l(y^i_{s_{syn}}, f(s^i_{syn}, \boldsymbol{\omega})) \tag{5}$$

s denotes the length of all tokens in S_{syn} and l denotes cross-entropy loss function. The loss coefficient is determined based on the return value of the score function. If a token does not trigger an event, the coefficient is defaulted to 1. If it is a trigger, the coefficient is calculated based on the weighting model. We gain $\hat{\boldsymbol{\omega}^t}$ as follows:

$$\hat{\boldsymbol{\omega}^t} = \boldsymbol{\omega}^t - \alpha \times \nabla_{\boldsymbol{\omega}^t} L_{s_{syn}} \tag{6}$$

At time $t+1$, we optimize the weighting model on the query set using $\hat{\boldsymbol{\omega}^t}$ (Step-1 in Figure 3). Specifically, we compute the loss on the Query set, with the equations as follows:

$$L_{s_{gold}} = \frac{1}{k}\sum_{i=1}^{k} L^i_{s_{gold}} \tag{7}$$

$$L^i_{s_{gold}} = l(y^i_{s_{gold}}, f(s^i_{gold}, \hat{\boldsymbol{\omega}^t})) \tag{8}$$

k is the length of the Query set tokens. Based on this, we calculate the gradients of the parameters of $g(x, \boldsymbol{\theta})$ and update the parameters from $\boldsymbol{\theta^t}$ to $\boldsymbol{\theta^{t+1}}$ (Step-2 in Figure 3). Then we freeze $\boldsymbol{\theta^{t+1}}$ and calculate the gradients of the classification network's parameters at time $t+1$ according to the loss function in 3. And finally update $\boldsymbol{\omega^t}$ to $\boldsymbol{\omega^{t+1}}$ (Step-3 in Figure 3)

5 Event Detection Model Training

After completing the learning process of the weight model, we use $g(x, \boldsymbol{\theta})$ to calculate the weights for generated data. During training, we merge the weighted synthetic data with the example data, adjust the loss coefficients based on the weights.

6 Experiments

Datasets We perform experiments on three different ED datasets including ACE05 (Walker et al., 2006), MAVEN (Wang et al., 2020) and Commodities News (Lee et al., 2021), which contain 33, 168 and 18 event types respectively. We refer to previous works to partition the test set from ACE05 (Du and Cardie, 2020) and Commodities News (Lee et al., 2021) for evaluating. For MAVEN, we use the officially provided validation set for testing.

Baselines We train a model without weighting on synthetic data and compare it with these baselines in zero-shot setting: UIE(Lu et al., 2022), ZEOP(Zhang et al., 2022b) ZED(Zhang et al., 2022a) , gpt-3.5-turbo and GtR_prompt. Compared with GtR, GtR_prompt omits the *Data Generation* stage and treats the targeted test set as unlabeled generated data (Generated triggers are used in this stage). Then GtR_prompt regards the labeled data obtained through the *Data Revision* stage as final results.

We train a model with weighting and compare the model with these baselines in few-shot setting: UIE (Lu et al., 2022), STAR(Ma et al., 2023a), gpt-3.5-turbo.

Evaluation Strategy We adopt the same evaluation metrics as previous works on ED. For trigger identification (TI), a predicted trigger is considered correct only if it matches the gold trigger exactly. For classification (TC), both triggers and types should be the same with the gold labels. We use F1 score as the evaluation metrice.

Implementation Details We use OpenAI's gpt-3.5-turbo for datasets generation. When constructing event templates, we refer to the official guideline for ACE05, and write event descriptions for other datasets on our own. For fair comparison, we do not use any examples but rely solely on the basic prompt in zero-shot setting. We generate 821 triggers and 6,387 sentences for ACE05, 1,647 triggers and 16,876 sentences for MAVEN, 447 triggers and 6,877 sentences for Commodities News.

We employ the pre-trained model Bert-base-uncased from the Hugging-face community[0] and use the model architecture from Du (2020) for training. We use the Adam optimizer with a learning rate of $4e^{-5}$ and set other hyper-parameters as Du (2020). All models are optimized on a single NVIDIA V100 GPU.

Model	ACE05		MAVEN		Commodities News	
	TI	TC	TI	TC	TI	TC
UIE	-	38.14	-	24.17	-	35.64
ZEOP	-	45.91	-	-	-	-
ZED	-	-	49.39	32.96	-	-
gpt-3.5-turbo	24.63	18.75	25.76	20.32	23.40	19.76
GtR_prompt	43.92	41.93	30.12	28.54	56.23	34.12
STAR	43.74	38.90	-	-	-	-
GtR	55.29	50.49	44.60	36.48	68.79	45.03

Table 1: The zero-shot performance evaluated on three datasets. We adopt the results of UIE on ACE05 from the original paper, and conduct experiments on the other two datasets. For prompting, we employ the same model, gpt-turbo-3.5, perform three rounds of experiments and use the same set of generated triggers. Regarding GtR, we report the average results of three experiments conducted with different random seeds on the same generated data.

[0]https://huggingface.co/bert-base-uncased

6.1 Main Results on Zero-Shot ED

We experimentally evaluate the generative performance of GtR in a zero-shot scenario, where no examples are used, allowing for a direct validation of the framework's generative capabilities. From the table 1, we can draw the following conclusion:

1. The model trained on synthetic data consistently outperforms LLMs and demonstrates a more stable performance. We argue that GtR, through its three-stage generation process and iterative refinement, transforms the challenging task of extracting complex structured events into a series of simpler and more answerable questions, which allows for better and more consistent performance.

2. GtR_prompt performs better than directly prompting, but still falls short of the model trained on GtR. This performance gap may arise from two reasons. Firstly, GtR includes adversarial samples, which can provide additional information and reduce the risk of the model being overly confident. We validate the effectiveness of adversarial samples in subsequent experiments. Secondly, the sentences encoded by the pretrained models contain certain features, such as part-of-speech features, which can enhance ED performance.

3. GtR outperforms UIE, indicating that the advantages gained by UIE through multitask pretraining do not surpass those achieved by synthesizing data using GtR. Additionally, other methods for zero-shot ed using external knowledge and contrastive learning have not shown significant superiority over GtR.

Model	TI		TC	
	5-shot	10-shot	5-shot	10-shot
UIE	-	-	51.21	53.23
gpt-3.5-turbo	34.22	34.64	11.17	12.54
STAR	61.39	63.57	56.41	59.10
GtR	63.47	65.78	58.82	61.21
w/o weighting	61.56	64.21	56.74	58.43

Table 2: Experiment results on few-shot ED

6.2 Main Results on Few-Shot ED

We also conduct experiments in the low-resource scenario of the ACE05 dataset, specifically in 5-shot and 10-shot settings.

1. From table 2, we find that even with an increase in examples, directly prompting LLMs did not result in linear growth in benefits. Performance on both TI and TC are comparatively poor, far behind supervised methods. Finding better prompting methods to improve the performance of LLMs in ED remains a topic worthy of attention.

2. GtR outperforms STAR, validating the superiority of our framework and the necessity of revising data. Compared to STAR, our proposed framework has a higher ceiling. It is noteworthy that both GtR and STAR outperform UIE, once again confirming that data synthesis can quickly improve the performance of ED.

3. Dynamic weighting improves the quality of synthetic data and leads to better performance. This indicates that synthetic data itself contains some noise, which becomes more constraining to the model as the scale increases. It also underscores the necessity of dynamically weighting synthetic data.

6.3 Ablations

Compared to other generation methods, GtR primarily introduces the *Data Revision* step for synthetic data. To explore the improvement in data quality through *Data Revision*, we conduct ablation studies on the generated datasets. As shown in Table 3, we first perform a base experiment on the basic dataset obtained during *Data Generation*. The column *Events* shows the number of events in each dataset. +Rev_old denotes the updated dataset in which only existing labels are revised, +Rev_new denotes the dataset based on +Rev_old in which accompanying events are further identified , and +Rev_com is the dataset based on +Rev_new where competing events are resolved.

	Events	P	R	F1
Basic data	3,980	32.47	24.69	28.05
+Rev_old	-626	36.95	22.72	28.13
+Rev_new	+1,828	52.08	46.42	**49.09**
+Rev_com	-77	58.88	44.20	**50.49**

Table 3: Experiments conducted on different data generated during the revision phase, where the column **Events** indicates the number of events in the current dataset. Every dataset is generated based on the previous dataset.

From Table 3, we observe that model trained on the basic data performs averagely. Basic dataset is filled with noise, which hampers the model's performance. In +Rev_old, the number of events decreases, while the F1 score remains comparable to before with an improvement in precision and decrease in recall. In +Rev_new, the number of events increases around by 50%, and the model's performance significantly improves 74%, approaching optimal results. 77 competing events are eliminated in +Rev_com, leading to a increase in precision, and yet a slightly drop in recall.

In general, the key to improving the model performance lies in the extra events identified in the *Data Revision* stage.

6.4 Analysis on Adversarial Samples

Setting	Models	F1	$R_{error} \downarrow$
zero-shot	GtR	50.49	**7**
	w/o Adv	48.68	53
	UIE-base	38.14	63.5
fully supervised	Bert-ED	**70.17**	10.5

Table 4: The error rates of different models on adversarial test data, where w/o ADV indicates the model trained without adversarial samples and Bert-ED is trained on all gold samples.

We generate adversarial samples for each trigger in *Data Generation*. Previous studies have shown that incorporating adversarial samples during training can improve the robustness of models. To evaluate the effectiveness, we conduct experiments on a brand-new test set for ACE05. The detailed process for data construction is as follows:

We first select the common triggers between the gold training dataset and $\mathcal{D}_{mt}$ to ensure a fair comparison. Then, with these triggers, we create 200 adversarial samples as test set. Each sample is manually inspected to ensure that neither does it contain any events nor appear in $\mathcal{D}_{mt}$. We use the error rate R_{error} as the evaluation metric, which is the proportion of incorrectly predicted samples out of all the samples.

Table 4 presents the experiments performance of different models on this test set. We find that removing the adversarial samples slightly compromises the overall performance of GtR (a 2% drop) and significantly reduces its robustness. UIE, which is not deliberately trained with adversarial samples, also exhibits high error rates. GtR trained with adversarial samples (7% R_{error}), and Bert-ED trained on the full training set with supervision (10.5% R_{error}) demonstrate much lower error rates. This finding validates the practical utility of generating adversarial data to enhance the model robustness.

6.5 Analysis on Weighting Model

This section analyzes the practical effects of the dynamic weighting. To do this, we first conduct a statistical analysis of the scores for each event in the ACE05 synthetic data. The horizontal axis of the graph represents the score range from 0.35 to 0.5, while the vertical axis represents the number of events corresponding to each range in the ACE05 synthetic dataset.

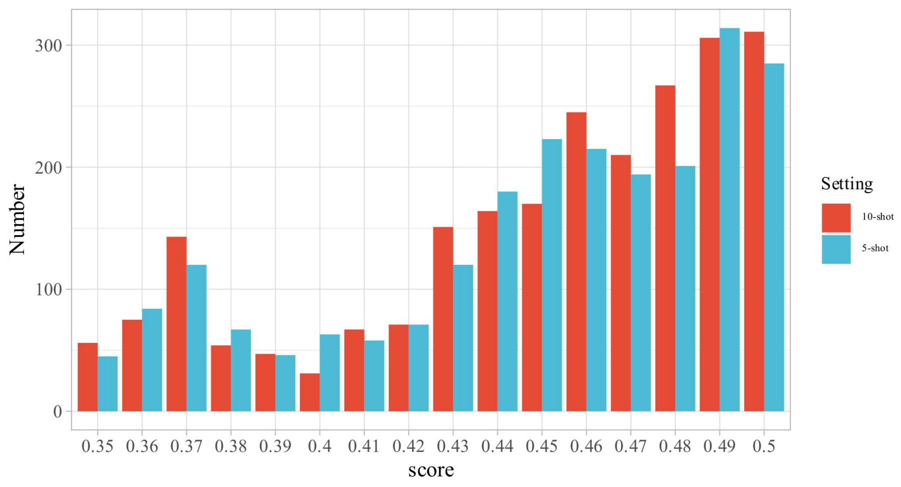

Figure 4: Illustration of the weights of events.

From the figure 4, it can be observed that the majority of synthetic data scores are concentrated in the range of 0.44 to 0.5, with some in the range of 0.35 to 0.38, and the remaining data dispersed in other score ranges. Under the 10-shot scenario, the distribution of synthetic data tends to skew slightly to the right compared to the 5-shot scenario.

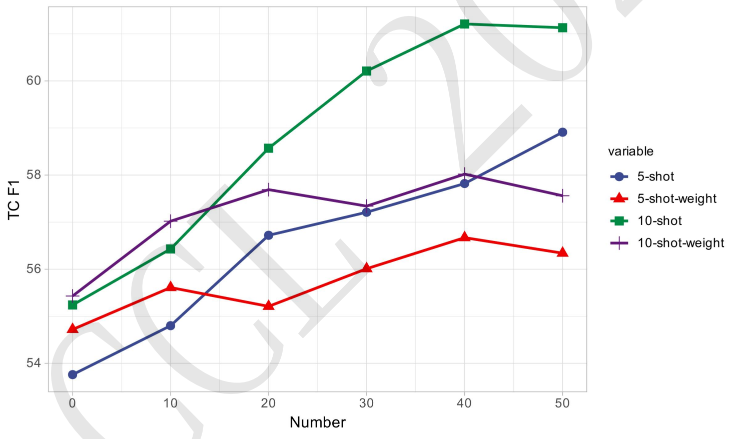

Figure 5: Illustration of the performance of models in different setting.

In Figure 5, we present a comparison of performance with and without the weighting module as the scale of synthetic data increases in both the 5-shot and 10-shot scenarios. The horizontal axis represents the number of synthetic data added for each event type, while the vertical axis represents the F1 score on TC. **5-shot-weight** and **10-shot-weight** denote the performance of models without incorporating the weighting model.

Figure 5 demonstrates that in both the 5-shot and 10-shot scenarios, models trained on dynamically weighted synthetic data outperform those trained directly on synthetic data. Under the 5-shot condition, as the data scale increases, the performance of the regular model reaches a peak, showing significant fluctuations and even slight declines. However, models trained with dynamic weighting still maintain a growth trend.

These results confirms the importance of the dynamic weighting module in improving data quality and the potential value it brings in enhancing model performance and stability.

	Self-Bleu	Accuracy
Gold	0.18	-
GtR	0.17	88.3

Table 5: Diversity and accuracy metrics of synthetic data

6.6 Analysis on Synthetic Data

The ablation experiment validates the role of synthetic data obtained during the data generation stage. Here, we further analyze the diversity and reliability of the synthetic data. The evaluation metric for diversity is the Self-Bleu score. We calculated the Self-Bleu scores for ACE05 gold training data and synthetic data on a randomly selected set of 300 samples. Self-Bleu values can to some extent reflect the diversity of data. It can be observed that the diversity of synthetic data is comparable to that of real data.

We manually check the accuracy of 200 adversarial data samples. Table 5 shows that the accuracy of adversarial data is relatively high, indicating that the model is proficient in synthesizing adversarial data.

6.7 Case Study

Event Types	Triggers	Data Types	Sentences
Attack	bashed	normal	The reckless driver **bashed** into the cyclist, injuring him badly and causing damage to his bike.
Attack	seize	adversarial	The company decided to **seize** the opportunity to expand business when a rival company had to shut down due to bankruptcy
Be-Born	deliver	normal	The pregnant woman was rushed to the emergency room as she went into labor and had to **deliver** her premature baby.
Be-Born	born	adversarial	The idea for the new product was **born** during a brainstorming session with the marketing team.

Table 6: Generated data cases on ACE05. The bold font indicates the correct events within basic data, the green words represent the newly recognized events and the red word suggest an incorrect event.

The section presents some successful and failed cases in the synthetic data, along with the results of the weight model. As shown in Table 6 if the event type is specific and easily distinguishable from other event types, GtR tends to synthesize successful and high-quality data, as exemplified by the high correctness of the synthetic data for the **Be-Born** event shown in the table. However, even after data revision, synthetic data may still contain incorrectly recognized events, such as the event in the first case where the victim of **Attack** should be a person.

Addressing the effectiveness of data synthesis on abstract event types and proposing better revision or filtering algorithms are topics worthy of future research. We leave the work for future research.

7 Conclusion

ED has long faced challenges of data scarcity and the time-consuming nature of manual annotation. The development of LLMs brings a ray of hope to address these issues. In this paper, we proposes GtR, the data synthesis framework, which includes three stages: trigger expansion, data generation, and data revision, along with a dynamic weighting module for synthetic data. In the experimental section, we test the generative performance of GtR in zero-shot and low-shot scenarios. Compared to direct prompting and other generation methods, our framework exhibits significant advantages, with the dynamic weighting mechanism improving the model's performance ceiling. We hope that the research in this paper can inspire future work in the field of data synthesis.

References

Xinya Du and Claire Cardie. 2020. Event Extraction by Answering (Almost) Natural Questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online, November. Association for Computational Linguistics.

Jun Gao, Changlong Yu, Wei Wang, Huan Zhao, and Ruifeng Xu. 2022. Mask-then-Fill: A Flexible and Effective Data Augmentation Framework for Event Extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4537–4544, Abu Dhabi, United Arab Emirates, December. Association for Computational Linguistics.

Jiahui Gao, Renjie Pi, Yong Lin, Hang Xu, Jiacheng Ye, Zhiyong Wu, Weizhong Zhang, Xiaodan Liang, Zhenguo Li, and Lingpeng Kong. 2023. Self-Guided Noise-Free Data Generation for Efficient Zero-Shot Learning, February. arXiv:2205.12679 [cs].

Ridong Han, Tao Peng, Chaohao Yang, Benyou Wang, Lu Liu, and Xiang Wan. 2023. Is Information Extraction Solved by ChatGPT? An Analysis of Performance, Evaluation Criteria, Robustness and Errors, May. arXiv:2305.14450 [cs].

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, July. Association for Computational Linguistics.

Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-Shot Transfer Learning for Event Extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2160–2170, Melbourne, Australia, July. Association for Computational Linguistics.

Kuan-Hao Huang, I.-Hung Hsu, Tanmay Parekh, Zhiyu Xie, Zixuan Zhang, Premkumar Natarajan, Kai-Wei Chang, Nanyun Peng, and Heng Ji. 2024. TextEE: Benchmark, Reevaluation, Reflections, and Future Challenges in Event Extraction, February. arXiv:2311.09562 [cs].

Martin Josifoski, Marija Sakota, Maxime Peyrard, and Robert West. 2023. Exploiting Asymmetry for Synthetic Training Data Generation: SynthIE and the Case of Information Extraction, March. arXiv:2303.04132 [cs].

Meisin Lee, Lay-Ki Soon, Eu-Gene Siew, and Ly Fie Sugianto. 2021. An Annotated Commodity News Corpus for Event Extraction, August. arXiv:2105.08214 [cs].

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, April. arXiv:2304.11633 [cs].

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified Structure Generation for Universal Information Extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772, Dublin, Ireland, May. Association for Computational Linguistics.

Mingyu Derek Ma, Xiaoxuan Wang, Po-Nien Kung, P. Jeffrey Brantingham, Nanyun Peng, and Wei Wang. 2023a. STAR: Boosting Low-Resource Event Extraction by Structure-to-Text Data Generation with Large Language Models, May. arXiv:2305.15090 [cs].

Yubo Ma, Yixin Cao, YongChing Hong, and Aixin Sun. 2023b. Large Language Model Is Not a Good Few-shot Information Extractor, but a Good Reranker for Hard Samples!, March. arXiv:2303.08559 [cs] version: 1.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is ChatGPT a General-Purpose Natural Language Processing Task Solver?, February. arXiv:2302.06476 [cs].

Timo Schick and Hinrich Schütze. 2021. Generating Datasets with Pretrained Language Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6943–6951, Online and Punta Cana, Dominican Republic, November. Association for Computational Linguistics.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-Weight-Net: Learning an Explicit Mapping For Sample Weighting, September. arXiv:1902.07379 [cs, stat].

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. ACE 2005 Multilingual Training Corpus, February. Artwork Size: 1572864 KB Pages: 1572864 KB.

Xiaozhi Wang, Ziqi Wang, Xu Han, Wangyi Jiang, Rong Han, Zhiyuan Liu, Juanzi Li, Peng Li, Yankai Lin, and Jie Zhou. 2020. MAVEN: A Massive General Domain Event Detection Dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1652–1671, Online, November. Association for Computational Linguistics.

Jiacheng Ye, Jiahui Gao, Qintong Li, Hang Xu, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022a. ZeroGen: Efficient Zero-shot Learning via Dataset Generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11653–11669, Abu Dhabi, United Arab Emirates, December. Association for Computational Linguistics.

Jiacheng Ye, Jiahui Gao, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2022b. ProGen: Progressive Zero-shot Dataset Generation via In-context Feedback. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3671–3683, Abu Dhabi, United Arab Emirates, December. Association for Computational Linguistics.

Hongming Zhang, Wenlin Yao, and Dong Yu. 2022a. Efficient Zero-shot Event Extraction with Context-Definition Alignment. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7169–7179, Abu Dhabi, United Arab Emirates, December. Association for Computational Linguistics.

Senhui Zhang, Tao Ji, Wendi Ji, and Xiaoling Wang. 2022b. Zero-Shot Event Detection Based on Ordered Contrastive Learning and Prompt-Based Prediction. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2572–2580, Seattle, United States, July. Association for Computational Linguistics.