Rule-Guided Extraction: A Hierarchical Rule Optimization Framework for Document-Level Event Argument Extraction

Anonymous ACL submission

Abstract

Document-level event argument extraction (EAE) is a critical task in natural language processing. While most prior approaches rely on supervised training with large labeled datasets or resource-intensive fine-tuning, recent studies explore in-context learning (ICL) with LLMs to reduce data dependence and training costs. However, the performance of ICL-based methods still lags behind fully supervised models. We highlight a key reason for this shortfall: the lack of sufficient extraction rules. In this paper, we conduct a systematic study of using hierarchical rules to enhance LLMs' ICL capabilities. We first define three types of hierarchical rules and demonstrate their effectiveness in enhancing the performance of LLMs for document-level EAE. Building on this, we further propose an LLM-driven **H**i**E**rarchical **R**ule **O**ptimization (HERO) framework that iteratively generates and selects optimal hierarchical rules. Specifically, in each iteration, high-value instances are selected to produce error feedback, which is used to update and expand hierarchical rule sets. This results in multiple candidate hierarchical rule sets, from which the optimal one is selected using a scoring-based mechanism. During inference, prompts are constructed using the optimal hierarchical rules to enhance ICL performance of LLMs. Extensive experiments demonstrate the effectiveness of HERO, surpassing few-shot supervised methods and outperforming state-of-the-art prompting baselines by 3.18% F1 on RAMS, 4.30% F1 on DocEE-N, and 3.17% F1 on DocEE-C.

1 Introduction

Document-level event argument extraction (EAE) aims to transform unstructured event information from documents into structured formats encapsulating event arguments, facilitating many downstream tasks such as machine reading comprehension (Han et al., 2021), summarization (Li et al., 2021a), news narrative understanding (Zhang et al., 2022; Keith Norambuena et al., 2023) and dialogue systems (Su et al., 2022). Most of the previous methods heavily depend on supervised training with large-scale labeled data (Du and Cardie, 2020b; Li et al., 2021b; Xu et al., 2022; Ma et al., 2022; He et al., 2023; Liu et al., 2024). Recently, there has been a notable surge in applications of large language models (LLMs) for document-level EAE (Zhu et al., 2024; Zhang et al., 2024; Uddin et al., 2024; Shuang et al., 2024; Hong and Liu, 2024). These methods aim to combine the strengths of SLMs and LLMs or fine-tune LLMs with labeled data. While effective, they still require sufficient labeled data, and fine-tuning LLMs incurs high training costs.

In-Context Learning (ICL) (Brown et al., 2020), a hallmark capability of LLMs, leverages simple task instructions and a few input-output demonstrations within prompts to circumvent traditional resource-intensive requirements. Recent work (Zhou et al., 2024) advances this paradigm by designing heuristic rules to steer LLMs in document-level EAE, effectively bypassing both model tuning and data dependency. However, despite these advantages, the performance of ICL-based methods still lags behind supervised models. We underline a critical bottleneck in current ICL approaches for document-level EAE: the lack of fine-grained extraction rules. This deficiency stems from the inherent complexity of document-level EAE, which typically involves hundreds of event types with diverse, fine-grained types of argument roles, especially, most argument roles may have long spans of non-entity arguments. Without sufficient extraction rules, LLMs frequently exhibit two types of errors: (1) Missing or redundant arguments, (2) Partial argument spans. As illustrated in Figure 1, the document describes a *Protest* event. When extracting the argument role *Location* using coarse-grained rule, the model incorrectly captures vague phrases like *"throughout the United States and around the*

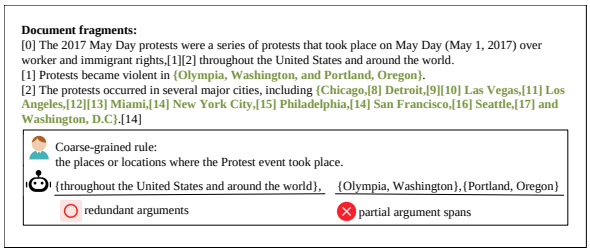

Figure 1: An example of LLMs making errors due to coarse-grained rule for argument role *Location* in the *Protest* event, gold annotations are shown in green.

world". In contrast, annotators label only specific locations rather than broad geopolitical regions. Additionally, LLMs often struggle with accurately extracting argument spans. For example, the model correctly identifies both *"Olympia, Washington"* and *"Portland, Oregon"*, but extracts them as separate spans instead of a single continuous one. The current coarse-grained rule offers little guidance on span boundaries, and even human annotators following this rule may make similar mistakes.

To address the aforementioned challenges, in this work, we explore the use of fine-grained hierarchical rules to guide LLMs for the document-level EAE via ICL. We first define hierarchical rules that mirror human coarse-to-fine reasoning: (1) **Role Semantics Rule** determines candidate arguments based on argument role definitions; (2) **Argument Validity Rule** verifies whether the arguments are contextually valid; (3) **Argument Span Rule** identifies the precise argument spans for extraction. Experimental results demonstrate the effectiveness of these hierarchical rules in enhancing the LLMs' performance on document-level EAE. Based on this, we further propose an LLM-driven **Hi**Erarchical **R**ule **O**ptimization (HERO) framework that iteratively generates and selects optimal hierarchical rules. Specifically, during the rule optimization, we select the high-value instance (i.e., the worst-performing instance corresponding to the worst-performing argument role) to generate detailed error feedback for the LLMs. This feedback is then used to refine the hierarchical rules. By iteratively repeating this process, we produce diverse sets of hierarchical rules. Finally, we score and select optimal hierarchical rules for inference. At inference time, HERO constructs prompts by leveraging all three types of optimal hierarchical rules, improving ICL performance for document-level EAE. Our contributions are as follows:

- We define fine-grained hierarchical rules and empirically demonstrate their effectiveness in improving LLMs' ICL performance on document-level EAE.
- We introduce an LLM-driven hierarchical rule optimization framework that iteratively generates and selects optimal hierarchical rules, leveraging only a small amount of labeled data and requiring no model training.
- We conduct extensive experiments demonstrating the effectiveness of HERO, which outperforms state-of-the-art prompting baselines and few-shot supervised methods on both RAMS and DocEE.

2 Preliminary Study

In this section, we first define fine-grained hierarchical rules and experimentally verify that our proposed rules can effectively guide LLMs and improve their performance on document-level EAE.

2.1 Hierarchical Rules

We define hierarchical rules that emulate the human annotation process from coarse to fine. The Role Semantics Rule (RSR) clarifies argument role definitions to identify candidate arguments. The Argument Validity Rule (AVR) introduces finer-grained constraints and specific conditions to filter and confirm valid arguments. Finally, the Argument Span Rule (ASR) provides detailed boundary guidelines to extract the precise argument spans. Figure 2 illustrates how hierarchical rules guide LLMs through this reasoning process. We provide a detailed description of hierarchical rules in the following sections.

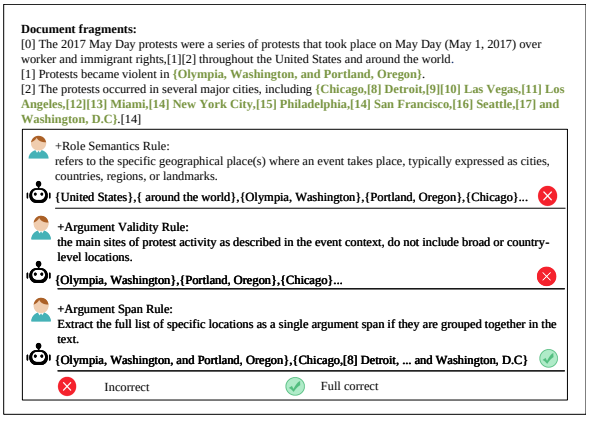

Figure 2: An example of using hierarchical rules to guide LLMs, "+" indicates stepwise rule integration.

2.1.1 Role Semantics Rule

RSR fully explains the meaning of argument roles and the attributes of corresponding arguments, with the aim of guiding LLMs in determining candidate arguments. For example, in the gold annotations of the DocEE dataset (Tong et al., 2022), for the argument role *Arrested*, only the number of arrests is annotated, not the arrest fragments. (e.g., *"four"* instead of *"four people were arrested"*). Therefore, the description *"the number of individuals or groups legally or illegally detained by law enforcement agencies during a protest event"* is more accurate than *"individuals or groups legally or illegally detained by law enforcement agencies during a protest event"*.

2.1.2 Argument Validity Rule

AVR introduces fine-grained rules, including role-specific requirements and constraints, to guide LLMs in accurately identifying valid arguments. Regarding specific requirements, as specified in the RAMS dataset (Ebner et al., 2020), when multiple text spans refer to the same entity, the span closest to the event trigger should be selected. For example, in the sentence *"John and Mary like candy. They ate some today."*, the gold arguments for the argument role *Consumer* is *"They"* rather than *"John and Mary"*. In the DocEE dataset, the argument role *Location* in *Protest* event requires specific physical locations rather than general regions. Furthermore, we collect frequent words or phrases associated with specific argument roles to aid extraction. For instance, *"against"* often appears with the argument role *Protest Reasons*.

2.1.3 Argument Span Rule

ASR provides rules to precisely determine argument spans. As reported in the RAMS dataset, annotators achieve only 55.3% agreement on argument span boundaries, emphasizing the importance of detailed and clear guidelines for argument span extraction. As illustrated in Figure 2, when two valid argument fragments appear consecutively, they should be extracted as a single continuous span (e.g., *"Olympia, Washington and Portland, Oregon"* instead of separating them as *"Olympia, Washington"*, *"Portland, Oregon"*). This challenge has also been identified in prior studies (Han et al., 2024; Zhang et al., 2024), which we attribute to the lack of explicit guidance on argument span extraction.

2.2 Impact of Hierarchical Rules Towards ICL Performance

We first reproduce the results of HD-LoA (Zhou et al., 2024), the current state-of-the-art prompt method for document-level EAE. Based on the reproduction, we select low-performing events: 49 events with 293 instances in RAMS, and 10 events with 110 instances in DocEE-N. For these events, we manually design hierarchical rules for each argument role and use them to prompt LLMs for document-level EAE. Section 4 provides detailed description of the dataset and the LLMs employed in our experiments. As shown in Figure 3, both

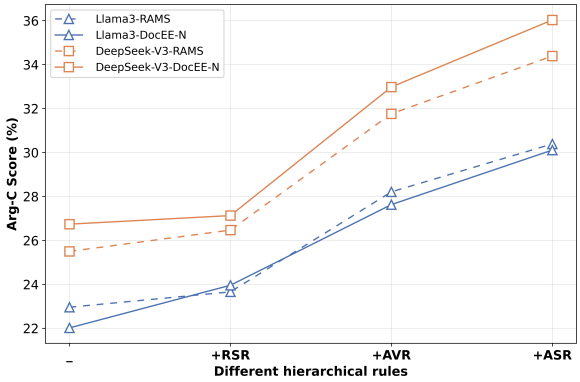

Figure 3: Comparing performance with incrementally added hierarchical rules. "-" indicates use of only the LLM's pretrained knowledge, "+" indicates stepwise rule integration.

LLMs perform suboptimally when relying only on pretrained knowledge. With the gradual integration of hierarchical rules, F1 scores steadily improve on both datasets. We further analyze the impact by comparing redundant roles and exact argument matches under different rule levels. The results, shown in Figure 4, reveal two key trends: (1) Redundancy Reduction: As more hierarchical rules are added, the number of redundant argument roles decreases. Notably, the AVR plays a central role by filtering out contextually irrelevant arguments. (2) Improved Span Accuracy: The number of exactly matched arguments increases with the addition of more hierarchical rules. This is primarily driven by the ASR, which provides detailed guidance for accurate span extraction. These findings suggest that fine-grained hierarchical rules enhance LLMs' ICL performance, and LLMs can effectively interpret and apply such natural language-based hierarchical rules.

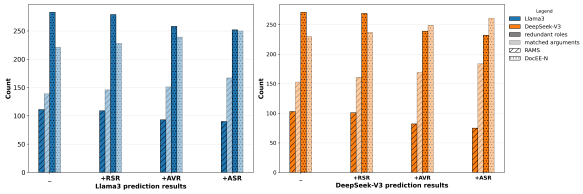

Figure 4: Comparison of redundant roles and matched arguments across hierarchical rule integration levels.

3 Approach

3.1 Task Definition

Given an instance $(X, e, A^{(e)}) \in \mathcal{D}$ with $R^{(e)}$, where X denotes the document context, e indicates the event type, $R^{(e)} = \{r_1, r_2, \ldots, r_{|R^{(e)}|}\}$ is the predefined set of argument roles specific to event type e. $A^{(e)} = \{a^{(r)} \mid r \in R^{(e)}\}$ contains the annotated arguments, where each $a^{(r)} \subseteq X$ is a segmentation of X corresponding to argument role r. Our goal is to extract a set of span $\hat{A}^{(e)}$, each $\hat{a}^{(r)} \in \hat{A}^{(e)}$ is a segmentation of X and represents the arguments about $r \in R^{(e)}$.

3.2 Overview

Section 2 shows that fine-grained hierarchical rules improve LLMs' performance on document-level EAE when detailed and accurate rules are provided. However, designing rules manually is both time-consuming and labor-intensive in practice. To address this, we introduce an LLM-driven hierarchical rule optimization framework that iteratively generates and selects optimal hierarchical rules. Specifically, for each event type e, we begin by randomly selecting an instance and applying the LLMs' pretrained knowledge to perform inference. The predicted results and gold annotations are then fed into an error feedback template to generate the first set of hierarchical rules. We then enter an iterative process: in each iteration, we use the hierarchical rules from the previous round for inference and evaluate their performance scores on a constructed balanced dataset $\mathcal{D}_e$. We select the worst-performing argument role and its lowest-performing instance, derive new error feedback, and refine the rules accordingly. This process repeats for a fixed number of iterations T. Afterward, the optimal hierarchical rules for each argument role are selected based on evaluation scores from all candidates hierarchical rules generated during iteration. During inference, optimal hierarchical rules are used to construct prompts, which guide the LLMs to perform argument extraction. See Figure 5 for an overview.

3.3 Inference

In this section, we introduce how to predict the answer of an instance $(X, e, A^{(e)})$ based on the optimal hierarchical rules. Suppose we have collected the optimal hierarchical rules $\{\mathcal{R}^{(e_i,r)}\}_{r \in R^{(e_i)}} \mid i = 1, \ldots, |N_e|$, where $\{\mathcal{R}^{(e_i,r)}\}$ is the rule set for argument role r in event type e_i, consisting of three types of hierarchical rules introduced before (i,e., $\{\mathcal{R}^{(e_i,r)}\} = \{\mathcal{R}^{(r)}_{\text{RSR}}, \mathcal{R}^{(r)}_{\text{AVR}}, \mathcal{R}^{(r)}_{\text{ASR}}\}$). The task instruction $\mathcal{T}_{test}$, the document content X, examples of a fixed demonstration N, and the optimal hierarchical rules $\{\mathcal{R}^{(e,r)}\}_{r \in R^{(e)}}$ searched for event type e are integrated to create a query prompt P_{test}, it is used to feed into LLMs, which identify the argument roles and their arguments present in X. The inference process unfolds as follows:

$$
\begin{aligned}
P_{\text{test}} &= f\left(\mathcal{T}_{\text{test}}, X, N, \{\mathcal{R}^{(e,r)}\}_{r \in R^{(e)}}\right), \\
\{\hat{A}^{(e,r)}\}_{r \in R^{(e)}} &= \text{Reason}(P_{\text{test}}).
\end{aligned} \tag{1}
$$

3.4 Hierarchical Rule Optimization Framework

In this section, we introduce an iterative hierarchical rule optimization framework, aimed at generating and selecting the optimal hierarchical rules for all argument roles of each event type (i.e., $\{\mathcal{R}^{(e_i,r)}\}_{r \in R^{(e_i)}} \mid i = 1, \ldots, |N_e|$).

Given an event type e with $R^{(e)}$, we construct a balanced dataset $\mathcal{D}_e$ consisting of a few training instances $(X, e, A^{(e)})$ (i.e., with the same or similar number of gold annotations for each argument role). We begin by prompting the LLMs to generate initial rules for each argument role $\{\mathcal{R}_0^{(e,r)}\}_{r \in R^{(e)}}$ using their pretrained knowledge. Then we randomly select an instance from the dataset $\mathcal{D}_e$ and use $\{\mathcal{R}_0^{(e,r)}\}_{r \in R^{(e)}}$ for inference. The predictions $\{\hat{A}^{(e,r)}\}_{r \in R^{(e)}}$ and gold annotations $\{A^{(e,r)}\}_{r \in R^{(e)}}$ are fed into an error feedback template to generate error feedback information. Then we combine the task description for optimizing rules $\mathcal{T}_{opti}$, hierarchical rules description D_{hr}, document content X, error feedback information F_{err}, and current rules $\{\mathcal{R}_0^{(e,r)}\}_{r \in R^{(e)}}$ into a refined hierarchical rules prompt P_{opti}, feed it into LLMs, and obtain updated hierarchical rules $\{\mathcal{R}_1^{(e,r)}\}_{r \in R^{(e)}}$, which includes three types of hierarchical rules for each argument role that we introduced in Section 2.

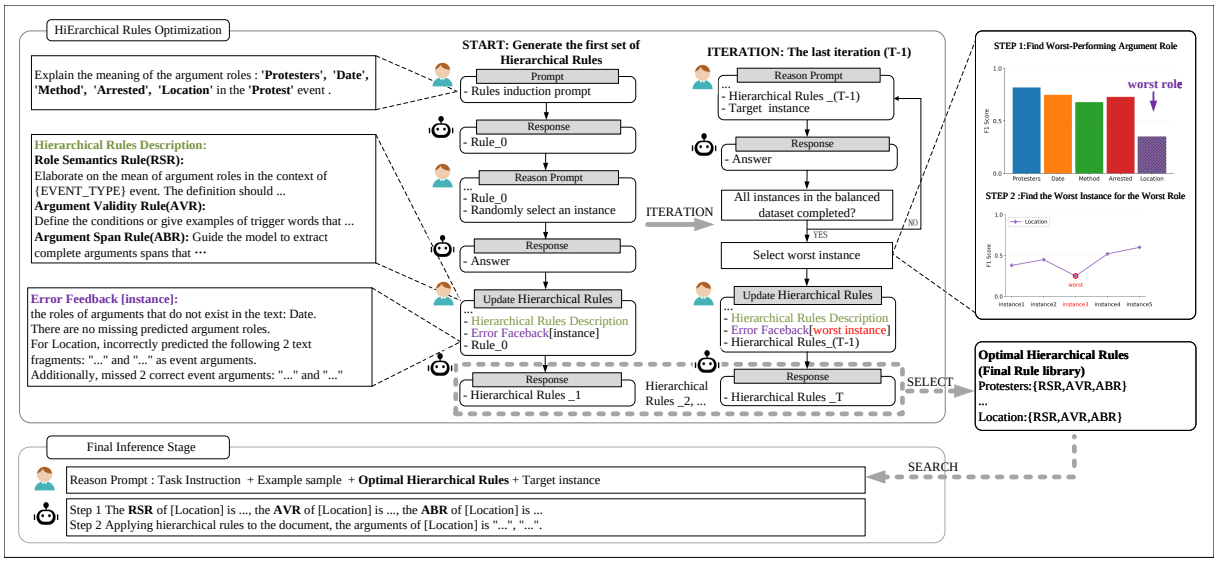

Figure 5: The Hierarchical Rule Optimization Framework, including both inference and optimization phases. Rule_0 denotes the initial rules derived from the LLMs' pretrained knowledge, corresponding to $\{\mathcal{R}_0^{(e,r)}\}_{r\in R^{(e)}}$ in the paper.

This process unfolds as follows:

$$P_{opti} = f\left(\mathcal{T}_{opti}, D_{hr}, X, F_{err}, \{\mathcal{R}_0^{(e,r)}\}_{r\in R^{(e)}}\right),$$
$$\{\mathcal{R}_1^{(e,r)}\}_{r\in R^{(e)}} = \text{Reason}(P_{opti}). \tag{2}$$

At this point, we obtain the first set of hierarchical rules. Next, we iteratively refine hierarchical rules by selecting the high-value instance to generate error feedback. Specifically, at iteration t, we evaluate the hierarchical rules $\{\mathcal{R}_t^{(e,r)}\}$ for each argument role $r \in R^{(e)}$ on dataset $\mathcal{D}_e$. The score for each role's rules on $\mathcal{D}_e$ is computed as follows:

$$s(\mathcal{R}_t^{(e,r)}, \mathcal{D}_e) = 2*P_{\mathcal{D}_e}^{(r)}*R_{\mathcal{D}_e}^{(r)}/(P_{\mathcal{D}_e}^{(r)}+R_{\mathcal{D}_e}^{(r)}), \tag{3}$$

where $P_{\mathcal{D}_e}^{(r)}$ and $R_{\mathcal{D}_e}^{(r)}$ represent the accuracy and recall of the role r. We calculate the score $s(\mathcal{R}_t^{(e,r)}, X_i)$ of each argument role's current rules $\{\mathcal{R}_t^{(e,r)}\}_{r\in R^{(e)}}$ on the instance $(X_i, e, A^{(e)})$ as follows:

$$s(\mathcal{R}_t^{(e,r)}, X_i) = 2*P_i^{(r)}*R_i^{(r)}/(P_i^{(r)}+R_i^{(r)}). \tag{4}$$

At the end of the t-th iteration, we select the worst-performing argument role as follows:

$$r_{\text{worst}} = \arg\min_{r\in R^{(e)}} s(\mathcal{R}_t^{(e,r)}, \mathcal{D}_e), \tag{5}$$

and then choose the corresponding worst-performing instance for that argumen role based on the score of $s(\mathcal{R}_t^{(e,r)}, X_i)$ as follows:

$$i_{\text{worst}} = \arg\min_{i\in\{1,\ldots,m\}} s(\mathcal{R}_t^{(e,r_{\text{worst}})}, X_i), \tag{6}$$

where m represents the number of instances in the balanced dataset $\mathcal{D}_e$. The predictions and gold annotations are fed back to regenerate hierarchical rules $\mathcal{R}_{t+1}^{(e,r)}{}_{r\in R^{(e)}}$. The iteration stops when the maximum iteration T is reached. In all experiments, we consider the number of iterations T is 3. At the end of all iterations, we select optimal hierarchical rules as follows:

$$\mathcal{R}_{\text{final}}^{(e,r)} = \arg\max_{\substack{t\in\{1,\ldots,T\}\\ r\in\mathcal{R}^{(e)}}} s(\mathcal{R}_t^{(e,r)}, \mathcal{D}_e). \tag{7}$$

4 Experiments

4.1 Experimental Setup

Datasets Follow previous work (Zhou et al., 2024), we conduct experiments on two common datasets in the document-level EAE: RAMS (Ebner et al., 2020) and DocEE (Tong et al., 2022) datasets. RAMS and DocEE contain 139 and 59 event types, 65 and 356 argument role types, respectively, with a total of 9,124 and 27,485 documents. DocEE offers two evaluation settings: DocEE-N (Normal), where event types in the training and test sets are the same, and DocEE-C (Cross), where they are disjoint. We leave the detailed statistics of the datasets in Appendix A.1.

Method	PLMs	Paradigm	RAMS		DocEE	
			Arg-I	Arg-C	Normal (Arg-C)	Cross (Arg-C)
EEQA (Du and Cardie, 2020b)	BERT-large	SFT	40.09	34.76	-	-
BART-Gen (Li et al., 2021b)	BART-large	SFT	40.77	35.51	-	-
TSAR (Xu et al., 2022)	BERT-large	SFT	44.13	38.88	-	-
PAIE (Ma et al., 2022)	BART-large	SFT	47.02	40.75	-	-
TabEAE (He et al., 2023)	BART-large	SFT	47.83	41.07	-	-
SCPRG (Liu et al., 2023)	BERT-large	SFT	45.96	39.73	-	-
DEEIA (Liu et al., 2024)	BART-large	SFT	47.14	40.41	-	-
ULTRA (Zhang et al., 2024)	Flan-UL2	FT	-	-	-	32.70*
CQWS (Uddin et al., 2024)	T5, GPT-4, BART-base	FT	-	45.20*	-	-
GLM2-6B (Shuang et al., 2024)	ChatGLM2-6B	FT	**50.90***	**45.80***	-	-
RLQG (Hong and Liu, 2024)	Llama2-7B, 13B	FT	-	19.61*	-	-
Standard (Agrawal et al., 2022)	Llama3	ICL	42.17	32.99	25.71	25.98
CoT (Wei et al., 2022)	Llama3	ICL	40.39	33.44	26.13	27.47
HD-LoA (Zhou et al., 2024)	Llama3	ICL	44.23	36.06	27.36	28.91
HERO (ours)	Llama3	ICL	43.47	37.21	30.52	31.90
Standard (Agrawal et al., 2022)	DeepSeek-V3	ICL	41.29	32.85	26.74	28.09
CoT (Wei et al., 2022)	DeepSeek-V3	ICL	44.33	36.24	28.07	30.34
HD-LoA (Zhou et al., 2024)	DeepSeek-V3	ICL	47.49	40.85	30.51	32.97
HERO (ours)	DeepSeek-V3	ICL	50.49	44.03	**34.81**	**36.14**

Table 1: Overall performance. We highlight the best result and underline the second best. * means the value from the original paper. FT, SFT, and PLMs respectively refer to fine-tuning, supervised fine-tuning, and pretrained language models.

Evaluation Metric Follow the metrics in (Ma et al., 2022; Zhou et al., 2024), we adopt two evaluation metrics. (1) Argument Identification F1 score (Arg-I): an event argument is correctly identified if its offsets and event type match those of any of the argument mentions. (2) Argument Classification F1 score (Arg-C): an event argument is correctly classified if its role type is also correct.

Baselines Our HERO is compared against several state-of-the-art models in three categories: (1) Three state-of-the-art prompting methods without fine-tuning, including the **Standard Prompting** (Agrawal et al., 2022) used in clinical EAE, the **Chain-of-Thought (CoT)** prompt (Wei et al., 2022), and **HD-LoA** (Zhou et al., 2024), which is the best prompting method currently tailored for the document-level EAE. (2) Four latest methods based on LLMs fine-tuning, **RLQG** (Hong and Liu, 2024), **ULTRA** (Zhang et al., 2024), **GLM2-6B** (Shuang et al., 2024), **CQWS** (Uddin et al., 2024). (3) Various supervised learning methods, including: two SOTA span-based methods, **TSAR** (Xu et al., 2022) and **SCPRG** (Liu et al., 2023); two typical generation-based methods, **EEQA** (Du and Cardie, 2020b), **BART-Gen** (Li et al., 2021b); three prompt-based approaches, **PAIE** (Ma et al., 2022), **TabEAE** (He et al., 2023), **DEEIA** (Liu et al., 2024). For fair comparison, all supervised learning methods comparison results are reproduced using the same data as our method. Our HERO uses approximately five labeled instances per event type for rule optimization. This results in a total of 645 instances on RAMS, 310 instances on DocEE-N, and 50 instances on DocEE-C. More details of baselines are listed in Appendix A.3. We also compare our HERO with supervised learning methods that trained on the entire dataset in Appendix B.

Implementation Details Please refer to Appendix A.2 for implementation details of HERO. All prompts used in this paper are provided in the appendix C.

LLM In our experiments, we employ two LLMs: Llama-3-8B-Instruct (Llama3) (Grattafiori et al., 2024) and DeepSeek-V3 (DeepSeek-AI et al., 2024). Each LLM independently optimizes its own hierarchical rules and subsequently uses these rules to guide inference.

4.2 Overall Experimental Results

The main experimental results presented in Table 1. We can observe that: (1) Our method is better than all SFTs and 2.96% (44.03% vs. 41.07%) higher than the best SFT method TabEAE on the RAMS. Notably, TabEAE requires a large num-

ber of training iterations, whereas our method achieves superior performance without any fine-tuning. (2) HERO consistently outperforms all three existing prompting methods. Compared to the strongest baseline HD-LoA, our approach achieves F1 improvements of 3.18% (44.03% vs. 40.85%), 4.30% (34.81% vs. 30.51%), and 3.17% (36.14% vs. 32.97%) on the RAMS, DocEE-N, and DocEE-C datasets, respectively. (3) Compared with fine-tuned LLM-based methods, our approach performs slightly lower than GLM2-6B and CQWS on RAMS by 1.77% (44.03% vs. 45.80%) and 1.17% (44.03% vs. 45.20%), respectively. However, these methods depend on large labeled datasets and costly fine-tuning, while our approach needs only a small amount of data and no training, yet achieves competitive performance.

4.3 Ablation Study

To evaluate the effectiveness of each proposed module, an ablation study is conducted as follow: we compare four settings: (1) using only RSR, (2) using both RSR and AVR, (3) replacing all three hierarchical rule types with R-base (rules derived solely from the LLMs' pretrained knowledge), and (4) HERO-r, a variant of our method that generates feedback using randomly selected instances instead of the worst-performing ones. Results are shown in Table 2.

Method	RAMS		DocEE	
	Arg-I	Arg-C	Normal (Arg-C)	Cross (Arg-C)
HERO	50.49	44.03	34.81	36.14
HERO-r	50.17	42.49	33.05	35.69
R-base	45.99	38.35	29.13	30.04
RSR,AVR	50.24	43.77	33.21	34.75
RSR	49.07	42.11	32.46	32.08

Table 2: Ablation study results

Experimental results demonstrate that: (1) focusing error feedback generation on the most high-value instances yields consistent F1 gains of 1.54% (44.03% vs. 42.49%), 1.76% (34.81% vs. 33.05%), and 0.45% (36.14% vs. 35.69%) on the RAMS, DocEE-N, and DocEE-C datasets, respectively. This indicates that focusing on challenging instances where LLMs often fail provides more effective supervision, while random instances offer limited gains. (2) When relying solely on pre-training knowledge (i.e., R-base), model performance consistently declined across all datasets, with up to a 3.76% (42.11% vs. 38.35%) drop on RAMS compared to using only RSR rules. This highlights the effectiveness of RSR rules refined through error feedback. Removing ASR and then AVR results in a stepwise performance drop across all datasets, confirming that each LLM-generated hierarchical rules contributes valuable guidance.

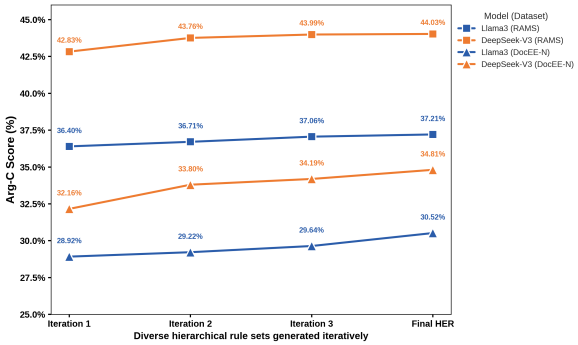

Figure 6: Performance comparison between hierarchical rules from different iterations and the optimal hierarchical rules selected by our method.

5 Analysis

5.1 Results from Different Iterations

To evaluate the effectiveness of our score-based hierarchical rules selection strategy, we compare the performance of hierarchical rules from each iteration with the final optimal rules selected for each argument role, as shown in Figure 6. Results show a consistent performance gain across iterations on both datasets and models, with the best results achieved using our selected optimal rules. Remarkably, for DeepSeek-V3, the rule refined from just a single instance in the first iteration already surpasses HD-LoA on both datasets. This demonstrates that even a single round of feedback is effective, as the three types of hierarchical rules provide specific guidance.

5.2 Error Analysis and Case Study

We randomly select 50 instances from the DocEE-N dataset, covering a total of 286 argument roles, and conduct a manual analysis of the prediction errors. The findings are summarized in Table 3. The most common error type (43.89%) is role redundancy, where the model predicts arguments such as *Arrest_time* and *Arrest_location* that are not present in the gold annotations. However, manual review confirms that these arguments are correctly, suggesting that some gold annotations in DocEE-N are incomplete. The second most frequent error (40%) involves partial spans, where the predicted spans

Error Type	Example
Role redundancy (43.89%)	[. . .]The US citizen was detained at [Karachi airport]PREDICTED_Arrest Location after security staff found 15 bullets for a 9mm handgun [. . .] when he was detained [on Monday night]PREDICTED_Arrest time.[. . .]
Partial spans (40%)	[. . .]An agent with the US FBI has been [arrested under anti-terrorism laws in Pakistan for [carrying ammunition while trying to board a flight]GOLD_The Charged Crime]PREDICTED_The Charged Crime.[. . .]
Wrong spans (10.56%)	[. . .][Three suicide attacks have hit an island on Lake Chad]PREDICTED_Attacker, killing at least 27 people, [. . .] but the region is under a state of emergency after attacks by the [Boko Haram militant group]GOLD_Arrest time. [. . .]

Table 3: Most common errors made by HERO on DocEE-N.

are either too short or too long. Although our ASR is designed to improve argument span precision, the model still struggles with instances that lack clear span boundaries. Wrong spans are relatively rare (10.56%), suggesting that the model seldom assigns arguments to unrelated text.

6 Related Work

6.1 Document-level EAE

Existing methods for document-level EAE can be classified into four main categories: (1) Span-based methods, which identify candidate spans and subsequently predict their roles (Xu et al., 2022; Liu et al., 2023; Yang et al., 2023). (2) Generation-based methods (Du and Cardie, 2020b; Li et al., 2021b; Du et al., 2022; Hsu et al., 2023; Zhang et al., 2023) utilize prompt templates and transformer-based encoder-decoder frameworks to extract the arguments within each event sequentially. (3) Prompt-based methods (Ma et al., 2022; He et al., 2023; Liu et al., 2024), which use slotted prompts and leverage a generative slot-filling approach for argument extraction. (4) Large language models methods. (Zhu et al., 2024) proposed an LLM-based error correction framework that corrects SLM predictions using feedback generated by LLMs. In parallel, (Shuang et al., 2024) explored LLM-generated external event-related features as supplementary context. Another line of work ((Hong and Liu, 2024; Uddin et al., 2024)) developed various question generation strategies to systematically interrogate event arguments through fine-tuned models. (Zhang et al., 2024) focused on enhancing argument span extraction via specialized LLM fine-tuning.

6.2 In-context learning

Constrained by the difficulties of fine-tuning LLMs, in-context learning (ICL) is proposed to emulate few-shot learning by providing several labeled examples in the prompt (Brown et al., 2020) However, LLMs are sensitive to the quality of in-context example (Liu et al., 2022). Recent studies highlight various strategies for selecting in-context examples, including complexity (Fu et al., 2023), diversity (Zhang et al.), and labeled data (Shum et al., 2023). (Fu et al., 2024) conduct the first study to explore this problem in EAE, focusing exclusively on sentence-level scenarios. Recent work (Zhou et al., 2024) addresses this challenge by introducing Heuristic-Driven Link-of-Analogy (HD-LoA) prompting, which uses heuristic rules to guide LLMs in performing EAE via ICL. However, its coarse-grained rules limit performance on complex document-level cases. We extend this line of work by proposing fine-grained hierarchical rules that offer more precise guidance.

7 Conclusion

In this paper, we explore guiding LLMs to perform document-level EAE through ICL using fine-grained hierarchical rules. We begin by defining three types of hierarchical rules and validating their effectiveness. Building on this, we propose a hierarchical rule optimization framework that requires only a small amount of labeled data. By iteratively selecting the high-value instances to generate error feedback, the framework refines and selects optimal hierarchical rules without any parameter tuning. Experimental results on RAMS and DocEE demonstrate the effectiveness of our approach.

Limitations

As our primary goal is to validate the effectiveness of our method. However, our work still lacks depth in certain aspects, and many potential research directions within this framework warrant further investigation.

Enhancing LLMs' Rule-Following Ability While we have demonstrated that LLMs can be prompted to induce and apply hierarchical rules, we have not explicitly measured the extent to which they effectively follow these rules. Improving LLMs' ability to accurately apply rules could further boost the performance of our method.

Broader Application Our HERO framework is designed to be general and can be adapted to other tasks with limited labeled data. In this paper, we focus solely on document-level EAE, but future work will explore applying HERO to other complex information extraction tasks, such as relation extraction.

References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are few-shot clinical information extractors. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1998–2022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

A Liu DeepSeek-AI, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, page 4.

Xinya Du and Claire Cardie. 2020a. Document-level event role filler extraction using multi-granularity contextualized encoding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8010–8020.

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

Xinya Du, Sha Li, and Heng Ji. 2022. Dynamic global memory for document-level argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5264–5275.

Seth Ebner, Patrick Xia, Ryan Culkin, Kyle Rawlins, and Benjamin Van Durme. 2020. Multi-sentence argument linking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8057–8077.

Yanhe Fu, Yanan Cao, Qingyue Wang, and Yi Liu. 2024. Tise: A tripartite in-context selection method for event argument extraction. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1801–1818.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. In *11th International Conference on Learning Representations, ICLR 2023*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Ridong Han, Chaohao Yang, Tao Peng, Prayag Tiwari, Xiang Wan, Lu Liu, and Benyou Wang. 2024. An empirical study on information extraction using large language models. *arXiv e-prints*, pages arXiv–2409.

Rujun Han, I-Hung Hsu, Jiao Sun, Julia Baylon, Qiang Ning, Dan Roth, and Nanyun Peng. 2021. Ester: A machine reading comprehension dataset for reasoning about event semantic relations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7543–7559.

Yuxin He, Jingyue Hu, and Buzhou Tang. 2023. Revisiting event argument extraction: Can eae models learn better when being aware of event co-occurrences? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12542–12556.

Zijin Hong and Jian Liu. 2024. Towards better question generation in qa-based event extraction. In *ACL (Findings)*.

I-Hung Hsu, Zhiyu Xie, Kuan-Hao Huang, Prem Natarajan, and Nanyun Peng. 2023. Ampere: Amr-aware prefix for generation-based event argument extraction model. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

Brian Felipe Keith Norambuena, Tanushree Mitra, and Chris North. 2023. A survey on event-based news narrative extraction. *ACM Computing Surveys*, 55(14s):1–39.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient

memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626.

Manling Li, Tengfei Ma, Mo Yu, Lingfei Wu, Tian Gao, Heng Ji, and Kathleen McKeown. 2021a. Timeline summarization based on event graph compression via time-aware optimal transport. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6443–6456.

Sha Li, Heng Ji, and Jiawei Han. 2021b. Document-level event argument extraction by conditional generation. In *2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021*, pages 894–908. Association for Computational Linguistics (ACL).

Jiachang Liu, Dinghan Shen, Yizhe Zhang, William B Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for gpt-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114.

Wanlong Liu, Shaohuan Cheng, Dingyi Zeng, and Qu Hong. 2023. Enhancing document-level event argument extraction with contextual clues and role relevance. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12908–12922.

Wanlong Liu, Li Zhou, Dingyi Zeng, Yichen Xiao, Shaohuan Cheng, Chen Zhang, Grandee Lee, Malu Zhang, and Wenyu Chen. 2024. Beyond single-event extraction: Towards efficient document-level multi-event argument extraction. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 9470–9487.

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022. Prompt for extraction? paie: Prompting argument interaction for event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6759–6774. Association for Computational Linguistics.

Kai Shuang, Zhouji Zhouji, Wang Qiwei, and Jinyu Guo. 2024. Thinking about how to extract: Energizing llms' emergence capabilities for document-level event argument extraction. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 5520–5532.

Kashun Shum, Shizhe Diao, and Tong Zhang. 2023. Automatic prompt augmentation and selection with chain-of-thought from labeled data. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12113–12139.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2022. Multi-task pre-training for plug-and-play task-oriented dialogue system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4661–4676.

MeiHan Tong, Bin Xu, Shuai Wang, Meihuan Han, Yixin Cao, Jiangqi Zhu, Siyu Chen, Lei Hou, and Juanzi Li. 2022. Docee: a large-scale and fine-grained benchmark for document-level event extraction. Association for Computational Linguistics.

Md Nayem Uddin, Enfa George, Eduardo Blanco, and Steven Corman. 2024. Generating uncontextualized and contextualized questions for document-level event argument extraction. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5612–5627.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jing Xu, Dandan Song, Siu Hui, Zhijing Wu, Meihuizi Jia, Hao Wang, Yanru Zhou, Changzhi Zhou, and Ziyi Yang. 2024. Separation and fusion: A novel multiple token linking model for event argument extraction. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6611–6624.

Runxin Xu, Peiyi Wang, Tianyu Liu, Shuang Zeng, Baobao Chang, and Zhifang Sui. 2022. A two-stream amr-enhanced model for document-level event argument extraction. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5025–5036.

Yuqing Yang, Qipeng Guo, Xiangkun Hu, Yue Zhang, Xipeng Qiu, and Zheng Zhang. 2023. An amr-based link prediction approach for document-level event argument extraction. In *The 61st Annual Meeting Of The Association For Computational Linguistics*.

Kaihang Zhang, Kai Shuang, Xinyue Yang, Xuyang Yao, and Jinyu Guo. 2023. What is overlap knowledge in event argument extraction? ape: A cross-datasets transfer learning model for eae. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 393–409.

Xinliang Frederick Zhang, Nick Beauchamp, and Lu Wang. 2022. Generative entity-to-entity stance detection with knowledge graph augmentation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9950–9969.

Xinliang Frederick Zhang, Carter Blum, Temma Choji, Shalin Shah, and Alakananda Vempala. 2024. Ultra: Unleash llms' potential for event argument extraction through hierarchical modeling and pair-wise self-refinement. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 8172–8185.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations*.

Hanzhang Zhou, Junlang Qian, Zijian Feng, Lu Hui, Zixiao Zhu, and Kezhi Mao. 2024. Llms learn task heuristics from demonstrations: A heuristic-driven prompting strategy for document-level event argument extraction. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11972–11990.

Mengna Zhu, Kaisheng Zeng, JibingWu JibingWu, Lihua Liu, Hongbin Huang, Lei Hou, and Juanzi Li. 2024. Lc4ee: llms as good corrector for event extraction. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 12028–12038.

A Dataset and Model

A.1 Dataset statistics

RAMS is a document-level dataset annotated with 139 event types and 65 semantic roles. Each sample is a 5-sentence document, with a trigger word indicating a pre-defined event type and its argument scattered throughout the document. DocEE contains 27,485 document-level events with 180,528 arguments, including 59 event types involving 356 fine-grained argument roles, far exceeding the scale of existing document-level EE datasets. Considering the extensive size of the DocEE dataset, which makes a full-scale evaluation using LLMs impractical, we follow (Zhou et al., 2024) and evaluate a subset of these datasets. The detailed statistics of the datasets and the number of tested samples are listed in Table 4.

A.2 Implementation Details

We evaluate HERO on Llama-3-8B-Instruct (Llama3) (Grattafiori et al., 2024) and DeepSeek-V3 (DeepSeek-AI et al., 2024). The Llama3 model checkpoint is publicly accessible on Hugging Face under the Llama3 Community License Agreement. We deploy Llama3 on a single 32GB virtual GPU environment, and use the vLLM (Kwon et al., 2023) to accelerate the inference. We employ DeepSeek-V3 from the official API. The pricing for running DeepSeek-V3 is RMB 0.008 per 1,000 tokens. During the all experiments, the temperature is fixed as 0.

A.3 Details of baseline models

We compare our model with following previous models. (1) **EEQA** (Du and Cardie, 2020b): the first Question Answering (QA) based model designed for the sentence-level EAE task. (2) **BART-Gen** (Li et al., 2021b): a conditional generation model generating (rather than recognizing the spans) arguments sequentially via a sequence-to-sequence model and prompt. (3) **TSAR** (Xu et al., 2022): a two-stream encoding model for document-level EAE, which encodes the document through two different perspectives to better utilize the context. (4) **PAIE** (Ma et al., 2022): a prompt tuning paradigm for extraction tasks which prompts multiple role knowledge from PLMs via role-specific selectors and joint prompts. (5) **TabEAE** (He et al., 2023): a novel text-to-table framework, that can extract multiple event in parallel. (6) **SCPRG** (Liu et al., 2023): A span-trigger-based contextual pooling and latent role guidance method. (7) **DEEIA** (Liu et al., 2024): a Multi-EAE model that overcomes the inefficiency limitations of traditional EAE methods. The implementation details of all baselines are as follows:

1. **EEQA** (Du and Cardie, 2020b): We use BERT-large and their publicly available code to test its performance on the RAMS dataset. The question template is: What is the ROLE in TRIGGER WORD?

2. **BART-Gen** (Li et al., 2021b) : We used the BART-large model and their publicly available code on the RAMS dataset with the same training data as ours.

3. **TSAR** (Xu et al., 2022): We use BERT-large and their publicly available code to test its performance on the RAMS dataset.

4. **PAIE** (Ma et al., 2022): We use BART-large and their publicly available code to test its performance on the RAMS dataset.

5. **TabEAE** (He et al., 2023): We use BART-large and their publicly available code to test its performance on the RAMS dataset, and we use their proposed single-to-single pattern.

Dataset	# EvTyp.	# ArgTyp.	# TrainDoc.	# DevDoc.	# TestDoc.	# Eval.
RAMS (Ebner et al., 2020)	139	65	7,329	924	871	871
DocEE (Normal) (Tong et al., 2022)	59	356	22k	2.7K	2.7K	400
DocEE (Cross) (Tong et al., 2022)	59	356	23.7K	1.6K	2.0K	400

Table 4: Datasets Overview (# EvTyp.: event type, # ArgTyp.: event argument type, # TrainDoc.: the number of documents in the training set, # DevDoc.: the number of documents in the validation set, # TeseDoc.: the number of documents in the test set, # Eval.:the number of samples used for evaluation of different methods of LLMs without fine-tuning.)

6. **SCPRG** (Liu et al., 2023): We use BERT-large and their publicly available code to test its performance on the RAMS dataset.

7. **DEEIA** (Liu et al., 2024): We use BART-large and their publicly available code to test its performance on the RAMS dataset.

We also compare with several fine-tuned LLMs methods, the specific implementations are as follows. (1) **RLQG** (Hong and Liu, 2024): They fine-tune LLaMA-2-7B to generate questions and use LLaMA-2-13B-Chat to answer them. (2) **ULTRA** (Zhang et al., 2024): This approach fine-tunes Flan-UL2 to address the issue of LLMs failing to accurately extract argument spans in document-level EAE. (3) **GLM2-6B** (Shuang et al., 2024): They perform instruction tuning on GLM2-6B using the document-level EAE dataset RAMS to better adapt the model to the document-level EAE. (4) **CQWS** (Uddin et al., 2024): They fine-tune T5 for question generation using weakly supervised data generated by GPT-4, and further fine-tune BART-base for question answering.

The implementation details of methods without fine-tuning LLMs are as follows. (1) **Standard Prompting** (Agrawal et al., 2022): We follow the method and prompts described in their paper, given the document and the specified target argument roles, we then directly extracts the corresponding argument spans from the document. (2) **CoT** (Wei et al., 2022): We follow the method and prompts described in their paper. Given a document and the specified target argument roles, we provide a demonstration that includes the correct answers and a detailed reasoning process to illustrate the extraction. The model then directly extracts the corresponding argument spans from the document. (3) **HD-LoA** (Shuang et al., 2024): Our implementation strictly adheres to the publicly available code provided by the authors. We replicate the experiments by employing different LLMs to assess the generalizability of the approach.

B Comparison with Fully Supervised Methods

We compare our HERO method with fully supervised models trained on the entire dataset for the document-level EAE task. As shown in Table 5, these models, trained on thousands of annotated data, achieve strong performance on datasets such as RAMS and DocEE-N. In contrast, HERO does not require any training. Instead, it relies on a small number of labeled data to induce guiding rules, which are then used to perform event argument extraction. While there remains a performance gap on RAMS and DocEE-N, HERO achieves a significant improvement of 6.34% (36.14% vs. 29.80%) over supervised models on the DocEE-C dataset. This demonstrates the effectiveness of our approach in cross-domain and low-resource scenarios, where large-scale annotation is impractical or unavailable.

C Full Prompts

We present all the prompts mentioned in the paper.

1. Table 6 presents a complete prompt used to guide LLMs in generating initial rules for argument roles based on their pretrained knowledge.

2. Table 7 presents a complete example of an error feedback prompt, illustrating the detailed error information used to guide rule refinement.

3. Table 8 presents the complete prompt for refined hierarchical rules incorporating error feedback.

4. Table 9 presents the complete prompt used for final testing on the RAMS dataset with the optimal hierarchical rules.

5. Table 10 presents the complete prompt used for final testing on the DocEE dataset with the optimal hierarchical rules.

Method	Model	RAMS		DocEE	
		Arg-I	Arg-C	Normal (Arg-C)	Cross (Arg-C)
Supervised learning	EEQA(Du and Cardie, 2020b)	48.70	46.70	33.50	24.00
	MG-Reader (Du and Cardie, 2020a)	-	-	32.90	21.40
	BART-Gen (Li et al., 2021b)	51.20	47.10	-	-
	OntologyQA (Tong et al., 2022)	-	-	41.00	29.80
	TSAR (Xu et al., 2022)	-	51.18	-	-
	PAIE (Ma et al., 2022)	56.80	52.20	-	-
	SCPRG (Liu et al., 2023)	-	52.32	-	-
	TabEAE (He et al., 2023)	57.30	52.70	-	-
	DEEIA (Liu et al., 2024)	58.00	53.40	-	-
	EAESR (Shuang et al., 2024)	60.20	53.20	-	-
	Sep2F(Xu et al., 2024)	58.70	53.70	-	-
Llama3	HERO (ours)	43.47	37.21	30.52	31.90
DeepSeek-V3	HERO (ours)	50.49	44.03	34.81	36.14

Table 5: Comparison with Fully Trained Supervised Models.

Objective: Your task is Event Argument Extraction. In this task, you will be provided with a document that describes an event and the goal is to extract the event arguments that correspond to each argument role associated with the event. The terminologies for this task is as follows: Key Terminologies: Event argument: an entity mention, temporal expression or value that serves as a participant or attribute with a specific role in an event. Event arguments should be quoted exactly as the it appears in the given document. Argument role: the relationship between an argument to the event in which it participates. Specifically, based on your existing knowledge, provide explanations for the argument roles. These explanations will be regarded as argument extraction rules and should be presented in the following format:

Rule output format:

Role Semantics Rule
[Argument Role]: "Role Semantics Rule"
Argument Validity Rule
[Argument Role]: "Argument Validity Rule"
Argument Span Rule
[Argument Role]: "Argument Span Rule"

Table 6: Prompt for inducing LLMs to generate initial rules based on pretrained knowledge.

Feedback information: You made a mistake in predicting the roles of arguments that do not exist in the text. You need to focus on updating the Argument Validity Rule for these argument roles: Date.
There are no missing predicted argument roles.
Argument roles are predicted correctly but event arguments are wrong. Details:
For Age, you incorrectly predicted the following 1 text fragments: "38 years old" as event arguments.
Additionally, you missed 1 correct event arguments: "38-year-old"
For Location/Hospital, you incorrectly predicted the following 2 text fragments: "Santiago" and "Nuevo Leon" as event arguments.
Additionally, you missed 2 correct event arguments: "a local road in the municipality of Santiago, some 30 km (18.6 miles) away from Monterrey" and "a rural road early on Wednesday outside his town of Santiago"
For Perpetrator, you incorrectly predicted the following 1 text fragments: "drug cartels" as event arguments.

Table 7: A complete example of an error feedback prompt.

Objective: Your task is to further refine the three corresponding rules for each argument role based on the predicted results and real annotations of the current example. In the process of refining rules, you need to pay attention to two points:

The meanings of three fine-grained hierarchical rules: 1.Each argument role has three rules, and the definition and objectives of each rule are as follows:

Role Semantics Rule: Elaborate on the mean of argument roles in the context of {EVENT_TYPE} event. The definition should specify its role within the event rather than being a generic concept. For example, if the argument is "Fire Date," define it as "the date when the fire occurred" rather than just "a date expression."

Argument Validity Rule: Define the conditions or give examples of trigger words that often trigger the existence of the argument character for deciding whether this argument role is present in {EVENT_TYPE} event. You need to provide strict rules, that is, only if there are argument corresponding to the argument role in the main descriptive text fragment about the {EVENT_TYPE} event, the argument role will exist. If some words in the text frequently trigger the existence of a certain argument role, the trigger word can be added to the corresponding rule of that argument role; For example, the word "against" often triggers the role of the Protest Reason of Protest events.

Argument Span Rule: Guide the model to extract complete arguments spans that are consistent with the actual annotations, helping the model determine the start and end spans of argument. (e.g., "197 deaths and 165 injuries" instead of "197 deaths, 165 injuries").

Update rule guidance: 2.Ensure that each rule is clear and concise. Please note that the rules you refine must have generalization, not just be customized for the current example. You should focus more on obtaining more detailed and accurate argument extraction rules from the predicted results of the current example, so that the final rules can better perform document level event argument parameter extraction tasks; You will be provided with three current rules for each argument role, which are rules summarized by accumulating previous examples. When further refining based on the current example, you need to pay more attention to the argument roles with prediction errors, analyze the differences between the predicted results and the true annotations, and further refine the corresponding rules for this erroneous argument role on the basis of existing rules.

Document context: The text content is omitted here.

Detailed information on error feedback: Below are the validation details showing the discrepancies between the predicted results and the correct annotations:

Current rules:

{"Role Semantics Rule": {"Age": , "Date": , "Deceased": , "Location/Hospital": , "Perpetrator": },
"Argument Validity Rule": {"Age": , "Date": , "Deceased": , "Location/Hospital": , "Perpetrator": } ,
"Argument Span Rule": {"Age": , "Date": , "Deceased": , "Location/Hospital": , "Perpetrator": }}

Rule output format: Output Format (Follow Exactly) The phrase **Role Semantics Rule**, **Argument Validity Rule**, **Argument Span Rule** is part of the output. It is not a placeholder or a title—it is a required part of the final output and must be included exactly as shown. And Each [Argument Role] must be written exactly as given in the provided argument roles. Your final output **must** strictly follow this structure: Please make sure to use this format -[Argument Role] instead of any other format.

Rule output format:

Role Semantics Rule
[Argument Role]: "Role Semantics Rule"
Argument Validity Rule
[Argument Role]: "Argument Validity Rule"
Argument Span Rule
[Argument Role]: "Argument Span Rule"

Table 8: The complete prompt for refined hierarchical rules incorporating error feedback.

Objective: Your task is Event Argument Extraction. In this task, you will be provided with a document that describes an event and the goal is to extract the event arguments that correspond to each argument role associated with the event. The terminologies for this task is as follows:
Key Terminologies:
Event argument: an entity mention, temporal expression or value that serves as a participant or attribute with a specific role in an event. Event arguments should be quoted exactly as the it appears in the given document.
Argument role: the relationship between an argument to the event in which it participates.
rules list: serving as guiding principles or strategies to aid the extraction of event arguments, tailored to specific argument roles.
Specifically, you will use a set of given rules for argument roles to complete the event argument extraction task for the provided document.

Example task:
Question: Extract the event arguments of giver, beneficiary, and recipient in the "transaction.transaction.giftgrantprovideaid" event in the provided document, with the trigger word being "granted", highlighted between "<t>" and "</t>". When pinpointing each event argument, it's crucial to quote the entity exactly as it appears in the text. If an event argument is not explicitly mentioned or cannot be directly associated with the event indicated by the trigger word, please respond with "not specified".
Document: a news document
Trigger sentence: "The access to the research center in the city was <t>granted</t> by the administrator. The man, Ripley Johnson, earned it."
Answer: Elaborate the meaning of event type:
"transaction.transaction.giftgrantprovideaid": The event involves a transfer of money or resources in the form of a gift, grant, or provision of aid, signaled by the action of granting.
Recognizing [giver] in the given document:
Step 1: Identify the argument of [giver] based on Optimal Hierarchical Rules. The given rules of [giver] in the Optimal Hierarchical Rules is: "Here is the content of the given rules of [giver] in the Optimal Hierarchical Rules".
Step 2: Applying Optimal Hierarchical Rules to the document, the argument of [giver] is "administrator".
Recognizing [recipient] in the given document:
Step 1: Identify the argument of [recipient] based on Optimal Hierarchical Rules. The given rules of [recipient] in the Optimal Hierarchical Rules are: "Here is the content of the given rules of [recipient] in the Optimal Hierarchical Rules".
Step 2: Applying Optimal Hierarchical Rules to the document, the argument of [recipient] is "Ripley Johnson".

Target task:
Question: Extract the event arguments of place, and participant in the "contact.discussion.meet" event in the provided document, with the trigger word being "debate", highlighted between "<t>" and "</t>" in the news document. When pinpointing each event argument, it's crucial to quote the entity exactly as it appears in the text. If an event argument is not explicitly mentioned or cannot be directly associated with the event indicated by the trigger word, please respond with "not specified".

Document: The text content is omitted here.
Trigger sentence: " "
Prioritize the identification of event arguments within the specified trigger sentence. If an event argument is not explicitly mentioned, please answer "not specified".

rules list:
{"Optimal Hierarchical Rules": {"place": , "participant": }}

Answer: (adapting the format of the answer in the example):

Table 9: The Demonstration for RAMS Dataset.

Objective:

Objective: Your task is Event Argument Extraction. In this task, you will be provided with a document that describes an event and the goal is to extract the event arguments that correspond to each argument role associated with the event. The terminologies for this task is as follows:

Key Terminologies:

Event argument: an entity mention, temporal expression or value that serves as a participant or attribute with a specific role in an event. Event arguments should be quoted exactly as the it appears in the given document.

Argument role: the relationship between an argument to the event in which it participates.

rules list: serving as guiding principles or strategies to aid the extraction of event arguments, tailored to specific argument roles.

Specifically, you will use a set of given rules for argument roles to complete the event argument extraction task for the provided document.

Example task:

Question: Extract the event arguments of "Date", "Casualties and Losses", "Magnitude", "Number of Destroyed Building" in the "Fire" event in the provided news document. When pinpointing each event argument, it's crucial to quote the entity exactly as it appears in the text. Note that if an event argument is not explicitly mentioned or cannot be directly associated with its argument role in question, please respond with "not specified".

Document: a news, the content is omitted here

Answer: Elaborate the meaning of event type:

"Fire": The event involves the rapid combustion of materials, resulting in the release of heat, light, and flame, often accompanied by smoke.

Recognizing [Date] in the given document:

Step 1: Identify the argument of [Date] based on Optimal Hierarchical Rules. The given rules of [Date] in the Optimal Hierarchical Rules is: "Here is the content of the given rules of [Date] in the Optimal Hierarchical Rules".

Step 2: Applying Optimal Hierarchical Rules to the document, the argument of [Date] is "not specified".

Recognizing [Casualties and Losses] in the given document:

Step 1: Identify the argument of [Casualties and Losses] based on Optimal Hierarchical Rules. The given rules of [Casualties and Losses] in the Optimal Hierarchical Rules are: "Here is the content of the given rules of [Casualties and Losses] in the Optimal Hierarchical Rules".

Step 2: Applying Optimal Hierarchical Rules to the document, the argument of [Casualties and Losses] is "claimed 142 deaths" and "800 houses were damaged".

Recognizing [Magnitude] in the given document:

Step 1: Identify the argument of [Magnitude] based on Optimal Hierarchical Rules. The given rules of [Magnitude] in the Optimal Hierarchical Rules are: "Here is the content of the given rules of [Magnitude] in the Optimal Hierarchical Rules".

Step 2: Applying Optimal Hierarchical Rules to the document, the argument of [Magnitude] is "6.6".

Target task:

Question: Extract the event arguments of "Suspect", "Criminal Evidence", "Police", "The Charged Crime", "Arrest Location" in the "CommitCrime - Arrest" event in the provided news document below. When pinpointing each event argument, it's crucial to quote the entity exactly as it appears in the text.

Document: The text content is omitted here.

rules list:

{"Optimal Hierarchical Rules": {"Suspect": , "Criminal Evidence": , "Police": , "The Charged Crime": , "Arrest Location": }}

Answer: (adapting the format of the answer in the example):

Table 10: The Demonstration for DocEE Dataset.