

Event Extraction Based on Self-Data Augmentation with Large Language Models

Lishan Yang

ylishan@mail.ustc.edu.cn

University of Science and Technology of China

Xi Fan

University of Science and Technology of China

Xiangyu Wang

University of Science and Technology of China

Xin Wang

University of Science and Technology of China

Qiuju Chen

University of Science and Technology of China

Research Article

Keywords: Event extraction, Data augmentation, Large language models, Logical Thoughts for Self-Data Augmentation (LoTSA)

Posted Date: June 3rd, 2024

DOI: <https://doi.org/10.21203/rs.3.rs-4444035/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at Memetic Computing on January 31st, 2025. See the published version at <https://doi.org/10.1007/s12293-025-00436-8>.

Event Extraction Based on Self-Data Augmentation with Large Language Models

Lishan Yang¹, Xi Fan¹, Xiangyu Wang¹, Xin Wang¹, Qiuju Chen²

¹*School of Computer Science and Technology, University of Science and Technology of China, Hefei, China.

²*School of Information Science and Technology, University of Science and Technology of China, Hefei, China.

Contributing authors: ylshan@mail.ustc.edu.cn; sa22011218@mail.ustc.edu.cn; sa312@ustc.edu.cn; wz520@mail.ustc.edu.cn; qqchern@ustc.edu.cn;

Abstract

Event extraction plays a crucial role in natural language processing (NLP), facilitating the transformation of unstructured text into structured representations. This conversion significantly enhances the performance of various applications, such as automated question answering and information retrieval systems. However, traditional event extraction methodologies often encounter challenges stemming from limited datasets, imbalanced sample distributions, the necessity for extra resources to annotate large datasets, and the potential for data quality degradation during the augmentation process. To surmount these obstacles, this study introduces an innovative self-data augmentation strategy that leverages a single large language model (LLM) to concurrently perform data augmentation and event extraction. By dynamically assessing and refining the quality of generated samples, this approach mitigates the inclusion of noisy data, ultimately bolstering the model’s performance. Demonstrable enhancements in precision, recall, and F1 scores across various model configurations underscore the efficacy of this strategy in managing small and imbalanced datasets. Furthermore, the incorporation of Logical Thoughts for Self-Data Augmentation (LoTSA) ensures the superior quality of augmented data, culminating in more accurate and reliable extraction outcomes.

Keywords: Event extraction, Data augmentation, Large language models, Logical Thoughts for Self-Data Augmentation (LoTSA)

1 Introduction

Event extraction, a critical task in the realm of Natural Language Processing (NLP), aims to transform unstructured text that contains information about events into a structured format [1], which is pivotal for applications such as automated question answering and information retrieval. Despite the considerable attention this area has received from a wide array of research

institutions and scholars [2, 3], the performance of event extraction systems in practical settings often falls short of expectations.

In practical deployments, several challenges impede the efficacy of event extraction systems. Primarily, the standard datasets available in most domains are limited in scale and lack comprehensive domain-specific textual resources [4, 5]. The training of robust automated information extraction models generally depends on access to

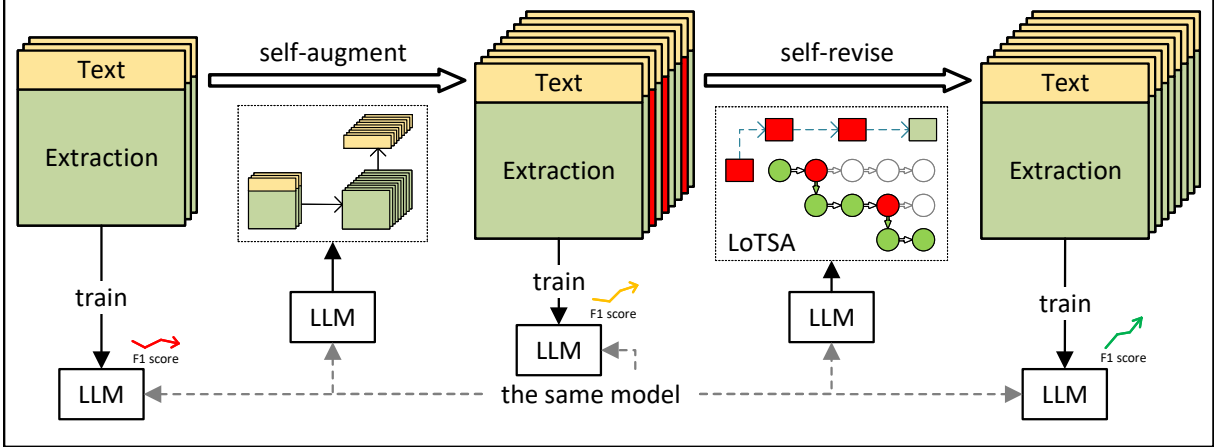


Fig. 1: Overview of the Self-Data Augmentation Strategy for Event Extraction Utilizing a Single Large Language Model. This diagram illustrates the integrated process of data augmentation and event extraction, highlighting the dynamic revision of augmented samples to enhance model performance.

extensive textual data. Moreover, acquiring high-quality annotated texts remains a formidable challenge, as traditional dataset construction methods demand significant annotation efforts. Additionally, the distribution of event instances in real-world data is frequently imbalanced, with some categories of events being underrepresented [6, 7]. These complications pose substantial hurdles for traditional machine learning frameworks when applied to event extraction tasks.

To mitigate these challenges, various data augmentation techniques have been introduced. These techniques commonly incorporate additional steps into the event extraction workflow, such as the integration of specialized data augmentation models or the inclusion of augmentation modules within existing frameworks [8]. While these strategies effectively enhance the diversity of event texts and improve sample balance, they also introduce significant overheads in terms of model development and design, as well as increased computational demands [5]. In the context of the contemporary NLP landscape, which is predominantly influenced by large language models, several approaches have adopted optimized large language models specifically for data augmentation, separate from the primary event extraction model [9–12]. This adaptation adds layers of complexity to the event extraction workflow and escalates the requirements for computational resources and operational costs.

Furthermore, potential noise from imprecise augmentation techniques can adversely affect the training of models.

Accordingly, this paper introduces a novel self-data augmentation based event extraction strategy that concentrates on the dual objectives of augmenting event data and extracting event information using a singular model. During the event data augmentation phase, the quality of the generated samples is dynamically evaluated and refined through the Logical Thoughts for Self-Data Augmentation (LoTSA) strategy. This systematic revision process effectively improves sample quality, thereby minimizing the detrimental effects of low-quality and noisy data on model training. Figure 1 depicts the cohesive process of the self-data augmentation strategy. In detail, the proposed strategy employs a large language model to alter the event extraction results in the training dataset, followed by leveraging the same model for generating reconstructed textual content. The quality of these newly generated event-text pairs is continuously monitored and adjusted, which facilitates the creation of a diverse and high-caliber dataset. This, in turn, significantly enhances the model’s performance in executing the event extraction task.

The contributions of this paper are summarized as follows:

- The proposed strategy facilitates event sample augmentation and expansion utilizing a

single model, thereby circumventing the additional consumption of computational resources or application costs typically associated with supplementary data augmentation processes.

- The strategy incorporates a dynamic self-revision mechanism for enhancing the quality and reliability of the generated samples during the data augmentation process, effectively mitigating the adverse impacts of noise of the augmented data on model training.
- The proposed strategy demonstrates versatility across different model frameworks and contributes to a reduction in the model’s parameter requirements without compromising performance.

The remainder of the paper is organized as follows: Section II reviews related work in the field of data augmentation, with a particular focus on the shift towards employing large language models in NLP. Section III elaborates on the methodologies employed, including data augmentation using large language models and the innovative concept of Reverse Data Augmentation. Section IV details the experimental setup and presents the findings, underscoring the principal outcomes from the model evaluations. Section V concludes the paper.

2 Related work

Data augmentation is a machine learning technique that artificially expands the volume of training data through transformations that preserve labels [13]. An illustrative early application of this technique is evident in the renowned LeNet model [14], where random distortions applied to training images expanded the MNIST dataset to nine times its original size, significantly improving the recognition of handwritten digits. The concept of “label preservation” is crucial here, referring to the maintenance of category information unchanged during the transformation of training data [15].

In the realm of Natural Language Processing (NLP), data augmentation poses a notable challenge, as defining text transformations that effectively preserve labels is complex. To address this, various methods have been explored. These methods range from word swapping [8], deletion [16, 17], the introduction of spelling errors [15, 18], and paraphrasing [19], to synonym replacement [20–23], utilizing close embeddings [24–

26], and employing language models for word-level prediction [27–30]. More advanced techniques include altering dependency trees [31–34], performing round-trip translation [35–38], or inserting existing input instances [39–42]. Additionally, further research has delved into text generation techniques for data augmentation, with studies employing recurrent neural networks and generative adversarial networks to augment short texts [43–46], whereas others have utilized variational autoencoders to generate text instances without length limitations [17].

In recent years, the rapid advancement of large language models (LLMs) has catalyzed their application in data augmentation, emerging as a focal point of research within the field. In the domain of natural language processing, two promising data augmentation strategies leveraging pretrained language models have been identified [47–50]. The first strategy entails the fine-tuning of language models for specific tasks, followed by utilizing these tailored models to generate novel data samples [51–55]. A notable limitation of this approach is the frequent scarcity of task-specific data, which can impede the fine-tuning process. Conversely, the second strategy employs existing pretrained models, such as GPT-3, thereby circumventing the fine-tuning bottleneck [56]. This method enables the direct generation of text without necessitating task-specific adjustments. For instance, Yoo et al. employed the GPT3Mix technique to significantly enhance the performance across several classification tasks [57], although this method is predominantly effective for tasks with limited categories (up to six) and distinct category boundaries (e.g., positive and negative scenarios). Recently, Kalpakchi and Boye introduced an innovative approach, utilizing GPT-3 to generate supplementary training data specifically for multiple-choice questions in Sweden [58]. Building on this and similar advancements, Samuel et al. [59] developed a GPT-4 based data augmentation technique tailored for low-resource extractive machine reading comprehension tasks. Their method not only generates synthetic contexts, questions, and answers but also employs cycle-consistent filtering to ensure the integrity of these additions. This strategy has demonstrated significant improvements in BERT-based performance on datasets such as CovidQA and PolicyQA, underscoring the potent utility of large

language models in enhancing data augmentation across diverse NLP tasks.

3 Method

3.1 Preliminaries

This subsection outlines the fundamental components and procedures involved in the data augmentation process utilizing a large language model, followed by the training of an enhanced model using this augmented dataset. The methodology for generating augmented data is articulated, involving a series of meticulously crafted prompts and filters applied through a pretrained language model. This augmented dataset is subsequently utilized to refine and train a novel model, thereby potentially augmenting its proficiency in handling more complex or diverse data patterns. This approach exemplifies a strategic utilization of existing datasets and models to systematically bootstrap models with improved capabilities.

3.1.1 Data Augmentation

In the formalization of the data augmentation process utilizing a large language model, several pivotal components are identified: the original datasets, the language model, the sequence of prompts, the sequence of filters, and the augmented results. The original training dataset is represented as $D_{\text{train}} = \{(x_i, y_i)\}_{i=1}^N$, where x_i denotes the input features, y_i represents the corresponding labels, and N is the total number of samples. Similarly, the test dataset is denoted by $D_{\text{test}} = \{(x_i, y_i)\}_{i=1}^M$.

A pre-trained large language model, such as Qwen1.5-72B-Chat [60], is denoted by L . The output of the model in response to a prompt P is expressed as $L(P)$. A sequence of prompt functions $\{p_1(x, y), p_2(x), \dots, p_k(x)\}$ and a corresponding sequence of filter functions $\{f_1(a_1), f_2(a_1, a_2), \dots, f_k(a_1, a_2, \dots, a_k)\}$ are applied to the outputs of the language model. These functions are utilized to generate and refine new, related data points. The entire process is defined recursively as follows:

$$\text{ans}_{i,j}^t = \begin{cases} L(p_1(x_i, y_i)) & \text{if } j = 1 \\ L(p_j(f_{j-1}(\text{ans}_{i,1}^t, \dots, \text{ans}_{i,j-1}^t))) & \text{if } j > 1 \end{cases}, \quad (1)$$

where $\text{ans}_{i,j}^t$ is the result after applying the j -th prompt and the $(j-1)$ -th filter to the i -th training sample during the t -th iteration through the training dataset.

The augmented data point for each original sample is generated by applying the final filter:

$$\{(x_i^t, y_i^t)\} = \{f_k(\text{ans}_{i,1}^t, \dots, \text{ans}_{i,k}^t)\}. \quad (2)$$

The data augmentation process encompasses the following procedural steps:

1. Constructing a series of nested prompts and filters tailored to the features and labels of the training dataset.
2. Employing the language model L and applying the devised prompts and filters recursively, to generate new samples corresponding to each original sample (x_i^t, y_i^t) in the dataset.
3. Integrating these newly generated samples back into the original training dataset, thereby forming an enriched augmented training dataset:

$$D_{\text{train.aug}} = D_{\text{train}} \cup \bigcup_{i=1}^N \bigcup_{t=1}^{N_T} \{(x_i^t, y_i^t)\}, \quad (3)$$

where N_T is the number of iterations, and due to the stochastic nature of the large language model, the answers are different in each iteration. This results in $N \times N_T$ augmented samples, which combined with the original samples, result in a total of $N \times N_T + N$ samples.

Additionally, for simplicity, it is assumed that all prompts $p_1, p_2, \dots, p_k$ utilized across the N_T iterations are identical. In reality, these prompts may vary with each iteration to enhance the randomness and effectiveness of the augmentation process. Within this framework, the input parameters for each prompt p_i (where $i = 1, \dots, k$) can extend beyond x and y to include an additional parameter c , commonly integrated into structured data c_j^t where $j = 1, \dots, k$ and $t = 1, \dots, N_T$, varying with each training sample and iteration. Consequently, the prompts can be represented as $\{p_1(x, y, c), p_2(x, c), \dots, p_k(x, c)\}$. For brevity, subsequent discussions will omit the parameter c unless specified otherwise.

3.1.2 Model Self-Data Augmentation

Upon preparation of the augmented training dataset $D_{\text{train\_aug}}$, it becomes feasible to train a new large language model, denoted as L' . The training process is delineated as follows:

1. Initialize the model L' with or without pre-training.
2. Define the loss function $\mathcal{L}(\theta)$ as the cross-entropy loss, which is commonly employed for classification tasks typical of large language models. The cross-entropy loss for a model outputting a probability distribution p over classes, given the true label distribution q , is defined as:

$$\mathcal{L}(\theta) = - \sum_{i=1}^N \sum_{c=1}^C q_{ic} \log p_{ic}(\theta), \quad (4)$$

where N represents the number of examples in $D_{\text{train\_aug}}$, C denotes the number of classes, q_{ic} is an indicator variable (1 if class label c is the correct classification for example i , and 0 otherwise), and $p_{ic}(\theta)$ is the probability predicted by the model L' for class c for example i , parameterized by θ .

3. Train L' by updating the parameters θ using stochastic gradient descent (SGD) or an analogous optimization algorithm. The parameter update at each iteration t is performed as follows:

$$\theta_{t+1} = \theta_t - \eta_t \nabla \mathcal{L}(\theta_t), \quad (5)$$

where η_t is the learning rate at iteration t , and $\nabla \mathcal{L}(\theta_t)$ is the gradient of the loss function with respect to the parameters at θ_t . Each training sample (x_i^t, y_i^t) from $D_{\text{train\_aug}}$ is employed to compute these gradients.

Upon successful training, model L' is anticipated to demonstrate enhanced performance owing to the diverse and expanded training data provided by the augmentation process. If the model used for data augmentation (L) and the model subsequently trained on the augmented data (L') are the same, this process is termed model self-data augmentation. In this scenario, the model iteratively refines its own training data and retrains itself on this enriched dataset. This strategy can potentially initiate a virtuous cycle where the model continuously improves through

iterative refinements and training sessions, significantly enhancing the model's performance, particularly in scenarios where additional real training data is scarce or costly to procure. However, it is crucial to mitigate risks such as overfitting to the augmented data or compounding errors from repeated data generations.

3.1.3 Reverse Data Augmentation

Reverse Data Augmentation is a process that leverages the generative capabilities of large language models, as adopted by recent data augmentation studies [61–63]. This method represents a reversal of the conventional event extraction approach, aimed at enriching the dataset. Specifically, the process entails modifying the extracted event information y_i from the training dataset D_{train} using a large language model L prompted with specific instructions p'_1 . The output is subsequently filtered through f'_1 to ensure both quality and relevance, producing varied versions $\hat{y}_i^t = f'_1(\text{ans}_{i,1}^{t'}) = f'_1(L(p'_1(\text{None}, y_i^t)))$ for $t = 1, \dots, N_T$. Following the modification of event extractions, the model L is again utilized, this time with prompt p'_2 , to reconstruct the text segment $\hat{x}_i^t$ that corresponds to the newly modified event extractions $\hat{y}_i$. This reconstructed text is then processed through f'_2 to ensure alignment and accuracy, resulting in $(\hat{x}_i, \hat{y}_i) = f'_2(\text{ans}_{i,1}^{t'}, \text{ans}_{i,2}^{t'})$.

The primary advantage of this method is its potential to disrupt the pattern of the model merely relearning its previous outputs. By inverting the usual data processing flow from generation to extraction, Reverse Data Augmentation encourages the model to explore novel narrative and structural possibilities within the data, thus potentially enhancing its generalization capabilities on unseen data.

3.2 Overview of Logical Thoughts (LoT)

Inspired by Chain-of-Thought (CoT) [64], Logical Thoughts (LoT) [65] represents a conceptual framework that integrates human-like logical reasoning into artificial intelligence systems. This approach aims to enhance AI capabilities in decision-making and problem-solving by simulating complex logical processes typically utilized by humans.

The methodology underpinning LoT is designed to closely mimic human logical thinking within AI models. It commences with the representation of logic in a format that is computationally manageable. This often entails encoding logic using symbolic representations or predicate logic, expressed as:

$$\text{Logic}(x) := \forall x(P(x) \rightarrow Q(x)), \quad (6)$$

where $P(x)$ and $Q(x)$ signify logical predicates associated with the properties of x . These representations are fundamental as they facilitate the manipulation of logical statements and the algorithmic application of logical rules.

At the core of LoT is a logical inference mechanism, enabling AI systems to derive conclusions from a set of premises and rules. This can be realized through diverse techniques, such as rule-based systems, where rules are systematically applied to the premises to infer conclusions, or through logic programming which employs a set of logical sentences to articulate facts and rules about a specific problem domain. An example of a logical operation in this context might be:

$$\text{if } P(a) \text{ and } (P(x) \rightarrow Q(x)) \text{ then } Q(a), \quad (7)$$

illustrating the application of modus ponens, a fundamental rule of inference in logic.

To ensure that the logical models developed under LoT are dynamic and adaptable, these logical elements are integrated with conventional machine learning algorithms. This integration allows the AI to continually refine and evolve its reasoning capabilities based on new data, thereby learning to reason more effectively over time.

The applications of LoT are extensive and significant, particularly in areas requiring nuanced decision-making such as legal reasoning, financial analysis, and strategic planning, where LoT can substantially enhance the depth and reliability of insights generated by AI systems. Additionally, in the domain of natural language processing, LoT improves the model's comprehension of context and its ability to generate responses that are both coherent and logically consistent.

3.3 Self-Data Augmentation Strategy

In the previous section, the concepts of Reverse Data Augmentation and Logical Thoughts (LoT) were examined. These concepts are now implemented in the Model Self-Data Augmentation process for event extraction tasks. The primary objective of this process is to leverage the generative capabilities of model L to enhance its performance in these tasks.

Initially, the prompt p'_1 used to modify the event extractions y_i is defined as follows:

Prompt $p'_1(\text{None}, y, c)$

%EVENT_EXTRACTION%

The content described above involves extracting several events from a text segment in JSON format. Please perform the rewriting extraction according to the following requirements:

1. Maintain the original JSON format. The fields of the JSON Object must not be changed, but the size of the JSON Array can be modified.
2. Change the event_type to %EVENT_TYPE%.
3. Based on the event_type, modify the subject and object. Note that both subject and object must remain consistent throughout the JSON, although their positions can be swapped.
4. Modify the action, time, and location based on the event_type, subject, and object. Ensure that the time and location are reasonable.
5. The modified JSON must depict a complete scene, where each event is a part of this scene.
6. Please output in JSON format, ensuring no extraneous content is included.

Based on prompt p'_1 , model L processes the original event extractions y_i by replacing %EVENT_EXTRACTION% with y_i and %EVENT_TYPE% with c_i^t to generate $\text{ans}_{i,1}^{t'}$. Then, using filter f'_1 , the output is formatted into the correct JSON format, such that $\hat{y}_i^t = f'_1(\text{ans}_{i,1}^{t'})$. Subsequently, prompt p'_2 is defined,

which is used to reconstruct the text segment x_i from these newly formatted event extractions $\hat{y}_i^t$:

Prompt $p'_2(x)$

%EVENT_EXTRACTION%

The content described above involves extracting several events from a text segment in JSON format. Please follow the requirements below to reconstruct the original text segment:

1. The reconstructed text must include all the information from the JSON, with nothing omitted.
2. Each piece of information from the JSON must appear completely in the text segment without being truncated or interrupted.
3. The length of the reconstructed text segment must be over 200 characters. You may include some sentences that do not describe the events to meet the word count.
4. The reconstructed text segment must be semantically coherent and complete, ensuring that the entire text segment is within a consistent scene.
5. Please output the reconstructed text segment only, without any extraneous content.

Using p'_2 , the large language model L generates $\text{ans}_{i,2}^k$, where **%EVENT_EXTRACTION%** is replaced with $\hat{y}_i^t$. Then, using filter f'_2 , non-textual parts in $\text{ans}_{i,2}^t$ are filtered out to obtain $\hat{x}_i^t$. Thus, augmented data $\{(\hat{x}_i^t, \hat{y}_i^t)\}$ is formed, such that $(\hat{x}_i, \hat{y}_i) = f'_2(\text{ans}_{i,1}^t, \text{ans}_{i,2}^t)$.

Upon the generation of the augmented data, a modified methodology derived from Logical Thoughts, termed Logical Thoughts for Self-Data Augmentation (LoTSA), is employed to ensure the integrity and accuracy of the generated data. The LoTSA process involves a systematic evaluation and revision through four distinct prompts, encompassing five processes. Each sample is assessed using the first three of these prompts to ascertain the necessity of any corrections for the respective processes. Should corrections be required, the fourth prompt is specifically utilized to make adjustments to the training sample.

Figure 2 illustrates the framework used in the LoTSA methodology. This visual representation aids in understanding how the LoTSA prompts interact with the event data to ensure that it meets the set quality standards.

The initial prompt, denoted as p_{L1} , is crafted to evaluate the correctness and completeness of the extracted events, based on a set of predefined questions.

Prompt $p_{L1}(x, y, c)$

%DATA%

The content described above consists of several events extracted from a text segment and compiled into a JSON file. Please think critically about the following questions and provide your reasoning and answer. If the final answer is “yes”, please conclude your thoughts with “yes”; if “no”, conclude with “no”: **%QUESTION%**

The **%DATA%** is replaced with the augmented data $\{(\hat{x}_i^t, \hat{y}_i^t)\}$ revised through previous processes, and **%QUESTION%** is replaced with one of the following questions in sequence for each process:

1. Does the “event_list” omit any important information from the “text”?
2. Does the “event_list” contain any redundant or duplicate information?
3. Does the “event_list” accurately represent the event class?
4. Do the “actor”, “action”, “object”, “time”, and “location” in the “event_list” match the descriptions in the original “text”?
5. Are the “actor”, “action”, “object”, “time”, and “location” in the “event_list” exact transcriptions from the original text segment?

After addressing each of the five questions mentioned earlier, judgments are made using the following prompt, denoted as p_{L2} :

Prompt $p_{L2}(x, y, c)$

%DATA%

The content described above comprises several events extracted from a text segment and compiled into a JSON file. Based on the judgment, **%STATEMENT%**

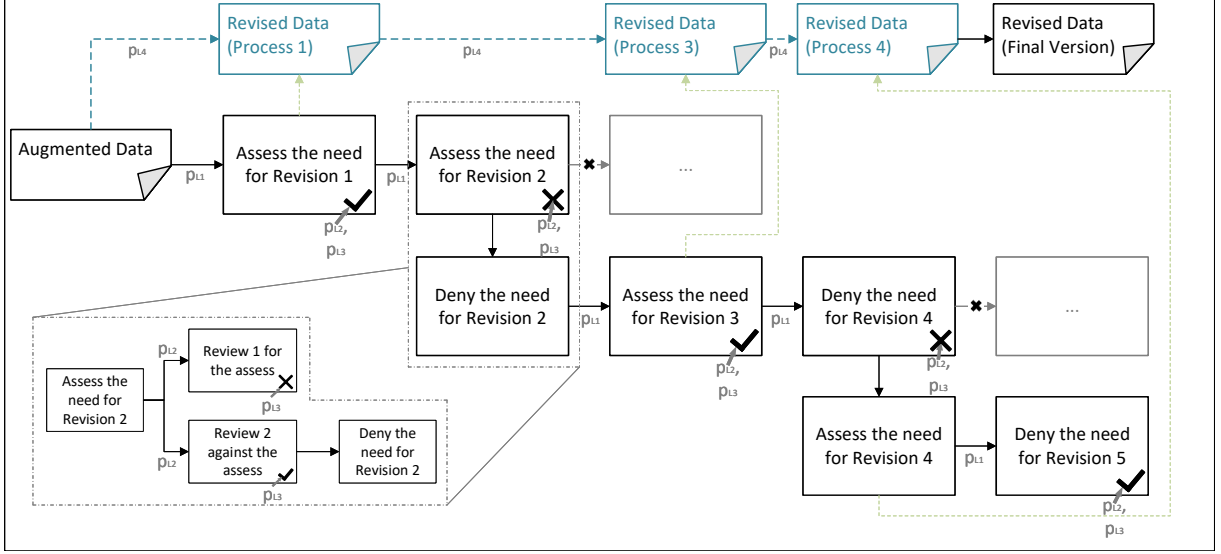


Fig. 2: Framework of the Logical Thoughts for Self-Data Augmentation (LoTSA) methodology applied in the revision of the augmented dataset.

This judgment is %TF%, for which please provide your reasoning.

The %DATA% is replaced with the augmented data $\{(\hat{x}_i^t, \hat{y}_i^t)\}$ revised through previous processes, and %STATEMENT% is replaced based on the conclusion from p_{L1} for each of the scenarios:

1. “The event list does not omit any important information from the text”, or “The event list omits important information from the text”.
2. “The event list contains no redundant or duplicate information”, or “The event list contains redundant or duplicate information”.
3. “The event list accurately represents the event class”, or “The event list does not accurately represent the event class”.
4. “The actors, actions, objects, times, and locations in the event list match the descriptions in the original text”, or “The actors, actions, objects, times, and locations in the event list do not match the descriptions in the original text”.
5. “The actors, actions, objects, times, and locations in the event list are exact transcriptions from the original text segment”, or “The actors, actions, objects, times, and locations in the event list are not exact transcriptions from the original text segment”.

The %TF% is replaced with “True” or “False”, leading to two questions querying the large language model.

Subsequently, these two judgments are verified using the following template p_{L3} :

Prompt $p_{L3}(x, y, c)$

%DATA%

The content described above comprises several events extracted from a text segment and compiled into a JSON file. Based on the judgment, %STATEMENT%

Below are two reviews for this judgment: Review1: %REVIEW1% Review2: %REVIEW2%

Please critique the incorrect judgment and praise the correct judgment. If Review1 is correct, end your answer with the characters REVIEW1. Otherwise, end with REVIEW2.

The %DATA% is replaced with the augmented data $\{(\hat{x}_i^t, \hat{y}_i^t)\}$ revised through previous processes, %STATEMENT% is what was used in p_{L2} , %REVIEW1% is the response when %TF% is “True”, and %REVIEW2% is the response when %TF% is “False”.

If p_{L1} concludes there is an issue and p_{L3} confirms Review1 is correct, or if p_{L2} concludes

there is no issue and p_{L3} confirms Review2 is correct, then corrections to $\{(\hat{x}_i^t, \hat{y}_i^t)\}$ are necessary through the process p_{L4} .

Prompt $p_{L4}(x, y, c)$

%DATA%

The content described above consists of several events extracted from a text segment and compiled into a JSON file. It has been found to contain the following issues: %PROBLEM% Please address this issue and output the corrected events and original text in a JSON file, without including any extraneous content.

The %DATA% is replaced with the problematic augmented data $\{(\hat{x}_i^t, \hat{y}_i^t)\}$ revised through previous processes, and %PROBLEM% corresponds to one of the identified issues based on the five questions:

1. “event_list” omits important information from “text”.
2. “event_list” contains redundant or duplicate information.
3. “event_list” does not accurately represent the event class.
4. “actor”, “action”, “object”, “time”, and “location” in “event_list” do not match the descriptions in the original “text”.
5. “actor”, “action”, “object”, “time”, and “location” in “event_list” are not exact transcriptions from the original text segment.

This structured approach ensures that the augmented data set $\{(\hat{x}_i^t, \hat{y}_i^t)\}$ maintains high quality and relevance, thereby enhancing the overall effectiveness of the Model Self-Data Augmentation process in event extraction tasks.

Subsequently, the large language model L is trained utilizing the augmented and revised data. The procedures of data augmentation and revision, as well as the training of the model, are delineated in Algorithm 1.

Algorithm 1 Model Self-Data Augmentation Algorithm for Event Extraction

Require: Training dataset $\{(x_i, y_i)\}$, additional parameters c_i^t , prompts p_1', p_2' , filters f_1', f_2' , and large language model L

Ensure: Updated large language model L'

Procedures:

GenerateModifiedEvents(y, c, p, f): Uses prompt p and filter f to modify extracted events based on type c . Output: Modified events $\hat{y}$.

ReconstructText(y, p, f): Uses prompt p and filter f to transform modified event extractions back into text. Output: Reconstructed text segments $\hat{x}$.

TrainModel($model, data$): Trains the model $model$ using the dataset $data$. Output: Updated model L' .

LoTSA($data$): Revise the dataset according to LoTSA method. Output: Revised dataset $Dataset$.

```

1:  $AugmentedDataset \leftarrow \emptyset$ 
2: for each  $y_i$  in original event extractions do
3:    $\hat{y}_i^t \leftarrow \text{GenerateModifiedEvents}(y_i, c_i^t, p_1', f_1')$ 
4:    $\hat{x}_i^t \leftarrow \text{ReconstructText}(\hat{y}_i^t, p_2', f_2')$ 
5:    $AugmentedDataset.Add((\hat{x}_i^t, \hat{y}_i^t))$ 
6: end for
7:  $CombinedDataset \leftarrow \{(x_i, y_i)\} \cup AugmentedDataset$ 
8:  $CombinedDatasetR \leftarrow \text{LoTSA}(CombinedDataset)$ 
9:  $L' \leftarrow \text{TrainModel}(L, CombinedDatasetR)$ 
10: return  $L'$ 

```

4 Experiment

4.1 Experimental Setup

4.1.1 Data Description

The original dataset for the experiment comprises two subsets: a training set and a testing set, each containing 100 samples. Each sample in the dataset is structured as shown in Figure 3.

In each sample, the “text” key contains the input text, and the “event_list” key provides the output, which is the result of the event extraction process. The training set resulted in the extraction of 350 events from its 100 samples, while the testing set yielded 357 events from its 100 samples.

```

{
  "text": "According to the AFP, on the 21st
local time, an accident occurred at a bridge
construction site in Gelderland, eastern
Netherlands, resulting in 2 deaths and at least 2
injuries. According to reports...",
  "event_list": [
    {
      "class": "Security Incidents",
      "actor": "an accident",
      "action": "occurred",
      "object": "",
      "time": "21st local time",
      "location": "a bridge construction site in
Gelderland, eastern Netherlands"
    },
    {
      "class": "Security Incidents",
      "actor": "an accident",
      "action": "resulting in",
      "object": "2 deaths and at least 2
injuries",
      "time": "21st local time",
      "location": "a bridge construction site in
Gelderland, eastern Netherlands"
    },
    ...
  ]
}

```

Fig. 3: Diagram of Sample Structure in the Dataset.

The type distribution of these events in both subsets is illustrated in Figures 4 and 5 respectively.

4.1.2 Experimental Design

The experimental design incorporates a methodology developed in the previous chapter. Initially, the original 100 training samples were augmented with an additional 3100 samples using a self-data augmentation strategy executed by the model itself. This augmentation strategy aims to enhance the diversity and volume of training data, potentially improving the model's ability to generalize across varied inputs.

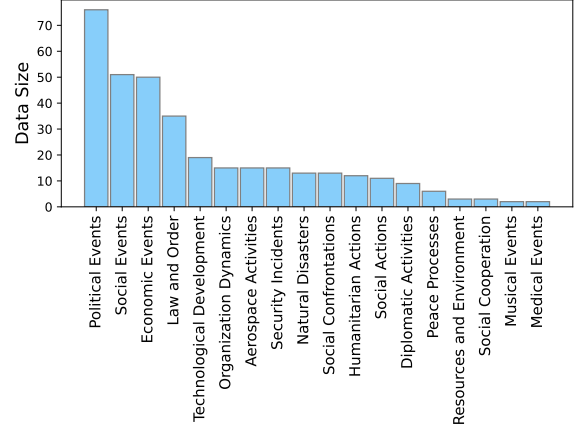


Fig. 4: Event Type Distribution in the Training Set.

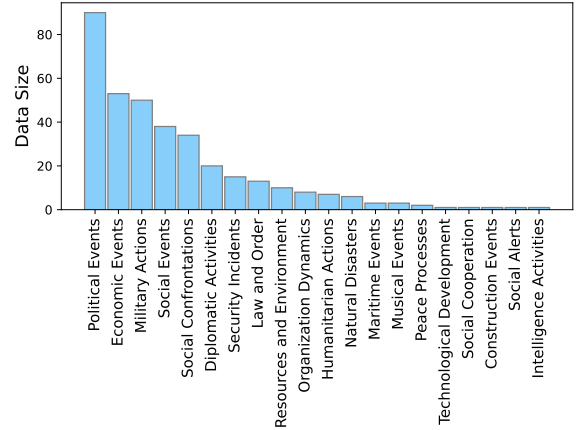


Fig. 5: Event Type Distribution in the Testing Set.

For comparative analysis, the experiment includes several models as experimental groups. These models include gpt-4-turbo-2024-04-09, gpt-3.5-turbo-0125, and Qwen1.5 models at various scales: Qwen1.5-72B-Chat, Qwen1.5-32B-Chat, Qwen1.5-14B-Chat, and Qwen1.5-7B-Chat. The reason for adopting the Qwen1.5 series models lies in that it includes a comprehensive range of models with varying sizes.

For the Qwen1.5 models, the training dataset sizes were systematically varied, incorporating datasets of 100 samples (original training dataset) and expanded datasets of 200, 400, 800, 1600, and 3200 samples. These expanded datasets combined the original training dataset with the augmented

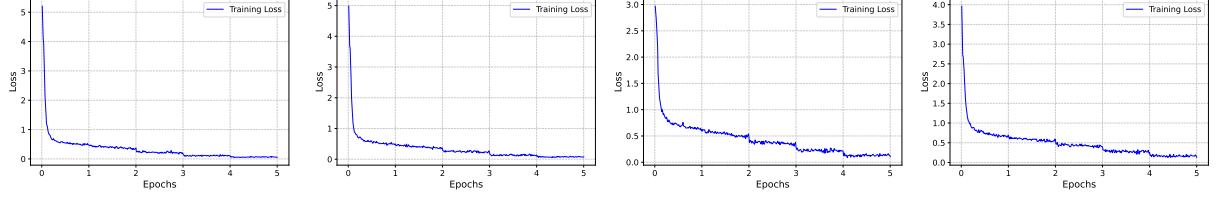


Fig. 6: Training Loss for Qwen1.5 Series with 3200 Samples.

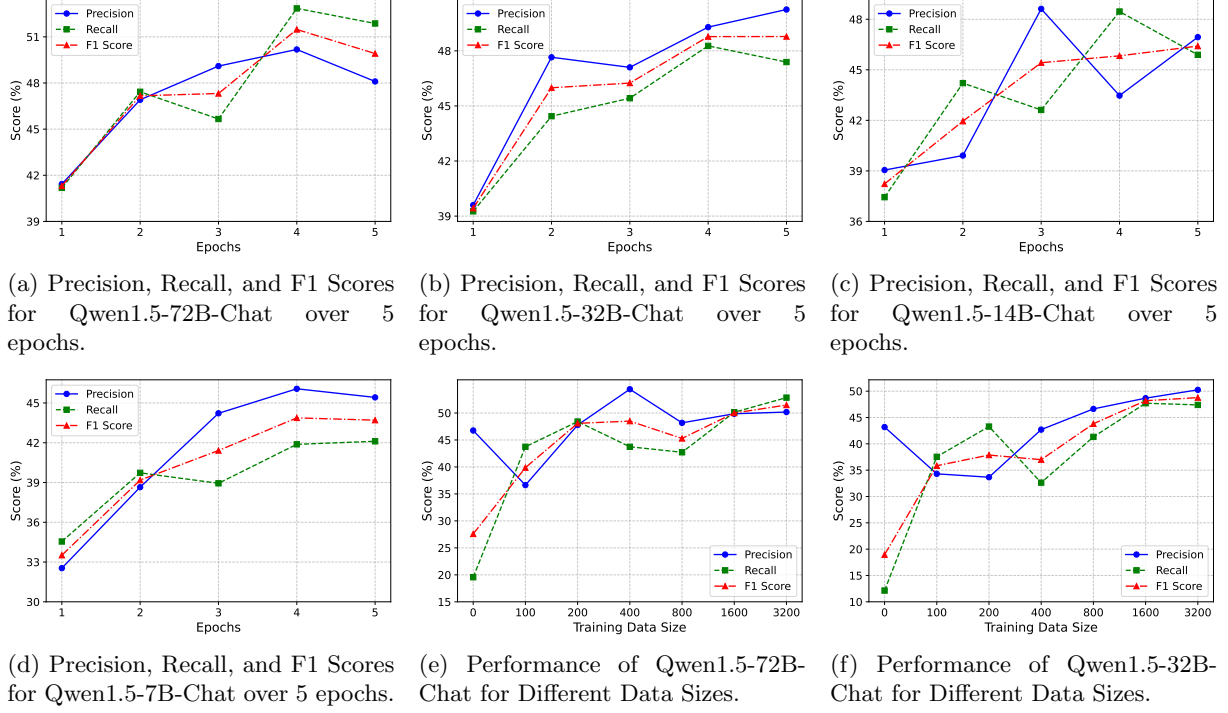


Fig. 7: Comprehensive Performance Metrics and Model Comparisons of Qwen1.5 series.

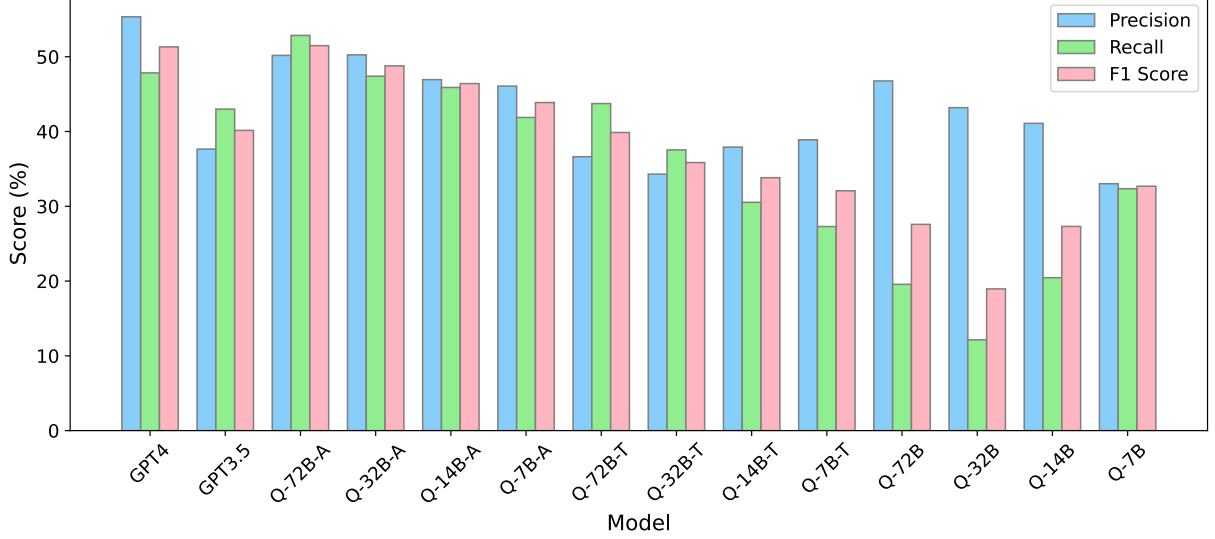


Fig. 8: Precision, Recall, and F1 Score comparison across different models and training conditions, where Q-72B-A refers to Qwen1.5-72B-Chat trained with data augmentation, Q-72B-T refers to Qwen1.5-72B-Chat trained without data augmentation, and Q-72B refers to the origin Qwen1.5-72B-Chat model. The model names for the 32B, 14B, and 7B sizes follow the same naming convention as the 72B model.

samples to rigorously test the scalability and adaptability of each model under different training conditions.

The training environment was configured with the following specifications:

- **CPU:** Intel(R) Xeon(R) Platinum 8352Y @ 2.20GHz, dual-socket
- **GPU:** NVIDIA(R) Tesla(R) A100 $\times$ 6
- **Training method:** Low-Rank Adaptation (LoRA)
- **Batch Size per GPU:** 8
- **Total Training Epochs:** 5
- **Learning Rate:** 0.0003
- **Warm-up Ratio:** 0.01

4.2 Experimental Results

The experimental results were derived from the setup described in Section 4.1.2. The experiments involved varying the training dataset sizes for the Qwen1.5 model series, including sizes of 100 (original dataset), 200, 400, 800, 1600, and 3200 samples, and testing the models' performance over five training epochs. Unless otherwise specified as "without revision", the term "augmented dataset" refers to data that has been revised using the LoTSA methodology.

The training loss for the Qwen1.5 series with 3200 samples is depicted in Figure 6.

Training performance across 1 to 5 epochs for the same models, in terms of precision, recall, and F1 score, is illustrated in Figures 7a, 7b, 7c, and 7d. Notably, performance peaks at the fourth epoch, which appears to be the most favorable. By the fifth epoch, training approaches saturation, as indicated by excessively small loss values, suggesting that performance has nearly maximized by the fourth and fifth epochs, and extending training beyond this point is likely to yield diminishing returns.

Figures 7e, 7f, 7g, and 7h show results for Qwen1.5 models (Qwen1.5-72B-Chat, Qwen1.5-32B-Chat, Qwen1.5-14B-Chat, and Qwen1.5-7B-Chat) trained on 0 (untrained), 100 (original dataset), 200, 400, 800, 1600, and 3200 samples. Expanding the training to 3200 samples considerably improved performance, with a general and consistent increase of above 10 percentage points in F1 score compared with the original dataset. Notably, the recall of the Qwen models was generally low before training but showed significant improvement post-training. The marked improvement in model performance with increased sample

Table 1: F1 Score across different training configurations of Qwen1.5 model series.

Model	Untrained	Trained w/o aug.	Trained w/ aug. w/o rev.	Trained w/ aug. & rev.
Qwen1.5-72B-Chat	27.58	39.87	47.77	51.48
Qwen1.5-32B-Chat	18.95	35.85	44.72	48.78
Qwen1.5-14B-Chat	27.31	33.82	42.82	46.41
Qwen1.5-7B-Chat	32.68	32.07	40.84	43.88

Table 2: Precision across different training configurations of Qwen1.5 model series.

Model	Untrained	Trained w/o aug.	Trained w/ aug. w/o rev.	Trained w/ aug. & rev.
Qwen1.5-72B-Chat	46.76	36.63	45.18	50.18
Qwen1.5-32B-Chat	43.19	34.30	43.12	50.25
Qwen1.5-14B-Chat	41.09	37.91	41.58	46.94
Qwen1.5-7B-Chat	33.02	38.89	40.78	46.07

size indicates that the additional samples effectively captured the characteristics of the original data and enhanced the models’ generalization ability, which is crucial for their application in diverse real-world scenarios.

Performance metrics of Precision, Recall, and F1 Score for various models including gpt-4-turbo-2024-04-09, gpt-3.5-turbo-0125, and the Qwen1.5 model series are compared in Figure 8. This comparison encompasses models trained without augmentation, and models trained with self-data augmentation on 3200 samples.

As indicated by the results, the original Qwen1.5 series models generally prioritize Precision, tending to produce more credible but fewer extracted events, which results in lower Recall scores. Following training on un-augmented data, there was a noticeable improvement in Recall, suggesting enhanced event extraction capabilities. However, due to the limited number of samples, the performance was suboptimal. Precision generally decreased, leading to a modest overall improvement in F1 Score.

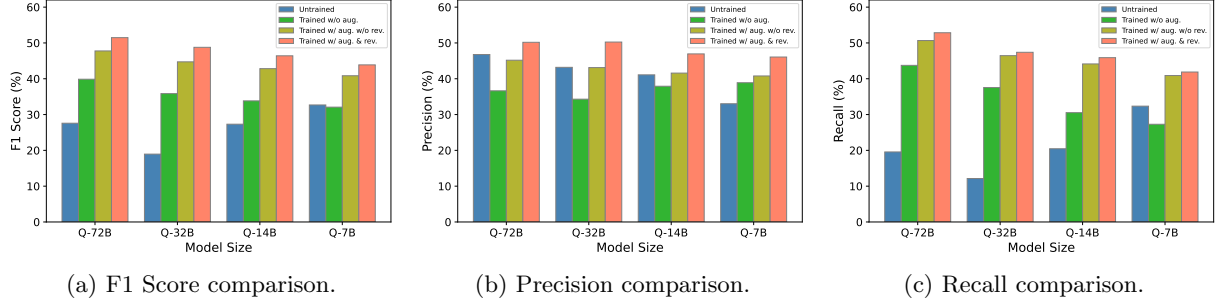
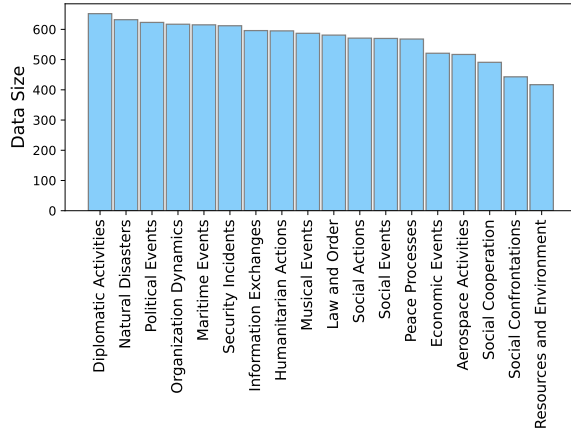
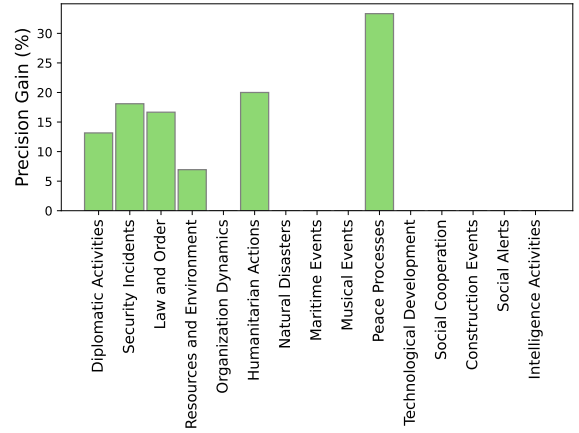
Through the training on 3200 samples, the accuracy of the models was restored, surpassing the capabilities of the original models, and recall was further enhanced. This resulted in a further increase in F1 Score. The augmented data training enabled the Qwen1.5-72B-Chat model to match the event extraction capabilities of GPT-4. Moreover, employing smaller-scale models (7B) with self-data augmentation training

achieved results comparable to larger-scale models (72B) trained on un-augmented data, confirming the effective impact of self-data augmentation. This approach significantly reduces computational costs by enabling smaller models to achieve high performance without the necessity to scale up the model size. Additionally, the model’s self-data augmentation capability eliminates the need for a second model to perform data augmentation, thereby conserving computational resources. The self-data augmentation strategy can also effectively address the limitations of small datasets, enhancing the model’s ability to generalize from limited data by artificially expanding the variety and volume of training samples.

Figure 10 presents the class distribution after self-data augmentation. Comparing this with Figures 4 and 5, it is evident that the event distribution has become more balanced, benefiting the extraction of events that belong to classes with smaller proportions. Figure 11 contrasts the precision of models trained with 3200 augmented data points (Qwen1.5-72B-Chat model) against models trained with non-augmented data, specifically focusing on events categorized under classes with fewer occurrences (events where the “class” field represents less than 20 occurrences in Figure 5). While there is no improvement in the accuracy for events where the “class” field represents very few occurrences (equal or less than 3 occurrences in Figure 5), there is a noticeable improvement

Table 3: Recall across different training configurations of Qwen1.5 model series.

Model	Untrained	Trained w/o aug.	Trained w/ aug. w/o rev.	Trained w/ aug. & rev.
Qwen1.5-72B-Chat	19.56	43.74	50.67	52.85
Qwen1.5-32B-Chat	12.14	37.54	46.45	47.39
Qwen1.5-14B-Chat	20.45	30.53	44.14	45.89
Qwen1.5-7B-Chat	32.35	27.29	40.90	41.88

**Fig. 9:** Comparative analysis of performance metrics across different models and training conditions of Qwen1.5 model series, where Q-72B refers to Qwen1.5-72B-Chat, Q-32B refers to Qwen1.5-32B-Chat, Q-14B refers to Qwen1.5-14B-Chat, and Q-7B refers to Qwen1.5-7B-Chat.**Fig. 10:** Class distribution after data augmentation.**Fig. 11:** Precision gain for less frequent event classes using augmented data versus non-augmented data in training the Qwen1.5-72B-Chat model.

in accuracy for other classes with fewer occurrences (more than 3 occurrences and less than 20 occurrences).

The content below revolves around the LoTSA process for data revision. The ensuing analysis offers a detailed examination of the performance metrics—F1 Score, Precision, and Recall—for the

Qwen1.5 Chat models across various configurations: untrained, trained without data augmentation, trained with augmentation but without revision, and trained with augmentation and revision. The specific metrics, F1 Score, Precision, and Recall, for each model configuration are detailed in Tables 1, 2, and 3. It is apparent that

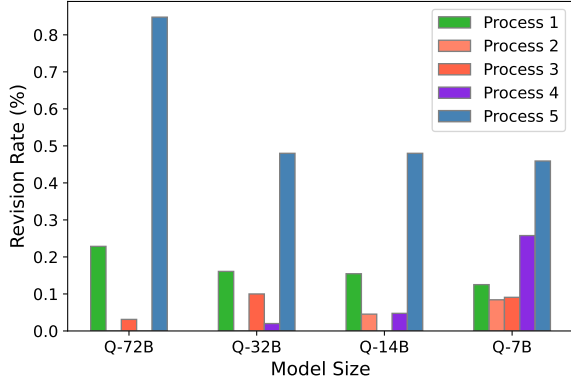


Fig. 12: Revision rates across different processes.

the utilization of augmented data without revision already provides a performance enhancement compared to models trained without data augmentation. Following the revision process, there is a further improvement in performance metrics. Notably, in terms of precision, models trained with revised augmented data exhibit a general increase exceeding 5% across different model sizes, while the improvement in recall is less pronounced. Figure 12 illustrates the revision rates of different processes (*i.e.*, the proportion of modifications relative to the total sample size), where Process 5 has the highest revision rate, corresponding to the accuracy of event extraction elements. Conversely, Process 1, while having the second highest rate, significantly trails behind Process 5, correlating with the event recall. This supports the observation that the increase in precision surpasses that of recall.

Together, these visual and tabular presentations not only highlight the effectiveness of the self-data augmentation strategy but also demonstrate how it significantly enhances the performance metrics of various Qwen1.5 series models. The overall improvements in F1 Score, Precision, and Recall elucidate the substantial benefits derived from adopting self-augmented training practices.

5 Conclusion

This paper presents a novel self-data augmentation strategy for event extraction, utilizing a single large language model (LLM) to concurrently perform both data augmentation and event extraction. This integrated approach not only

streamlines the workflow but also increases the efficiency and effectiveness of the event extraction process. By dynamically assessing and revising the quality of generated samples within the same framework, the introduction of noisy data is minimized, thereby enhancing the model’s performance across various metrics such as precision, recall, and F1 scores. Demonstrated results indicate substantial improvements in managing small and imbalanced datasets, affirming the effectiveness of this strategy in addressing traditional challenges associated with event extraction tasks. Moreover, the incorporation of Logical Thoughts for Self-Data Augmentation (LoTSA) within this strategy facilitates the creation of high-quality and reliable augmented datasets. This approach not only counteracts the typical degradation found in traditional data augmentation methods but also diminishes the computational and resource overhead by obviating the need for multiple models or extensive external augmentation frameworks.

References

- [1] Christopher Manning and Hinrich Schutze. *Foundations of statistical natural language processing*. MIT press, 1999.
- [2] Jun Gao, Huan Zhao, Changlong Yu, and Ruifeng Xu. Exploring the feasibility of chatgpt for event extraction. *arXiv preprint arXiv:2303.03836*, 2023.
- [3] Jun Gao, Changlong Yu, Wei Wang, Huan Zhao, and Ruifeng Xu. Mask-then-fill: A flexible and effective data augmentation framework for event extraction. *arXiv preprint arXiv:2301.02427*, 2023.
- [4] Timo Möller, Anthony Reina, Raghavan Jayakumar, and Malte Pietsch. Covid-qa: A question answering dataset for covid-19. In *ACL 2020 Workshop on Natural Language Processing for COVID-19 (NLP-COVID)*, 2020.
- [5] Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*, 2021.
- [6] Raj Kumar Gupta, Naveena N, B. Srinivasa Rao, Rajasree RS, Swagata Sarkar, and Kallakunta Ravi Kumar. Deep learning for uneven data in industrial iot using

- a distributed bias-aware adversarial network. In *2023 5th International Conference on Inventive Research in Computing Applications (ICIRCA)*, pages 1435–1440, 2023.
- [7] Agnieszka Mikołajczyk and Michał Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 international interdisciplinary PhD workshop (IIPhDW)*, pages 117–122. IEEE, 2018.
 - [8] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.
 - [9] Meishan Zhang, Gongyao Jiang, Shuang Liu, Jing Chen, and Min Zhang. Llm-assisted data augmentation for chinese dialogue-level dependency parsing. *Computational Linguistics*, pages 1–24, 2024.
 - [10] Huanhuan Chen, Bingbing Jiang, and Xin Yao. Semisupervised negative correlation learning. *IEEE transactions on neural networks and learning systems*, 29(11):5366–5379, 2018.
 - [11] Xiangyu Wang, Lyuzhou Chen, Derui Lyu, Taiyu Ban, Yifeng Guan, and Qiuju Chen. Research concept link prediction via graph convolutional network. In *2022 8th International Conference on Big Data and Information Analytics (BigDIA)*, pages 220–225. IEEE, 2022.
 - [12] Anders Giovanni Møller, Arianna Pera, Jacob Dalsgaard, and Luca Aiello. The parrot dilemma: Human-labeled vs. llm-augmented data in classification tasks. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 179–192, 2024.
 - [13] Luke Taylor and Geoff Nitschke. Improving deep learning with generic data augmentation. In *2018 IEEE symposium series on computational intelligence (SSCI)*, pages 1542–1547. IEEE, 2018.
 - [14] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
 - [15] Claude Coulombe. Text data augmentation made simple by leveraging nlp cloud apis. *arXiv preprint arXiv:1812.04718*, 2018.
 - [16] Thien Ho Huong and Vinh Truong Hoang. A data augmentation technique based on text for vietnamese sentiment analysis. In *Proceedings of the 11th International Conference on Advances in Information Technology*, pages 1–5, 2020.
 - [17] Siyuan Qiu, Binxia Xu, Jie Zhang, Yafang Wang, Xiaoyu Shen, Gerard De Melo, Chong Long, and Xiaolong Li. Easyaug: An automatic textual data augmentation platform for classification tasks. In *Companion proceedings of the web conference 2020*, pages 249–252, 2020.
 - [18] Yonatan Belinkov and Yonatan Bisk. Synthetic and natural noise both break neural machine translation. *arXiv preprint arXiv:1711.02173*, 2017.
 - [19] Ashutosh Kumar, Satwik Bhattamishra, Manik Bhandari, and Partha Talukdar. Submodular optimization-based diverse paraphrasing and its effectiveness in data augmentation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3609–3619, 2019.
 - [20] Rong Xiang, Emmanuele Chersoni, Qin Lu, Chu-Ren Huang, Wenjie Li, and Yunfei Long. Lexical data augmentation for sentiment analysis. *Journal of the Association for Information Science and Technology*, 72(11):1432–1447, 2021.
 - [21] Huanhuan Chen and Xin Yao. Multiobjective neural network ensembles based on regularized negative correlation learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(12):1738–1751, 2010.
 - [22] Xin Wang, Shengfei Lyu, Lishan Yang, Yibing Zhan, and Huanhuan Chen. A dual-module framework for counterfactual estimation over time. In *The 41st International Conference on Machine Learning*. JMLR, 2024.
 - [23] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.
 - [24] Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. *arXiv preprint*

- arXiv:1804.07998*, 2018.
- [25] Huanhuan Chen, Fengzhen Tang, Peter Tino, and Xin Yao. Model-based kernel for efficient time series analysis. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 392–400, 2013.
 - [26] William Yang Wang and Diyi Yang. That’s so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2557–2563, 2015.
 - [27] Marzieh Fadaee, Arianna Bisazza, and Christof Monz. Data augmentation for low-resource neural machine translation. *arXiv preprint arXiv:1705.00440*, 2017.
 - [28] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.
 - [29] Xiangyu Wang, Lyuzhou Chen, Taiyu Ban, Muhammad Usman, Yifeng Guan, Shikang Liu, Tianhao Wu, and Huanhuan Chen. Knowledge graph quality control: A survey. *Fundamental Research*, 1(5):607–626, 2021.
 - [30] Sosuke Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. *arXiv preprint arXiv:1805.06201*, 2018.
 - [31] Gözde Gül Şahin and Mark Steedman. Data augmentation via dependency tree morphing for low-resource languages. *arXiv preprint arXiv:1903.09460*, 2019.
 - [32] Huanhuan Chen, Peter Tiño, Ali Rodan, and Xin Yao. Learning in the model space for cognitive fault diagnosis. *IEEE transactions on neural networks and learning systems*, 25(1):124–136, 2013.
 - [33] Xiangyu Wang, Taiyu Ban, Lyuzhou Chen, Muhammad Usman, Yifeng Guan, Derui Lyu, Jian Cheng, Huanhuan Chen, Cyril Leung, and Chunyan Miao. Decentralised knowledge graph evolution via blockchain. *IEEE Transactions on Services Computing*, 2023.
 - [34] Yan Xu, Ran Jia, Lili Mou, Ge Li, Yunchuan Chen, Yangyang Lu, and Zhi Jin. Improved relation classification by deep recurrent neural networks with data augmentation. *arXiv preprint arXiv:1601.03651*, 2016.
 - [35] Anna Kruspe, Jens Kersten, Matti Wiegmann, Benno Stein, and Friederike Klan. Classification of incident-related tweets: Tackling imbalanced training data using hybrid cnns and translation-based data augmentation. *Notebook papers of TREC*, 2018.
 - [36] Huanhuan Chen, Fengzhen Tang, Peter Tino, Anthony G Cohn, and Xin Yao. Model metric co-learning for time series classification. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 3387–3394. AAAI Press, 2015.
 - [37] Xiangyu Wang, Lyuzhou Chen, Taiyu Ban, Derui Lyu, Yifeng Guan, Xingyu Wu, Xiren Zhou, and Huanhuan Chen. Accurate label refinement from multiannotator of remote sensing data. *IEEE Transactions on Geoscience and Remote Sensing*, 61:1–13, 2023.
 - [38] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*, 2015.
 - [39] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.
 - [40] Huanhuan Chen, Hao Zhang, Wei Tang, Qian Xi, Xiaoyun Liu, Yu Duan, and Chao Liu. Thyroid function and morphology in overweight and obese children and adolescents in a chinese population. *Journal of Pediatric Endocrinology and Metabolism*, 26(5-6):489–496, 2013.
 - [41] Xiangyu Wang, Yuan Li, Taiyu Ban, Jiarun Zhu, Lyuzhou Chen, Muhammad Usman, Xin Wang, Huanhuan Chen, Xiaomei Chen, Cyril Leung, et al. Dynamic link prediction for discovery of new impactful covid-19 research approaches. *IEEE Journal of Biomedical and Health Informatics*, 26(12):5883–5894, 2022.
 - [42] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.
 - [43] Georgios Rizos, Konstantin Hemker, and

- Björn Schuller. Augment to prevent: short-text data augmentation in deep learning for hate-speech classification. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 991–1000, 2019.
- [44] Huanhuan Chen, Lei Zhang, Zhenqiang Wang, Haoya Chang, Xiaogang Xie, Liangzheng Fu, Yong Zhang, and Fusheng Quan. Resveratrol improved the developmental potential of oocytes after vitrification by modifying the epigenetics. *Molecular reproduction and development*, 86(7):862–870, 2019.
- [45] Xiangyu Wang, Taiyu Ban, Lyuzhou Chen, Muhammad Usman, Tianhao Wu, Qiuju Chen, and Huanhuan Chen. A distribution-based representation of knowledge quality. *Knowledge-Based Systems*, 281:111054, 2023.
- [46] Xiao Sun and Jiajin He. A novel approach to generate a large scale of supervised data for short text sentiment analysis. *Multimedia Tools and Applications*, 79(9):5439–5459, 2020.
- [47] Marilyn Walker, Heng Ji, and Amanda Stent. Proceedings of the 2018 conference of the north american chapter of the association for computational linguistics: Human language technologies, volume 1 (long papers). In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018.
- [48] Huanhuan Chen, Lei Zhang, Li Meng, Linlin Liang, and Cuilian Zhang. Advantages of vitrification preservation in assisted reproduction and potential influences on imprinted genes. *Clinical Epigenetics*, 14(1):141, 2022.
- [49] Xin Wang, Shengfei Lyu, Xiangyu Wang, Xingyu Wu, and Huanhuan Chen. Temporal knowledge graph embedding via sparse transfer matrix. *Information Sciences*, 623:56–69, 2023.
- [50] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [51] Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. Conditional bert contextual augmentation. In *Computational Science–ICCS 2019: 19th International Conference, Faro, Portugal, June 12–14, 2019, Proceedings, Part IV 19*, pages 84–95. Springer, 2019.
- [52] Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. Do not have enough data? deep learning to the rescue! In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7383–7390, 2020.
- [53] Huanhuan Chen, Peter Tiño, and Xin Yao. Cognitive fault diagnosis in tennessee eastman process using learning in the model space. *Computers & chemical engineering*, 67:33–42, 2014.
- [54] Xin Wang, Shengfei Lyu, Xingyu Wu, Tianhao Wu, and Huanhuan Chen. Generalization bounds for estimating causal effects of continuous treatments. *Advances in Neural Information Processing Systems*, 35:8605–8617, 2022.
- [55] Kenton Lee, Kelvin Guu, Luheng He, Tim Dozat, and Hyung Won Chung. Neural data augmentation via example extrapolation. *arXiv preprint arXiv:2102.01335*, 2021.
- [56] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [57] Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyeong Park. Gpt3mix: Leveraging large-scale language models for text augmentation. *arXiv preprint arXiv:2104.08826*, 2021.
- [58] Dmytro Kalpakchi and Johan Boye. Quasi: a synthetic question-answering dataset in swedish using gpt-3 and zero-shot learning. In *The 24th Nordic Conference on Computational Linguistics (NoDaLiDa 2023), 22-24 May 2023, Tórshavn, Faroe Islands*, pages 477–491, 2023.
- [59] Vinay Samuel, Houda Aynaou, Arijit Ghosh Chowdhury, Karthik Venkat Ramanan, and Aman Chadha. Can llms augment low-resource reading comprehension datasets? opportunities and challenges. *arXiv preprint*

arXiv:2309.12426, 2023.

- [60] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.
- [61] Ruirui Chen, Chengwei Qin, Weifeng Jiang, and Dongkyu Choi. Is a large language model a good annotator for event extraction? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17772–17780, 2024.
- [62] Junjie Yu, Xing Wang, and Wenliang Chen. Reliable data generation and selection for low-resource relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19440–19448, 2024.
- [63] Martin Josifoski, Marija Sakota, Maxime Peyrard, and Robert West. Exploiting asymmetry for synthetic training data generation: Synthie and the case of information extraction. *arXiv preprint arXiv:2303.04132*, 2023.
- [64] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
- [65] Xufeng Zhao, Mengdi Li, Wenhao Lu, Cornelius Weber, Jae Hee Lee, Kun Chu, and Stefan Wermter. Enhancing zero-shot chain-of-thought reasoning in large language models through logic. *arXiv preprint arXiv:2309.13339*, 2023.