LC4EE: LLMs as Good Corrector for Event Extraction

Mengna Zhu[1], **Kaisheng Zeng**[2,3], **Jibing Wu**[1], **Lihua Liu**[1],
Hongbin Huang[1†], **Lei Hou**[2†], **Juanzi Li**[2]
[1] Laboratory for Big Data and Decision, National University of Defense Technology
[2] Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
[3]College of Information and Communication, National University of Defense Technology
zhumengna16@nudt.edu.cn

Abstract

Event extraction (EE) is a critical task in natural language processing, yet deploying a practical EE system remains challenging. On one hand, powerful large language models (LLMs) currently show poor performance because EE task is more complex than other tasks. On the other hand, state-of-the-art (SOTA) small language models (SLMs) for EE tasks are typically developed through fine-tuning, lack flexibility, and have considerable room for improvement. We propose an approach, **L**LMs-as-**C**orrector for **E**vent **E**xtraction (**LC4EE**), aiming to leverage the superior extraction capability of SLMs and the instruction-following ability of LLMs to construct a robust and highly available EE system. By utilizing LLMs to identify and correct errors of SLMs predictions based on automatically generated feedback information, EE performances can be improved significantly. Experimental results on the representative datasets ACE2005 and MAVEN-Arg for Event Detection (ED) and EE tasks validated the effectiveness of our method.

1 Introduction

Events are basic units of human activities and interactions, containing rich information which is vital for downstream applications (Huang et al., 2019; Wang et al., 2021a). Event extraction is the process of identifying and extracting structured event information from unstructured text. It includes: 1) Event Detection, which detects event triggers and classifies them into pre-defined event types. 2) Event Argument Extraction, which extracts relevant arguments associated with each event.

However, developing a highly practical and flexible EE system remains challenging (Lu et al., 2022). Current best performance of EE task on the most commonly used dataset ACE 2005 only reaches 57.9% (Lu et al., 2023), far from meeting practical requirements.

One approach to overcoming this challenge is to explore the utilization of powerful LLMs, and there have been some attempts (Li et al., 2023; Guo et al., 2023; Pang et al., 2023). These methods have exhibited promising performance of LLMs in EE tasks, yet they still fall short compared to SLMs (Wang et al., 2021b). Although SLMs have superior performances, they mainly complete EE tasks through fine-tuning, which is not flexible enough to adjust predictions based on user feedback, and there is still significant room for improvement in performance. Existing research has demonstrated that LLMs can effectively adjust their responses based on feedback information, which is beneficial for improving task performance(Gao et al., 2023; Yin et al., 2023; Huang et al., 2023).

Therefore, we aim to combine the advantages of LLMs and SLMs, fully utilizing the instruction-following ability and feedback understanding capability of LLMs, along with the superior event extraction ability of SLMs. By correcting prediction errors of SLMs based on error feedback, we seek to enhance the performance of EE tasks and build a robust, highly available EE system.

We developed a method, **LC4EE**, which used LLMs as corrector to correct SLMs prediction errors for EE task based on automatically generated error feedback information. Our method consists of two main components: 1) **Rules Inducer for Feedback Generation.** By guiding LLMs to analyze training set samples and summarize rules to generate feedback information automatically for identification and correction of SLMs prediction errors. Human feedback combined with annotation guidelines are provided to refine the rules, leading to the formation of a rule repository for feedback generation after validation on the valid set. 2) **LLMs corrector.** According to retrieved rules from the rule repository for Feedback Generation, LLMs are guided to verify predictions and generate error feedback information for incorrect samples, finally

†Corresponding authors.

Findings of the Association for Computational Linguistics ACL 2024, pages 12028–12038

making corrections based on these information.

Our contributions are summarized as follows:

1. We validated that LLMs can learn from error feedback well through experiments in Section 2.

2. We designed LC4EE to identify and correct errors in SLMs predictions based on automatically generated error feedback in Section 3, which can fully utilize the rich information in errors and maximizes advantages of both SLMs and LLMs.

3. We conducted experiments for ED and EE tasks on ACE2005 and MAVEN-Arg based on some mainstream SLMs and completed corrections with advanced LLMs, GPT-3.5 and GPT-4. Experimental results show that LC4EE could bring effective performance improvements.

2 Preliminary Study

In this section, we validated the ability of the LLMs to correct errors based on the error feedback information under the oracle setting (i.e., only conduct corrections based on incorrect samples) first of all. To thoroughly explore this ability, we designed four granularity levels of feedback information and compared their performance differences.

2.1 Experiment setting

Our method does not rely on specific EE models and schema. Hence structured EE outputs will work with our method. We refer to current mainstream research in EE task (Peng et al., 2023; Wang et al., 2023; Hsu et al., 2022) to select datasets, metrics, and related models to validate our method.

Dataset and Metrics. Datasets we selected are ACE 2005 (Christopher et al., 2006) and MAVEN-Arg (Wang et al., 2023), which are representative datasets in EE task. Dataset details are shown in Appendix A. Following mainstream research, Precision, Recall, and F1 scores are used for evaluation.

Small Language Models. We selected some of the most advanced EE models, including: **1) EEQA** (Du and Cardie, 2020), which transforms EE task into a question-answering task. **2) TEXT2EVENT** (Lu et al., 2021), which unifies the entire EE process within a neural network-based sequence-to-structure architecture. **3) CLEVE** (Wang et al., 2021b), which completes EE based on a contrastive pre-training framework. Training details of these models can be seen in Appendix B.

Large Language Models. The performance of prompt engineering methods largely depends on the capabilities of the chosen LLMs (Brown et al., 2020). Therefore, we choose to use the currently most advanced LLMs, **GPT-3.5** (OpenAI, 2022) and **GPT-4** (OpenAI, 2023), to explore the boundaries of LLMs error correction capabilities.

2.2 Feedback for Error Correction

To obtain more effective and accurate error feedback, we conducted error analysis on the SLMs predictions and summarized error types. Detailed information can be seen in Appendix C.

Based on the error analysis results, we obtained error feedback for each sample by automatically comparing prediction with label information. To explore the impact of error feedback on the LLMs error correction capabilities, we designed four different granularity levels of error feedback and relevant correct rules. The granularity of feedback is progressively refined, including: **L1:** simply "incorrect", **L2:** error types information with corresponding quantities, **L3:** incorrect elements, **L4:** specific error information. Detailed definitions and examples are provided in Appendix D.

2.3 Experimental results and analysis

We sampled 100 instances with prediction errors from SLMs predictions on the test set. Results can be seen in Figure 1 and Appendix E.

According to the experimental results, it can be seen that except for the EE task on MAVEN-Arg, all other tasks have very obvious improvement, with the F1 score showing an increase up to more than **50%**, and the finer the feedback granularity, the more obvious the performance improvement. As for EE task on MAVEN-Arg, since the dataset itself has more event types, finer classification, and more complex text, which is much more difficult compared to ACE 2005, a finer granularity feedback may be needed. But overall, the results can verify that based on the detailed error feedback information, LLMs can correct errors well.

3 LC4EE

Preliminary Study in Section 2 has validated that LLMs possesses strong capabilities for error correction, with significant improvements when detailed error information is available. However, in practical applications, obtaining a large volume of detailed error feedback is not feasible.

Therefore, in this section, we will explore the ability of LLMs to provide as fine as possible er-

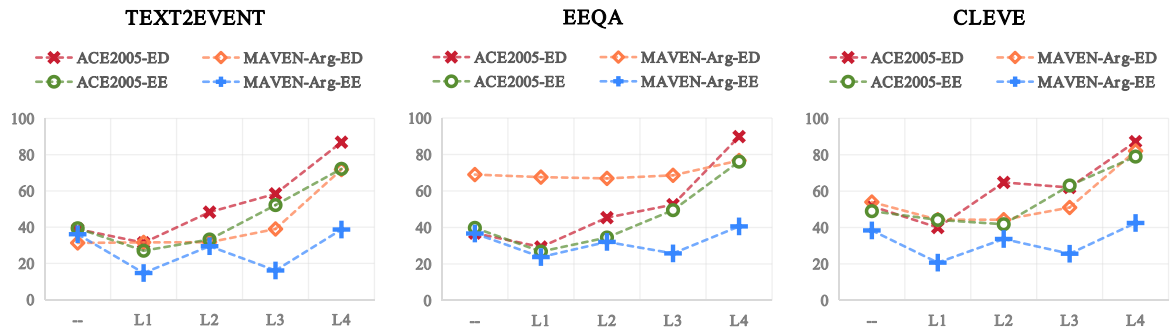

Figure 1: Results of Preliminary Study which compares different performances with 4 granularity level feedback information. "-" represents the origin performance of SLMs before correction.

ror feedback automatically and then make corrections based on feedback for incorrect samples. The model LC4EE we designed is illustrated in Figure 2, which consists of two main components: a Rules Inducer for Feedback Generation and a LLMs Corrector. Rules Inducer for Feedback Generation is used to summarize rules based on incorrect samples for error feedback generation. These rules will be retrieved in LLMs Corrector, then corrections are made based on the generated feedback. We will describe these two components in detail as follows.

3.1 Rules Inducer for Feedback Generation

To generate error feedback information as finer and accurate as possible, we guide the LLMs to induce rules for prediction errors identification and generate feedback according to these rules. These rules are iteratively refined in conjunction with human feedback and annotation guidelines information.

To facilitate rules induction by LLMs, we categorize samples according to event types. For event types with high frequencies of errors in the error analysis, we sampled SLMs predictions from the training set, providing both prediction results and label information, instructing GPT-4 to analyze samples to summarize rules to identify prediction errors and generate feedback for incorrect samples, along with corresponding demonstrations. Humans provide feedback combined with annotation guidelines information by pointing out inaccuracies or improprieties of rules, thus guiding LLMs to refine rules iteratively, enhancing understanding of event schema. Examples can be seen in Appendix F.

We validated rules approved by humans on the valid set. When $\Delta F1 > 0$, it indicates that the F1 score has been improved and the rule is effective. We retain such rules and establish a rule repository. Rules details are shown in Appendix G.

3.2 LLMs Corrector based on Feedback

With several high-quality feedback generation rules available, we designed LLMs Corrector consisting of Retriever, Verifier and Corrector. Retriever is used to retrieve suitable rules for feedback generation. Verifier is used to identify incorrect predictions and generate feedback. Corrector is used to correct errors according to feedback. We provided an example in Appendix H.

Rules Retriever: Firstly, to better identify incorrect samples and generate more accurate feedback, we aim to find the most appropriate rules. By analyzing data from the training set, we have built a dictionary of candidate triggers corresponding to event types which serves as a keyword list. Leveraging superior keyword matching capabilities of LLMs, we obtained candidate triggers of the sample. Rules corresponding to event type associated with candidate triggers and event types in the predictions can be retrieved from the rule repository.

Verifier: Based on the feedback generation rules retrieved by the Retriever according to event types in the predicted result and candidate triggers extracted from the sample text, we instruct LLMs to evaluate each sample combined with semantic information and generate feedback information, similar to the L4 granularity mentioned in Section 2. If any rule is not met, the sample is verified as an incorrect sample. Utilizing the Verifier, we filter out incorrect samples and obtain error feedback information.

Corrector: Corrector is used to make corrections on the incorrect predictions based on error feedback information provided by Verifier and correction rules mentioned in Section 2. We retain original correct predictions, ultimately obtaining the final prediction results corrected by LLMs.

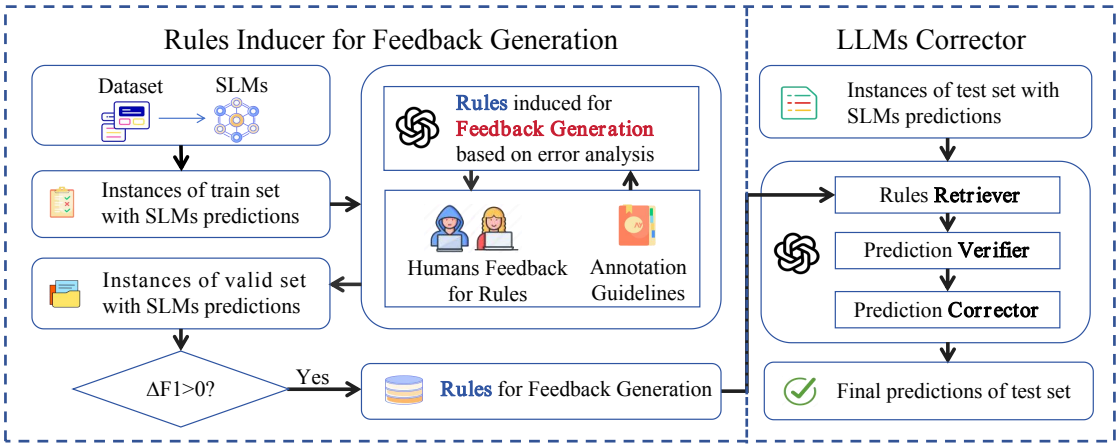

Figure 2: Overview of LC4EE, which consists of a Rules Inducer for Feedback Generation and a LLMs corrector.

model	Event Detection						Event Extraction					
	ACE2005			MAVEN-Arg			ACE2005			MAVEN-Arg		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
TEXT2EVENT	66.7	70.0	68.3	**74.5**	**20.5**	**32.2**	45.9	52.6	49.0	41.8	29.4	34.5
+GPT-3.5	66.1	73.7	69.7	71.1	20.0	31.2	46.3	52.9	49.4	42.1	29.8	34.9
+GPT-4	**68.1**	**74.5**	**71.2**	71.8	20.5	31.9	**46.8**	**53.6**	**50.0**	**42.5**	**30.3**	**35.4**
EEQA	71.9	67.2	69.5	69.1	71.2	70.1	47.1	41.3	44.0	40.1	31.9	35.5
+GPT-3.5	71.7	72.5	72.1	70.2	71.7	71.0	48.2	42.4	45.1	41.1	32.5	36.3
+GPT-4	**72.7**	**72.7**	**72.7**	**71.3**	**72.1**	**71.7**	**49.1**	**43.3**	**46.0**	**41.9**	**33.1**	**37.0**
CLEVE	72.4	78.1	75.1	68.1	37.7	48.5	48.0	61.0	53.7	47.4	19.1	27.2
+GPT-3.5	74.8	78.7	76.7	68.6	39.5	50.1	48.4	61.7	54.2	47.6	19.8	28.0
+GPT-4	**75.2**	**79.4**	**77.2**	**70.5**	**40.1**	**51.1**	**49.1**	**62.3**	**54.9**	**48.3**	**21.5**	**29.8**

Table 1: Results obtained by LC4EE which completed correction based on automatically generated feedback.

3.3 Results and Analysis of LC4EE

Correction results of LC4EE are shown in Table 1, which validate the effectiveness of our method and the capability of LLMs for error identification and correction. Detailed analysis is as follows.

Overall results. Through experimental results, it was found that feedback generated by LLMs based on a few limited rules led to significant improvements in almost all performances. Particularly, the best performance was observed on the relatively simpler ACE 2005 dataset for the ED task, with an average improvement of **2.7%**. The LLMs require relatively less analysis and can produce higher-quality feedback. However, the performance on EE tasks was less impressive compared to ED tasks. An important factor is error propagation from ED tasks which has been revealed in error analysis mentioned in Appendix C. Furthermore, EE tasks are inherently more complex, demanding LLMs have a more comprehensive semantic understanding of context, but noticeable improvements were still achieved. Results obtained from Maven-Arg were also inferior to those from ACE2005. This is because the Maven-Arg dataset itself is more complex, with a notably higher proportion of incorrect samples in the SLMs predictions, making it significantly more challenging for LLMs.

Analysis of Verifier. We evaluate the Verifier from two dimensions: error identification ability and quality of generated feedback. The average error identification rate based on retrieved rules of Verifier is **92.4%**, indicating that the model has a good ability to identify errors. We sampled 100 feedback information for manual inspection, with **87%** of them meeting human criteria.

Analysis of Corrector. The accuracy of the correction based on feedback information which represents the proportion of correctly corrected samples among all modified samples is **72.4%** overall. It indicates that although the LLMs inevitably makes some corrections that turn correct answers into errors, the benefit of correcting errors obviously outweighs the cost of introducing new errors.

For more comprehensive analysis, we comple-

mented details of the corrections based on the sampled data in Table 3 in Appendix I.

4 Conclusions and Future Work

We proposed the LC4EE model, which combined the superior EE capabilities of SLMs with the powerful semantic understanding and instruction-following abilities of LLMs. In this way, LLMs are used to correct prediction errors of SLMs based on automatically generated feedback information. Our model not only effectively improves EE task performances but also establishes a more flexible and practical EE system. We have validated the effectiveness of our method through some experiments. In our future work, we plan to extend LC4EE to more tasks, models, and datasets. In addition, we will increase the number of feedback generation rules and enhance their quality to further improve EE task performance.

Acknowledgements

This work is supported by a grant from the Institute for Guo Qiang, Tsinghua University (2019GQB0003).

Limitations

As our primary goal is to validate the effectiveness of our method, there are still some limitations in the current work. The major limitations of LC4EE are twofold: 1) In order to guarantee the proportion of incorrect samples, we choose the most complex task, Event Extraction task, performance of which still has significant room to improve. The models, tasks and datasets that we chose are limited. In the future, we plan to conduct more experiments to extend our method. 2) To improve correction efficiency and save cost, we just inducted rules for event types which have high frequencies of prediction errors. In this way, performance can be improved more obviously. In our future work, we will increase the number of rules and optimise our rules to further improve EE performance.

Ethical Considerations

We will discuss the ethical considerations and broader impact of this work here: (1) **Intellectual property.** LC4EE adhere to the original licenses for all datasets and models used. Regarding the issue of data copyright, we do not provide the original data and we will only provide processing scripts for the original data and relevant prompts. (2) **Intended Use.** LC4EE can be utilized to provide event understanding services for users, and it can also serve as an important reference for the design of methods for other tasks. (3) **Misuse risks.** The output of LC4EE is determined by the input text and should not be used to process and analyze sensitive or uncopyrighted data and should not be used to support political claims. (4) **Environmental Impact.** The experiments are conducted on the RTX4090 GPUs and consume approximately 100 GPU hours in total. This results in some carbon emissions, which incurs a negative influence on our environment.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Walker Christopher, Strassel Stephanie, Medero Julie, and Maeda Kazuaki. 2006. Ace 2005 multilingual training corpus. `https://catalog.ldc.upenn.edu/LDC2006T06`.

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

Ge Gao, Hung-Ting Chen, Yoav Artzi, and Eunsol Choi. 2023. Continually improving extractive qa via human feedback.

Yucan Guo, Zixuan Li, Xiaolong Jin, Yantao Liu, Yutao Zeng, Wenxuan Liu, Xiang Li, Pan Yang, Long Bai, Jiafeng Guo, and Xueqi Cheng. 2023. Retrieval-augmented code generation for universal information extraction.

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.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*

and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Zixian Huang, Jiaying Zhou, Gengyang Xiao, and Gong Cheng. 2023. Enhancing in-context learning with answer feedback for multi-span question answering.

Peng Li, Tianxiang Sun, Qiong Tang, Hang Yan, Yuanbin Wu, Xuanjing Huang, and Xipeng Qiu. 2023. Codeie: Large code generation models are better few-shot information extractors.

Di Lu, Shihao Ran, Joel Tetreault, and Alejandro Jaimes. 2023. Event extraction as question generation and answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1666–1688, Toronto, Canada. Association for Computational Linguistics.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2event: Controllable sequence-to-structure generation for end-to-end event extraction.

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. Association for Computational Linguistics.

OpenAI. 2022. Introducing chatgpt.

OpenAI. 2023. Gpt-4 technical report.

Chaoxu Pang, Yixuan Cao, Qiang Ding, and Ping Luo. 2023. Guideline learning for in-context information extraction.

Hao Peng, Xiaozhi Wang, Feng Yao, Zimu Wang, Chuzhao Zhu, Kaisheng Zeng, Lei Hou, and Juanzi Li. 2023. Omnievent: A comprehensive, fair, and easy-to-use toolkit for event understanding.

Shichao Wang, Xiangrui Cai, HongBin Wang, and Xiaojie Yuan. 2021a. Incorporating circumstances into narrative event prediction. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4840–4849, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaozhi Wang, Hao Peng, Yong Guan, Kaisheng Zeng, Jianhui Chen, Lei Hou, Xu Han, Yankai Lin, Zhiyuan Liu, Ruobing Xie, Jie Zhou, and Juanzi Li. 2023. Maven-arg: Completing the puzzle of all-in-one event understanding dataset with event argument annotation.

Ziqi Wang, Xiaozhi Wang, Xu Han, Yankai Lin, Lei Hou, Zhiyuan Liu, Peng Li, Juanzi Li, and Jie Zhou. 2021b. CLEVE: Contrastive Pre-training for Event Extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*

Dataset	Docs	Events	Triggers	Args
ACE 2005	599	4,090	5,349	9,683
MAVEN-Arg	4,480	98,591	107,507	290,613

Table 2: Details of ACE 2005 and MAVEN-Arg.

and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 6283–6297, Online. Association for Computational Linguistics.

Wenpeng Yin, Qinyuan Ye, Pengfei Liu, Xiang Ren, and Hinrich Schütze. 2023. LLM-driven instruction following: Progresses and concerns. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 19–25, Singapore. Association for Computational Linguistics.

A Dataset Details

Dataset details are shown in Table 2. Data split and preprocessing approach for ACE2005 is consistent with Du and Cardie (2020). Those for MAVEN-Arg followed Wang et al. (2023). MAVEN-Arg is significantly more challenging compared to ACE2005 due to its document-based annotation, much longer text, greater variety of event types, finer classification, and more comprehensive event information within each instance.

B Training Details

We reproduced all models on Nvidia RTX 4090 GPUs and kept the parameters the same as their source codes. Because the focus of our approach is not on the performances of SLMs, but to validate the effectiveness of our method, LC4EE, we did not seek to maintain full consistency with their orginal results. The details of the training are as follows.

EEQA[1] was trained for 6 epochs with batch-size of 8, learning-rate of 4e-5. TEXT2EVENT[2] was trained for 30 epochs with batch-size of 16, learning-rate of 5e-5. CLEVE[3] was trained for 5 epochs with batch-size of 32, learning-rate of with 3e-5.

C Error Analysis

To provide more effective and accurate error feedback information, we conducted error analysis on SLMs prediction results.

[1]https://github.com/xinyadu/eeqa
[2]https://github.com/luyaojie/Text2Event
[3]https://github.com/THU-KEG/OmniEvent

For ease of elaboration in this section, we use data from the event extraction task completed by the EEQA model on the ACE 2005 dataset as an example. The data and methods for other datasets are similar.

We sampled 2,000 instances from the predicted results on the training set, which included 179 error samples in the event detection task and 249 error samples in the event extraction task. We define an error sample as the instance where any mistake occurs in the prediction.

Several error types are summarized after analyzing these error samples. The definition and proportion of error types can be seen in Table 3.

D Description of different granularity levels

In this section, definitions of different granularity levels error feedback are specified with the example of the ED task, which is similar to the EAE task.

Sentence: "After years of intense war, diplomatic talks finally began, but sadly, many soldiers died before peace could be achieved." The original predicted result of SLMs is [["talks", "Contact.Meet"]. Label information is [["war", "Conflict.Attact"], ["died", "Life.Die"]].

Comparing the orginal predicted result with the label information, we can obtain error information. In order to fully explore the correction ability of LLMs based on error feedback, we designed 4 different granularity levels.

L1: Error feedback information is simply "incorrect"

This level of feedback indicates the prediction is incorrect but does not specify how. Given this, we know a correction is needed but lack detailed guidance. The correction relies heavily on the semantic comprehension ability of LLMs.

L2: Error feedback information consists of Error Types and relevant Quantities

For example, {"Prediction omission": 2, "Prediction redundancy": 1}. This indicates that two events were omitted from the predictions, and one event was redundantly predicted. In this case, we omitted "Conflict.Attack" for "war" and "Life.Die" for "died", and redundantly predicted "Contact.Meet" for "talks".

L3: Error feedback information provides incorrect Element information

Element errors indicate incorrect triggers or arguments information, e.g., [["war", "Prediction omission"], ["died", "Prediction omission"], ["talks", "Prediction redundancy"]].

L4: Error feedback information provides Specific Errors. This level provides detailed errors in the predictions, e.g., [[["war", "Conflict.Attack"], "Prediction omission"], [["died", "Life.Die"], "Prediction omission"], [["talks", "Contact.Meet"], "Prediction redundancy"]].

Correction:

Based on L4 feedback, which provides the most detailed information:

The word "war" should trigger a "Conflict.Attack" event, but it was omitted.

The word "died" should indicate a "Life.Die" event, but it was also omitted.

The word "talks" was incorrectly predicted as "Contact.Meet", but given the context, this prediction might be considered redundant because it does not align with the definition of "Contact.Meet" which emphasizes communicating face-to-face.

Corrected Prediction: To correct the prediction according to the L4 feedback:

Add [["war", "Conflict.Attack"]] to capture the event of intense war. Add [["died", "Life.Die"]] to reflect the consequence that many soldiers died. Evaluate the relevance of [["talks", "Contact.Meet"]] in the context; if it aligns with the definition of "Contact.Meet", it may be retained; otherwise, it should be removed from the predicted result.

This correction process highlights the importance of detailed error feedback in refining predictions for event extraction tasks, ensuring more accurate and comprehensive correction of event extraction information from text.

E Specific results of Preliminary Study

Specific results of the preliminary study can be seen in Table 4.

F An example of Rules Inducer

Figure 3 is an example of the Rules Inducer Process for ED task on "Contact.Meet" event type.

According to error analysis, we found that in ED task, predictions of "Contact.Meet" have a high frequency of errors. We selected 8 samples from training set predictions, including correct samples and incorrect samples. Combined with the definition of "Contact.Meet" in the annotation guidelines, we instruct GPT-4 to analyze error reasons and summarizes some rules to generate feedback information.

error type	Event Detection		error type	Event Extraction	
	ACE2005	MAVEN-Arg		ACE2005	MAVEN-Arg
event type error	5.8%	12.1%	role error	1.0%	0.4%
event trigger error	1.2%	3.0%	argument error	13.7%	28.3%
prediction omission	46.1%	25.8%	argument omission	23.2%	31.2%
prediction redundancy	45.7%	45.3%	argument redundancy	24.8%	11.5%
multi-times omission	1.2%	13.8%	event type redundancy	14.1%	13.9%
–	–	–	event type omission	23.2%	14.6%

Table 3: Error analysis results of predictions from EEQA based on ACE2005.

An example of Rules Induction process

There are some examples about Contact.Meet, some of them are correct but some of them are incorrect. In the annotation guidelines, the definition of Contact.Meet is that the event occurs whenever two or more entities come together at a single location and interact with one another face-to-face. Please try to tell me why some examples are incorrect and summarize some rules to tell me how can I find these errors in the predictions. I hope your response is concise.

1. Sentence: Israeli Foreign Minister Silvan Shalom on Wednesday stressed Qatar's role in the Middle East peace process during talks with his Qatari counterpart Sheikh Hamad bin Jassem bin Jabr al - Thani in Paris. Predicted result is [['talks', 'Contact.Meet']]. Label is [['talks', 'Contact.Meet']]. It is correct.

.....

4. Sentence: The Hong Kong Jockey Club is in talks about buying out the horse racing club in neighboring Macau, a newspaper reported Tuesday. Predicted result is [['talks', 'Contact.Meet']]. Label is [ ]. It is incorrect.

.....

To identify correct Contact.Meet events:

1. **Ensure face-to-face interaction** is explicitly described.
2. **Verify the presence of a specific location** for the meeting.
3. **Confirm multiple entities** are involved.
4. **Contextualize 'talks'** to ensure they refer to physical gatherings.

Errors typically arise from vague interactions, lack of location, ambiguous entity involvement, or misuse of 'talks'.

I want you to generate a prompt based on the above rules to determine if the prediction omits the "Contact.Meet" event and if the existing "Contact.Meet" event in the prediction is correct. In the prompt, you are identified as an experienced linguist who is well skilled in the task of event detection. The prompt you generate should contain some demonstrations, so please use your prompt to analyse the samples above to form demonstrations.

[{
"role": "system",
"content": "You are a senior linguist expert in event detection tasks, particularly skilled in identifying 'Contact.Meet' events from sentences. Your expertise enables you to meticulously analyze sentences to determine if a 'Contact.Meet' event is present and correctly predicted. The 'Contact.Meet' event is accurately identified when two or more entities interact face-to-face at a specific location. Your task involves scrutinizing predicted results to confirm the presence of this event and ensuring its accuracy based on syntactic and semantic cues. Remember, the key aspects are face-to-face interaction, specific location, and involvement of multiple entities. Please provide a concise analysis for the following examples, explaining why a prediction is correct or incorrect, based on the guidelines."
},
{
"role": "user",
"content": "The Hong Kong Jockey Club is in talks about buying out the horse racing club in neighboring Macau, a newspaper reported Tuesday. Predicted result is [['talks', 'Contact.Meet']]."
},
{
"role": "assistant",
"content": "{'explanation': 'This prediction is incorrect. Although the sentence mentions 'talks', it lacks explicit mention of a face-to-face meeting or a specific location where this interaction occurs. The context suggests the talks could be metaphorical or not in person, thus not meeting the 'Contact.Meet' event criteria.', 'answer': [ ]}"
},...]

.....

Figure 3: An example of Rules Induction.

In this process, we will give feedback to GPT-4 by pointing out improprieties and inaccuracies of the summarized rules as well as proposing our practical requirements to refine rules iteratively.

In the process of practical application, we found that GPT-4 is able to generate rules for feedback generation and corresponding prompts well, requiring less human interaction and corrections, which reduces the burden on humans.

G Rules for corrector

In this section, after categorizing the rules, we selected a few examples for illustration.

G.1 Rules for ED task

Rules examples for ED task are as follows.

Rule Type 1: Correction for common triggers. Some event types, due to their inherent specificity, have very typical triggers that require minimal or straightforward judgment to complete the event detection task. However, these types, which are quite simple for human annotators, still frequently result in prediction omissions in current SLMs predictions. Moreover, common triggers are more intuitive and easier to understand than the definitions of event types, making judgments by LLMs easier. For example, the word "marriage", when used as an independent noun and not as part of a proper noun like "Marriage Law", typically triggers the "Life.Marry" event. We instruct LLMs to generate prompts to perform keyword matching based on the typical triggers of these event types and then determine if the matched word aligns with the event definition according to semantic information.

Rule Type 2: Correction combining key entity information and event type definitions. Similar to Rule 1, some event types also have key triggers that are easily identified, but these triggers do not always trigger events and require careful judgment in combination with contextual information. Entity information associated with the event can help the LLMs in better understanding semantic information, thereby identifying these events. For example, words derived from "become" and "begin" can trigger "Personnel.Start-Position" event, but it is essential to note that: 1) the subject of the event must be a Person; 2) there must be explicit Position information.

Rule Type 3: Corrections based on the conditions that trigger events. Certain words trigger events under specific conditions, and many prediction results do not meet these conditions. For instance, word derived from "talk" has strict conditions for triggering "Contact.Meet" events, as summarized from the analysis of the annotation guidelines and dataset: 1) the text must explicitly mention two or more entities; 2) the communication must be face-to-face, and the presence of explicit location information in the text can also indicate that the communication is face-to-face.

G.2 Rules for EE task

Rules examples for EE task are as follows.

Rule Type 1: Correction based on event type information corrected in ED task. Firstly, in order to exclude the effects of event type errors, we eliminate a large number of event type redundancy errors in argument extraction based on the corrected event type information in ED task. Then, in response to event type omission errors, we instruct LLMs to generate prompts to find suitable arguments from the text according to the corresponding roles list of the corrected event type defined in the event schema.

Rule Type 2: Correction combined with entity information. During the error analysis process, we found that several roles with high frequent errors in their predicted results. Considering the superior performance of LLMs on Named Entity Recognition (NER) tasks, we guide LLMs to make corrections based on certain entity information. For instance, arguments for role "Destination" of "Movement.Transport" events frequently commits mistakes. Therefore, we instruct LLMs to generate prompts to pay particular attention to "Location", "Geo-Political" and "Facility" entity information in the sentence and to judge whether the "Location" entity is the "Destination" information of the event based on semantic information.

Rule Type 3: Correction based on syntax analysis result. Guiding the LLMs to analyze the syntactic structure of sentences around triggers is crucial for argument extraction. The syntactic components of words aid in enhancing the understanding of semantic information, allowing for better determination of candidate arguments and their corresponding roles. For instance, in the extraction results, arguments for role "Person" and "Entity" or similar roles of certain event types such as "Personnel.Start-Position" are often confused

model	level	Event Detection						Event Extraction					
		ACE2005			MAVEN-Arg			ACE2005			MAVEN-Arg		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1
text2event	–	35.5	43.1	38.9	74.8	19.8	31.3	35.5	44.2	39.4	41.1	32.1	36.1
+GPT-3.5	L1	55.9	21.8	31.4	21.0	64.5	31.6	23.6	31.5	27.0	33.8	9.4	14.7
	L2	55.9	42.5	48.3	22.9	52.0	31.7	35.0	31.5	33.2	41.8	22.8	29.5
	L3	81.4	78.3	79.8	28.3	54.0	37.1	41.1	50.3	45.2	33.9	9.5	14.8
	L4	84.3	89.6	86.9	59.2	91.4	71.8	69.0	75.6	72.2	60.0	28.6	38.7
EEQA	–	40.0	34.2	36.9	68.2	70.8	69.0	43.0	37.1	39.8	40.5	33.6	36.7
+GPT-3.5	L1	50.4	20.6	29.3	69.9	65.6	67.6	25.3	28.6	26.8	37.6	17.4	23.8
	L2	50.4	41.3	45.4	68.1	65.8	66.9	33.8	35.3	34.5	39.5	27.1	32.1
	L3	82.1	80.7	81.4	72.5	69.0	70.7	54.0	56.7	55.3	35.7	14.5	20.7
	L4	90.6	89.1	89.8	80.9	73.0	76.7	76.2	75.9	76.1	56.0	31.8	40.6
CLEVE	–	46.7	57.9	51.7	69.8	44.0	53.9	43.5	56.1	49.0	55.1	29.5	38.4
+GPT-3.5	L1	58.6	30.4	40.0	35.9	57.9	44.3	47.8	41.2	44.2	49.4	13.1	20.7
	L2	69.9	60.4	64.8	41.2	47.9	44.3	44.7	39.2	41.8	54.9	24.3	33.7
	L3	85.0	79.0	81.9	52.0	62.9	56.9	52.1	51.1	51.6	53.6	12.5	20.2
	L4	85.0	90.0	87.2	78.8	85.7	82.1	76.2	82.4	79.0	71.9	30.2	42.5

Table 4: Specific Results of Preliminary Study on EE tasks with different granularity level feedback information.

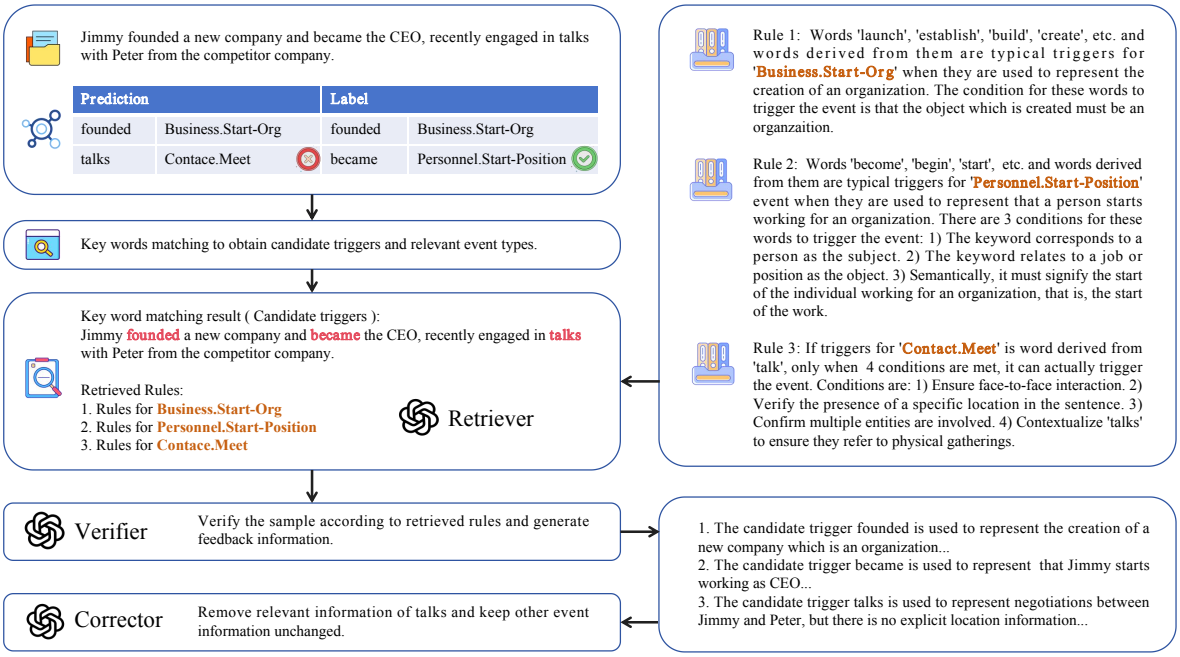

Figure 4: An example of LLMs Corrector.

or commit mistakes. Through syntactic structure analysis, obtaining the subject and object components associated with the trigger, allows for more accurate extraction of arguments.

H An example of LLMs Corrector

In this section, we chose a simple example for illustrating how LLMs Corrector works.

An example of correction on ED task is shown in Figure 4. Detailed description is as follows.

Firstly, based on the input sentence and the candi-

error type	Event Detection		error type	Event Extraction	
	origin	correction		origin	correction
event type error	5.8%	-	role error	1.0%	-
event trigger error	1.2%	-	argument error	13.7%	1.6% (↓0.1%)
prediction omission	46.1%	3.4% (↓0.8%)	argument omission	23.2%	2.7% (↓0.2%)
prediction redundancy	45.7%	3.7% (↓0.4%)	argument redundancy	24.8%	3.0% (↓0.1%)
multi-times omission	1.2%	0.1% (↓0.1%)	event type redundancy	14.1%	1.1% (↓0.7%)
–	–	–	event type omission	23.2%	2.8% (↓0.1%)

Table 5: Correction details for LC4EE on ACE 2005 based on EEQA. "Origin" indicates the proportion of incorrect samples relevant to the error type among all samples. "Correction" indicates the proportion after correction by LC4EE, with the percentage decrease shown in parentheses.

date triggers dictionary analyzed from the training set, LLMs are used to perform keyword matching tasks to obtain the candidate triggers and their corresponding event types. The results obtained are ["founded", "Business.Start_Org"], ["talks", "Contact.Meet"]] and the candidate trigger is "became". Then, using the Retriever, the rules corresponding to the event types of the candidate triggers and the event types in the predicted results are retrieved from the rule repository. If retrieval fails, the predicted results are assumed to be correct by default and keep unchanged. Next, using the Verifier, the retrieved rules are used to verify whether the predicted results are correct and generate corresponding feedback information. In the example, "founded" and its corresponding event type "Business.Start_Org" meet the rule which is used to represent the creation of a new company, indicating the prediction is correct. "became" and its corresponding event type "Personnel.Start_Position" meet the rule, suggesting the start of a person's work, but it is not in the predicted results, indicating an omission in the prediction that should be added. "talks" does not meet the rule because it is unclear whether the conversation is face-to-face, but the relevant information is in the predicted results, so it should be removed. Finally, the Corrector is used to correct predicted results based on the feedback information. The final corrected prediction results corrected by LLMss are [["founded", "Business.Start_Org"], ["became", "Personnel.Start_Position"]].

I Correction details

The correction details for samples used in Table 3 can be seen in Table 5.