Towards Robust Universal Information Extraction: Dataset, Evaluation, and Solution

Jizhao Zhu[1,2], Akang Shi[1,2], Zixuan Li[1,3]*, Long Bai[1,3], Xiaolong Jin[1,3,4]*, Jiafeng Guo[1,3,4], Xueqi Cheng[1,3,4]

[1]Key Laboratory of Network Data Science and Technology,
Institute of Computing Technology, Chinese Academy of Sciences
[2]School of Computer Science, Shenyang Aerospace University, Shenyang, China
[3]State Key Laboratory of AI Safety
[4]School of Computer Science, University of Chinese Academy of Sciences
{zhujz@sau.edu.cn, shiakang@stu.sau.edu.cn,lizixuan@ict.ac.cn}

Abstract

In this paper, we aim to enhance the robustness of Universal Information Extraction (UIE) by introducing a new benchmark dataset, a comprehensive evaluation, and a feasible solution. Existing robust benchmark datasets have two key limitations: 1) They generate only a limited range of perturbations for a single Information Extraction (IE) task, which fails to evaluate the robustness of UIE models effectively; 2) They rely on small models or handcrafted rules to generate perturbations, often resulting in unnatural adversarial examples. Considering the powerful generation capabilities of Large Language Models (LLMs), we introduce a new benchmark dataset for Robust UIE, called RUIE-Bench, which utilizes LLMs to generate more diverse and realistic perturbations across different IE tasks. Based on this dataset, we comprehensively evaluate existing UIE models and reveal that both LLM-based models and other models suffer from significant performance drops. To improve robustness and reduce training costs, we propose a data-augmentation solution that dynamically selects hard samples for iterative training based on the model's inference loss. Experimental results show that training with only **15%** of the data leads to an average **8.1%** relative performance improvement across three IE tasks. Our code and dataset are available at: https://github.com/ICT-GoKnow/RobustUIE.

1 Introduction

Information Extraction (IE) aims to extract structured knowledge from unstructured text based on predefined types of entities, relations, and events. It plays a fundamental role in downstream applications such as knowledge graph construction (Ji et al., 2021), information retrieval (Zhu et al., 2023), and reasoning (Guan et al., 2020). Universal Information Extraction (UIE), which seeks to unify the extraction of various knowledge types through a single model, has achieved significant progress in recent years. Most existing studies have primarily focused on enhancing the overall performance of UIE models, typically evaluated on fixed test sets. However, they often overlook the robustness (and generalization ability) of UIE models, which are crucial when handling real-world text.

To measure the robustness of IE models, some studies focus on constructing benchmark datasets by generating adversarial examples with small perturbations. For example, RockNER (Lin et al., 2021) employs a rule-based approach and BERT (Devlin et al., 2019) to generate two kinds of perturbations for Named Entity Recognition (NER); Li et al. (2021) generate adversarial examples for Relation Extraction (RE) by random replacing words with synonyms or similar words generated by some Natural Language Processing (NLP) tools. Liu et al. (2020) replace verbs and context using similar words generated by GloVe (Pennington et al., 2014) for Event Detection (ED). Overall, existing benchmark datasets typically have two limitations: 1) They generate limited kinds of perturbations for individual IE tasks, making it difficult to comprehensively evaluate the robustness on UIE models across various IE tasks; 2) They generate adversarial examples typically using small models or handcrafted rules, often resulting in unnatural samples.

Considering the powerful NLP capabilities of Large Language Models (LLMs) (OpenAI., 2023; Qin et al., 2024; Li et al., 2024a), we leverage them in this paper to generate more diverse and realistic perturbations. After human verification of the annotation accuracy of LLMs, we obtain a new benchmark dataset for Robust UIE, called RUIE-Bench. RUIE-Bench contains **12,700** samples and includes **14** distinct kinds of perturbation across three mainstream IE tasks, i.e., NER, RE, and ED. Based on RUIE-Bench, we conduct a comprehen-

*Corresponding authors.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 28052–28070
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

sive evaluation of existing UIE models, including 8 open-source LLMs, 6 closed-source LLMs, 4 traditional IE models, and 4 fine-tuned UIE models. We obtain some intriguing observations from the experimental results, such as open-source LLMs have a significant performance gap compared with closed-source LLMs in both original tests and perturbation tests. LLM-based UIE models demonstrate better robustness than traditional IE models under certain perturbations. However, both types of models suffer from significant performance drops.

To improve the robustness of UIE models, a common solution is data augmentation. Since the training cost is also a key factor, especially for LLM-based UIE models, we further propose a **L**oss-guided **D**ata **A**ugmentation (LDA) solution to enhance the robustness of models using a limited number of samples. Specifically, we first generate additional adversarial examples for training. Then, the inference loss on these samples is leveraged to dynamically select the most challenging ones to fine-tune the UIE model. Using the fine-tuned model, we iteratively calculate the inference loss and select hard samples for the next round of training. The experimental results demonstrate that training KnowCoder (Li et al., 2024b) with just 15% of the augmented data using LDA yields a **8.1%** relative improvement in average performance on RUIE-Bench, compared with the state-of-the-art models. This performance is comparable to the fully trained model using the entire augmented dataset. Additionally, when evaluated on the unseen dataset, KnowCoder with LDA outperforms the fully trained model by an average of **8.9%** across three IE tasks.

In summary, our contributions are as follows:

- We construct RUIE-bench, which contains **12,700** samples with **14** distinct perturbations generated by LLMs across various IE tasks, which is the most comprehensive benchmark dataset with the most diverse perturbations for robust UIE.

- Based on the RUIE bench, we comprehensively evaluate existing IE models. The evaluation results highlight that current IE models exhibit robustness issues against perturbations.

- To improve the robustness with limited samples, we further propose a loss-guided data augmentation solution, which achieves performance comparable to training with the full dataset by using only 15% of the data. Moreover, when evaluated on unseen datasets, LDA outperforms the fully trained model with 8.9% F1 on average across three IE tasks.

2 Related Work

Universal Information Extraction can be classifier into two kinds of methods, classification-based (Lin et al., 2020a; Nguyen et al., 2022) and generation-based UIE methods (Lu et al., 2022; Wang et al., 2023; Li et al., 2024b). The former mainly adopts the end-to-end joint extraction mode, enhancing cross-task interactions with global dependency modeling for unified extraction. The latter aims to generate structural information rather than extracting structural information from plain text. Recently, some UIE methods employ various types of prompts to enable LLMs to understand schemas and extract the corresponding knowledge. For example, InstructUIE (Wang et al., 2023) applies a text-style prompt for schema understanding. In contrast, KnowCoder (Li et al., 2024b) uses a code-style prompt to transform UIE into code generation, achieving state-of-the-art performance.

Robustness Research for IE primarily focuses on constructing benchmark datasets for individual IE tasks. For NER, RockNER (Lin et al., 2021) replaces the original entities with entities of relevant types from Wikidata[1], and employs BERT (Devlin et al., 2019) to substitute context words; Jin et al. (2023) adopt disentanglement and word attribution methods to identify keywords and injects character-level perturbations for these words; Srinivasan and Vajjala (2023) apply multiple perturbations, including random entity replacement, Bert-based contextual modifications, and paraphrase generation. For RE, Li et al. (2021) generate adversarial examples by randomizing word substitutions using RoBERTa (Liu, 2019) or using synonyms; Nolano et al. (2024) generate adversarial examples by replacing entities in triples using various strategies, including same-type and different-type substitutions. For ED, Liu et al. (2020) use GloVe (Pennington et al., 2014) to replace similar words to generate adversarial examples.

To enhance the robustness of IE, the existing method (Lin et al., 2021) expands the original training data by generating additional examples.

[1] https://www.wikidata.org

Furthermore, Li et al. (2021); Jin et al. (2023); Srinivasan and Vajjala (2023) employ perturbation techniques to generate adversarial examples for training. While these methods typically require a large number of adversarial examples, none focus on improving model robustness using only a small number of augmented samples.

3 The Construction of RUIE-Bench

In this section, we first introduce the methods of utilizing LLMs to generate adversarial examples. Subsequently, we provide the concrete construction process of the RUIE-Bench dataset.

3.1 LLM-based Adversarial Example Generation

In UIE, adversarial examples refer to samples intentionally perturbed to mislead UIE models while maintaining their original semantics or appearance. Mathematically, given an input sample s with its corresponding label y, an adversarial example s' is generated by applying a small perturbation, which is constrained to ensure that it remains close to the input space. The goal of adversarial examples is to cause a model to predict an incorrect label.

To generate more diverse and realistic adversarial examples, we employed LLMs to simulate different kinds of perturbations for NER, RE, and ED tasks by designing various prompts for different IE tasks. Additionally, we utilized two general rule-based perturbations, which were applied across all IE tasks. A comprehensive demonstration of these adversarial examples is presented in Figure 1. In what follows, we will introduce these different kinds of perturbations for IE tasks in detail.

Replace Entity, Triple, and Trigger. A robust UIE model should be able to identify entities, relations, and events based on the context of the sentence corresponding to the task rather than memorizing each entity, relational triple, or event trigger and their corresponding types. To prevent the model from memorizing specific patterns instead of reasoning based on context, we introduce perturbations to existing sentences.

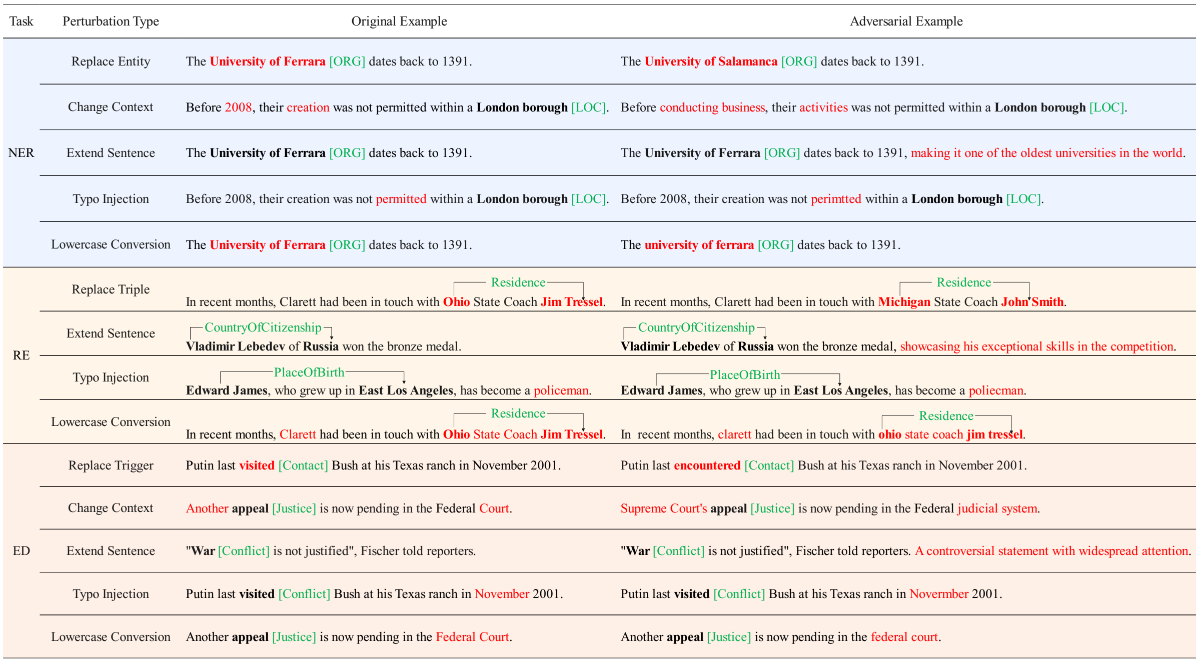
Such perturbations need to ensure the consistency of the original type during replacement. Although previous studies (Lin et al., 2021; Nolano et al., 2024) have introduced similar perturbations in NER and RE samples, such as replacing entities based on rules, these methods may lead to incorrect replacement. Moreover, for ED samples, replacing triggers while ensuring the type remains unchanged is an extremely challenging task. Given this, we instruct GPT-4 (OpenAI., 2023) to replace entities, relational triples, or event triggers while preserving their types and ensuring that other content remains unchanged. The corresponding prompts are provided in Appendix A.

Change Context. A robust UIE model should keep its performance even when the contextual content of the sample changes due to various perturbations. To evaluate the robustness of the UIE model against contextual variations, we introduce perturbations to context words. It should be noted that this method is only used for samples of NER and ED tasks since altering context words in RE may disrupt the semantic relations between entity pairs.

In the previous methods, the mask language model BERT (Devlin et al., 2019) is used for changing context, but these methods often generate semantically inaccurate or syntactically incorrect words, and only one word can be generated for a single mask position. Therefore, we use LLMs to change the context in sentences. Specifically, we first get rid of punctuation, entities, event triggers, and stop words in sentences, leaving only meaningful context words. And randomly choose up to four words to replace with the [MASK] token. Then, GPT-4 (OpenAI., 2023) is instructed to generate three predictions for each [MASK] token and randomly select one for replacement. The prompts can be found in Appendix A.

Extend Sentence. Generally, a robust UIE model is capable of accurately extracting the required information, even in the face of complex sentence structures or long text situations. In order to better evaluate the robustness of the UIE model in handling complex sentences or long text situations, we enhance the semantic depth of the sentences by adding semantically relevant content (such as contextual details, historical facts, or explanatory clauses), thereby increasing the complexity of the sentences.

Prior research has not explored the robustness of IE models under similar perturbations. Considering that added content must maintain semantic coherence and meaningfulness, we employ LLMs to implement this perturbation. For specific tasks, we instruct GPT-4 (OpenAI., 2023) to adhere to corresponding constraints. For NER, new sentences must preserve the original entity boundaries



Task	Perturbation Type	Original Example	Adversarial Example
NER	Replace Entity	The **University of Ferrara** [ORG] dates back to 1391.	The **University of Salamanca** [ORG] dates back to 1391.
	Change Context	Before 2008, their creation was not permitted within a **London borough** [LOC].	Before conducting business, their activities was not permitted within a **London borough** [LOC].
	Extend Sentence	The **University of Ferrara** [ORG] dates back to 1391.	The **University of Ferrara** [ORG] dates back to 1391, making it one of the oldest universities in the world.
	Typo Injection	Before 2008, their creation was not permitted within a **London borough** [LOC].	Before 2008, their creation was not perimtted within a **London borough** [LOC].
	Lowercase Conversion	The **University of Ferrara** [ORG] dates back to 1391.	The **university of ferrara** [ORG] dates back to 1391.
RE	Replace Triple	Residence In recent months, Clarett had been in touch with **Ohio** State Coach **Jim Tressel**.	Residence In recent months, Clarett had been in touch with **Michigan** State Coach **John Smith**.
	Extend Sentence	CountryOfCitizenship **Vladimir Lebedev** of **Russia** won the bronze medal.	CountryOfCitizenship **Vladimir Lebedev** of **Russia** won the bronze medal, showcasing his exceptional skills in the competition.
	Typo Injection	PlaceOfBirth **Edward James**, who grew up in **East Los Angeles**, has become a policeman.	PlaceOfBirth **Edward James**, who grew up in **East Los Angeles**, has become a poliecman.
	Lowercase Conversion	Residence In recent months, Clarett had been in touch with **Ohio** State Coach **Jim Tressel**.	Residence In recent months, clarett had been in touch with **ohio** state coach **jim tressel**.
ED	Replace Trigger	Putin last **visited** [Contact] Bush at his Texas ranch in November 2001.	Putin last **encountered** [Contact] Bush at his Texas ranch in November 2001.
	Change Context	Another **appeal** [Justice] is now pending in the Federal Court.	Supreme Court's **appeal** [Justice] is now pending in the Federal judicial system.
	Extend Sentence	"**War** [Conflict] is not justified", Fischer told reporters.	"**War** [Conflict] is not justified", Fischer told reporters. A controversial statement with widespread attention.
	Typo Injection	Putin last **visited** [Conflict] Bush at his Texas ranch in November 2001.	Putin last **visited** [Conflict] Bush at his Texas ranch in Novermber 2001.
	Lowercase Conversion	Another **appeal** [Justice] is now pending in the Federal Court.	Another **appeal** [Justice] is now pending in the federal court.

Figure 1: Illustration of generated adversarial examples with different kinds of perturbations.

and types without introducing new entities. For RE, the original relational triples must remain unchanged, with no additional relational information introduced. For ED, the original event triggers in sentences must be retained without incorporating new event information. The prompts are provided in Appendix A.

Typo Injection. In reality, typos are common. However, a robust UIE model should continue to make accurate predictions when dealing with these typos. To simulate these common text spelling mistakes, we introduce typo injection. Namely, spelling mistakes are added to the words prone to errors. Initially, we tried using LLMs to inject, but it often introduced unrealistic errors. Therefore, we switched to a rule-based approach to achieve this.

We focus on sentences with more than eight words and select longer words (over six characters), as longer sentences and words are more prone to spelling errors. Additionally, we filter out stop words and high-frequency vocabulary to avoid selecting common words. We randomly choose 1-3 words from the remaining words for typo injection and avoid changing the first character for each selected word and introduce errors by randomly replacing characters, deleting characters, inserting characters, or swapping adjacent characters.

Lowercase Conversion. Consider that in practice, users overwhelmingly spell in lowercase, especially in informal environments such as social media, email, or search queries. Therefore, lowercase conversion is used to simulate non-standard input, which helps evaluate the robustness of the UIE model in response to changes in text format.

In this method, all characters of each word are converted to lowercase, except for the first letter of the first word. This tests whether the model can still accurately extract information under non-standard input conditions, forcing it to rely on semantic understanding rather than surface features. By doing so, it not only assesses the model's robustness but also highlights how upper and lowercase expressions affect task performance.

3.2 Dataset Construction

To construct the RUIE-Bench dataset, we select seven datasets across the three subtasks of UIE. For NER, we use the ACE05-Ent (Walker and Consortium, 2005), CoNLL03 (Sang and Meulder, 2003), and WikiANN (Pan et al., 2017) datasets; for RE, we select the ACE05-Rel (Walker and Consortium, 2005) and NYT (Riedel et al., 2010) datasets; for ED, we use the ACE05-Evt (Walker and Consortium, 2005) and CASIE (Satyapanich et al., 2020) datasets. To balance sample sizes across these subtasks, we perform stratified sampling from the test sets of respective datasets, adhering to the principle of maintaining distributional consistency. Specifically, we conduct stratified sampling on all label types (including NULL-type) within each test set, obtaining 1,000 NER samples, 800 RE samples,

Benchmark Dataset	RUIE-Bench (Ours)	RockNER (Lin et al., 2021)	DWR (Jin et al., 2023)	Adv_re (Nolano et al., 2024)	CSMG (Liu et al., 2020)
Supported tasks	NER & RE & ED	NER	NER	RE	ED
Covered datasets	7	1	3	1	2
Number of perturbation types	14	3	1	4	2
Number of adversarial examples	12,700	13,169	-	6,277	-
Methods for generating perturbations	LLM & Rule	Small model & Rule	Small model	Small model & Rule	Small model

Table 1: Comparison between RUIE-Bench and existing IE robust evaluation benchmark datasets. The - indicates that the detailed statistics of the datasets are not reported in their papers.

and 900 ED samples. For adversarial example generation, we apply the perturbation methods detailed in Section 3.1. For each original sample, corresponding adversarial samples are generated for every perturbation type, while retaining those samples that cannot be perturbed. During generation, we implement strict quality control through manual verification: any sample containing errors is immediately discarded and regenerated until accurate samples are obtained. Through this rigorous process, we successfully construct the RUIE-Bench dataset. We present a comprehensive comparison between RUIE-Bench and existing IE robustness evaluation benchmark datasets in Table 1. For detailed statistics and further information regarding RUIE-Bench, please refer to Appendix B.

4 Loss-guided Data Augmentation

Data augmentation is a common strategy to improve model robustness (Rebuffi et al., 2021; Wang et al., 2022; Li and Spratling, 2023). Existing methods primarily focus on synthesizing in-distribution adversarial samples for training. However, none of them focus on improving efficiency by selecting a minimal number of training samples to enhance robustness. This is especially critical for LLMs, where training costs are significantly higher due to their scale and complexity.

Inspired by Song et al. (2023); Buchnik and Cohen (2020); Werner (2023), we propose a loss-guided data augmentation solution for robust UIE. The core idea is to focus on samples where the model's performance is currently suboptimal, as indicated by higher loss values, thereby potentially accelerating convergence and improving overall model performance. First, we train the initial model on the original training set and use it to compute the inference loss of the augmented samples. For each IE task, samples with high inference loss were selected to fine-tune the model. Next, based on the obtained fine-tuned model, the inference loss of the augmented samples is recalculated, followed by another round of sample selection and fine-tuning. This iterative process was repeated t times until a robust model was obtained.

Specifically, we first fine-tune the initial model M using the original training data to obtain M_0, and employ LLMs to generate augmented data D_{aug} for the original training set. During each iteration, we use M_{t-1} to compute inference loss L_i for each augmented sample. Then, based on selection ratio β, the samples with higher loss are selected to form a new training dataset $D^{(t)}_{\text{retrain}}$, which is subsequently used to fine-tune M_{t-1} to obtain model M_t. After each iteration, the model is evaluated on the validation set. The algorithm terminates and returns the final model M_t when the improvement on the validation set falls below the convergence threshold δ. Algorithm 1 presents the training process of the proposed strategy. The details of augmented data generation are presented in Appendix C.

Algorithm 1 Loss-guided Data Augmentation

```
Input: Training data $D$, Initial model $M$, Selection ratio $\beta$, Convergence threshold $\delta$
Output: Fine-tuned model $M_t$
Use LLMs to generate augmented data $D_{\text{aug}}$ based on $D$;
Fine-tune $M$ on data $D$ to obtain model $M_0$;
$t \leftarrow 1$;
do
    for each sample $(x_i, y_i) \in D_{\text{aug}}$ do
        Compute loss: $L_i = L(\theta_{t-1}; x_i, y_i)$
    end for
    Sort samples in $D_{\text{aug}}$ by loss $L_i$ in descending order;
    Select top $\beta$ samples to form $D^{(t)}_{\text{retrain}}$;
    Fine-tune model $M_{t-1}$ on $D^{(t)}_{\text{retrain}}$ to obtain new model $M_t$;
    $\beta \leftarrow \beta/2$;    $t \leftarrow t+1$;
while performance improvement of $M_t$ on the validation set falls below threshold $\delta$;
return $M_t$
```

5 Experiment Setup

We use RUIE-Bench to evaluate the robustness of the current models, including UIE ones, traditional IE ones, and LLMs. Meanwhile, we construct an unseen dataset to measure their generalization ability.

5.1 Metrics

We report the span-based offset Micro F1, following previous methods (Lu et al., 2022; Lin et al., 2020b). For NER, an entity is considered correct if both its boundaries and type are accurately predicted. For RE, a relation is deemed correct if the triple, including the relation type, head entity, and tail entity, matches the gold annotations. For ED, an event trigger is considered correct if both the event type and trigger are aligned with the gold annotations.

5.2 Training Details

To explore effective data augmentation methods, the KnowCoder-7B-base[2] is selected as the initial model M. Then, we fine-tune it on the seven datasets constructed for RUIE-Bench, obtaining the model M_0, which is denoted as KnowCoder-7B$_{\text{partial}}$. Additionally, the perturbation methods proposed in Section 3.1 are employed to generate augmented data D_{aug} based on the original training set. Next, we fine-tune the initial model using both original training data and all augmented data, obtaining the model KnowCoder-7B-Robust. Finally, we fine-tune the initial model using high-loss augmented samples selected according to the LDA strategy, with an initial selection ratio of 10%. After two iterations, we successfully construct the model KnowCoder-7B-Robust$_{\text{LDA}}$, utilizing a total of 15% augmented samples.

During the fine-tuning phase, we employ LoRA (Hu et al., 2021) for efficient parameter tuning. The LoRA rank and LoRA alpha parameters are set to 32 and 64, respectively. The learning rate is set to 3×10^{-4}, with a warm-up rate of 0.1 and a dropout rate of 0.1. The sequence length is limited to 2048 and the batch size is set to 256. Additionally, for LDA training, the selection ratio β is set to 10%, and the convergence threshold δ is set to 0.3, meaning that iteration stops when the Micro F1 score improvement of the new model is less than 0.3. During validation phase, we use greedy search with a temperature of 0 and set the maximum output length to 640. All experiments are conducted on 8 × NVIDIA-A100 80G.

5.3 Baselines

We adopt the state-of-the-art UIE models to validate the robustness, including UIE (Lu et al., 2022), IstructUIE (Wang et al., 2023), YAYI-UIE (Xiao et al., 2024), and KnowCoder (Li et al., 2024b). We also employ traditional IE models for robustness evaluation across the NER, RE, and ED tasks. For NER, we choose Stanza (Qi et al., 2020) and TNER (Ushio and Camacho-Collados, 2021); for RE, we select PFN (Yan et al., 2021); and for ED, we choose EEQA (Du and Cardie, 2020). Additionally, we evaluate the robustness of two categories of LLMs: open-source models, including Qwen2.5-3B-Instruct, Qwen2.5-7B-Instruct, Qwen2.5-14B-Instruct(Yang et al., 2024), Llama3-8B-Instruct (Dubey et al., 2024), Glm-4-9B-Chat (GLM et al., 2024), CodeLlama-7B-Instruct (Rozière et al., 2024), Internlm2.5-7B-Chat (Team, 2023), and Vicuna-7B-v1.5 (Zheng et al., 2023); Commercial model API services, including GPT-3.5-turbo, GPT-4-turbo (OpenAI., 2023), DeepSeek-V3 (Liu et al., 2024), DeepSeek-R1 (Guo et al., 2025), GLM4-Plus (GLM et al., 2024) and Qwen2.5-Max (Yang et al., 2024). For the evaluation of LLMs, we employ the 10-shot approach to instruct LLMs to conduct IE tasks with the specific prompts provided in Appendix D.

6 Results and Analyses

6.1 Results on the RUIE-Bench dataset

We report the Micro F1 scores of all the models across the three IE tasks on RUIE-Bench in Table 2. These results cover both the original test set and various perturbation settings. For the sake of space limitation in the table, we use abbreviations for these perturbations. For NER, we use P1-P5 to represent the perturbations of Replace Entity, Change Context, Extend Sentence, Typo Injection, and Lowercase Conversion, respectively. For RE, the perturbations of Replace Triple, Extend Sentence, Typo Injection, and Lowercase Conversion are denoted as P6-P9, respectively. For ED, we employ P10-P14 to represent Replace Trigger, Change Context, Extend Sentence, Typo Injection, and Lowercase Conversion, respectively. We also report the overall performance drop of all the models under the three tasks for all perturbations in

[2] https://huggingface.co/golaxy/KnowCoder-7B-base

Model	**NER**							**RE**						**ED**							**Overall**	
	None	P1	P2	P3	P4	P5	Drop$_{avg}$	None	P6	P7	P8	P9	Drop$_{avg}$	None	P10	P11	P12	P13	P14	Drop$_{avg}$	Avg	Rank
Open-source LLMs																						
Qwen2.5-14B-Instruct	58.6	53.6	57.7	55.0	56.9	46.6	7.9%$_\downarrow$	22.6	19.2	21.3	17.1	8.8	26.5%$_\downarrow$	29.8	27.5	28.3	28.5	28.5	28.9	4.9%$_\downarrow$	34.6	13
Qwen2.5-7B-Instruct	53.3	49.8	51.2	50.5	51.2	41.3	8.4%$_\downarrow$	15.6	13.4	14.0	13.8	3.8	27.9%$_\downarrow$	18.5	18.0	17.4	17.4	17.6	18.1	4.3%$_\downarrow$	27.3	15
Qwen2.5-3B-Instruct	49.5	47.5	46.7	45.3	45.5	40.2	9.0%$_\downarrow$	8.9	7.6	8.6	7.4	2.0	28.1%$_\downarrow$	11.9	11.4	10.7	11.2	11.0	11.6	6.0%$_\downarrow$	22.2	18
Llama3-8B-Instruct	55.4	52.6	52.9	51.1	53.5	25.7	14.9%$_\downarrow$	17.3	15.0	15.7	13.6	2.5	32.4%$_\downarrow$	11.9	11.8	11.9	10.5	11.3	11.0	5.0%$_\downarrow$	24.9	16
Glm-4-9B-Chat	57.4	54.0	55.8	51.4	56.6	43.2	9.0%$_\downarrow$	8.8	7.5	7.5	7.4	1.8	31.2%$_\downarrow$	7.6	8.0	6.8	5.5	6.1	7.6	10.5%$_\downarrow$	23.1	17
Internlm2.5-7B-Chat	51.6	48.0	48.8	46.9	45.3	31.0	14.7%$_\downarrow$	12.0	11.3	10.1	9.0	1.7	33.1%$_\downarrow$	10.6	10.4	10.0	7.8	9.0	10.4	10.2%$_\downarrow$	22.0	19
CodeLlama-7B-Instruct	46.3	45.0	45.0	38.9	42.4	14.5	19.7%$_\downarrow$	13.7	11.6	12.2	11.3	2.8	30.8%$_\downarrow$	9.5	9.6	9.4	7.1	8.6	9.8	6.3%$_\downarrow$	19.8	20
Vicuna-7B-v1.5	39.0	38.2	37.4	35.0	38.0	16.7	15.2%$_\downarrow$	11.2	11.0	10.1	7.6	0.8	34.1%$_\downarrow$	6.9	7.0	6.8	5.4	5.8	6.2	9.5%$_\downarrow$	16.7	21
Commercial LLMs API Services																						
DeepSeek-R1	67.1	63.2	66.8	65.9	65.0	57.2	5.2%$_\downarrow$	37.4	33.1	35.4	32.2	20.6	18.9%$_\downarrow$	45.8	41.8	45.5	44.1	44.1	45.7	3.4%$_\downarrow$	47.7	8
DeepSeek-V3	62.3	59.8	61.5	61.3	58.7	55.0	4.9%$_\downarrow$	31.3	29.0	29.6	26.2	10.0	24.3%$_\downarrow$	35.3	34.6	34.5	31.4	32.4	35.0	4.9%$_\downarrow$	40.5	11
Qwen2.5-Max	64.0	56.6	63.8	62.4	61.0	58.1	5.6%$_\downarrow$	34.9	30.9	33.9	28.3	13.2	23.8%$_\downarrow$	38.8	36.3	38.0	38.1	37.1	38.3	3.2%$_\downarrow$	43.2	9
GLM4-Plus	63.2	59.8	63.0	61.6	60.9	49.7	6.6%$_\downarrow$	32.2	29.2	31.3	26.1	5.3	28.6%$_\downarrow$	37.5	34.9	37.4	30.3	35.1	37.4	6.6%$_\downarrow$	40.9	10
GPT-4-turbo	60.6	57.5	59.8	58.2	56.2	33.4	12.5%$_\downarrow$	33.0	30.0	31.6	26.8	4.5	29.6%$_\downarrow$	35.8	33.9	35.4	31.5	33.3	35.7	5.1%$_\downarrow$	38.7	12
GPT-3.5-turbo	51.8	47.9	48.9	50.5	39.0	33.1	15.3%$_\downarrow$	23.8	20.6	21.3	16.7	2.4	35.9%$_\downarrow$	31.4	24.6	29.8	29.6	27.4	30.4	9.7%$_\downarrow$	31.1	14
Traditional IE Models																						
Stanza	80.7	70.1	77.1	71.5	78.1	51.1	13.8%$_\downarrow$	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
TNER	83.0	73.3	78.0	73.9	81.0	73.2	8.6%$_\downarrow$	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PFN	-	-	-	-	-	-	-	76.3	58.6	73.8	68.4	20.4	27.5%$_\downarrow$	-	-	-	-	-	-	-	-	-
EEQA	-	-	-	-	-	-	-	-	-	-	-	-	-	67.5	52.7	63.7	63.2	58.1	66.1	10.0%$_\downarrow$	-	-
UIE Models																						
UIE	83.9	74.3	81.1	75.6	81.7	70.3	8.7%$_\downarrow$	**84.1**	63.3	81.1	77.0	35.9	23.5%$_\downarrow$	70.5	52.5	65.0	66.7	65.2	68.4	9.8%$_\downarrow$	70.4	5
InstructUIE-11B	73.9	65.7	69.5	64.3	72.0	70.3	7.5%$_\downarrow$	68.4	48.3	66.3	61.6	56.1	15.1%$_\downarrow$	59.3	49.2	55.9	58.2	57.5	58.4	5.8%$_\downarrow$	62.1	6
YAYI-UIE-13B	80.7	69.3	75.3	72.6	79.2	75.6	7.8%$_\downarrow$	66.4	47.3	65.0	59.9	38.4	20.7%$_\downarrow$	43.8	36.8	41.9	41.3	41.3	42.6	6.9%$_\downarrow$	57.5	7
KnowCoder-7B	**87.4**	76.4	81.3	79.6	84.7	81.5	7.7%$_\downarrow$	84.0	57.3	80.5	76.4	73.3	14.4%$_\downarrow$	**72.9**	53.5	**68.5**	69.2	68.3	**70.7**	9.4%$_\downarrow$	74.4	3
KnowCoder-7B$_{\text{partial}}$	84.4	73.8	80.1	81.1	82.1	79.0	6.1%$_\downarrow$	81.4	60.6	79.1	74.5	52.8	18.0%$_\downarrow$	68.2	54.6	65.1	64.5	64.7	66.1	7.6%$_\downarrow$	71.3	4
KnowCoder-7B-Robust	85.9	**81.3**	83.5	86.4	**86.1**	**84.6**	**1.7%**$_\downarrow$	83.1	66.0	**82.9**	81.1	79.8	6.8%$_\downarrow$	69.8	65.5	67.2	**69.9**	68.0	69.5	2.6%$_\downarrow$	**77.1**	1
KnowCoder-7B-Robust$_{\text{LDA}}$	86.1	81.2	**84.9**	**86.5**	85.6	83.8	1.9%$_\downarrow$	82.2	**66.5**	82.5	**81.3**	**81.3**	**5.2%**$_\downarrow$	69.1	**65.7**	67.9	69.5	**68.5**	68.8	**1.5%**$_\downarrow$	**77.1**	1

Table 2: The performance of all baselines and our models on RUIE-Bench.

the "Drop$_{\text{avg}}$" column. The robustness evaluation results of all the models are ranked, and the final ranking is shown in the "Rank" column. Although there are differences in the evaluation methods employed by different categories of models, we can still draw some interesting observations from the results:

(1) The models with data augmentation training show the best performance. The model KnowCoder-7B-Robust$_{\text{LDA}}$ trained with only 15% of the augmented data using LDA achieves results comparable with KnowCoder-7B-Robust. It convincingly verifies the effectiveness of the proposed LDA training strategy. Furthermore, a comprehensive comparison between these two models is in Appendix E.

(2) LLM-based models experience relatively smaller performance drops than other models, suggesting that LLMs have stronger generalization ability. This indicates that using LLMs to improve the robustness of UIE models is a promising approach for future work.

(3) All the LLMs exhibit a significant performance drop under various perturbations, especially in the NER and RE tasks. This indicates that LLMs face serious robustness issues when dealing with UIE tasks in few-shot prompting scenarios. However, LLM-based reasoning models such as DeepSeekR1 and Qwen2.5-Max demonstrate relatively better robustness compared to other LLMs, suggesting that incorporating stronger reasoning capabilities may enhance the stability of few-shot UIE performance under perturbations.

(4) From the results of the Qwen models with different parameter scales, it is evident that there is a significant positive correlation between the model's parameter scale and its robustness in NER and RE tasks. In other words, an increase in the number of model parameters often accompanies an improvement in robustness. For ED, although there is no clear positive correlation between parameter scale and robustness, the smallest model still demonstrates the weakest robustness.

6.2 Results on the Unseen Dataset

To verify whether the model trained on data with different perturbations can generalize to unseen datasets, we create an unseen dataset that does not include any samples from RUIE-Bench. Furthermore, we ensure that the types in this dataset are a subset of those in RUIE-Bench. For NER, we select OntoNotes 5.0 (Hovy et al., 2006) and random sample some instances as unseen data. Similarly, we obtain the unseen data for RE from CoNLL04 (Roth and tau Yih, 2004) and GIDS (Jat et al., 2018). For the ED task, since no datasets with the same event types exist, we use GPT-4 (OpenAI., 2023) to generate 100 unseen samples, which are then manually verified for correctness.

Table 3 shows the results of different models on the unseen dataset. From the table, we can find that: 1) The two LLM-based reasoning models, DeepSeekR1 and Qwen2.5-Max, show rela-

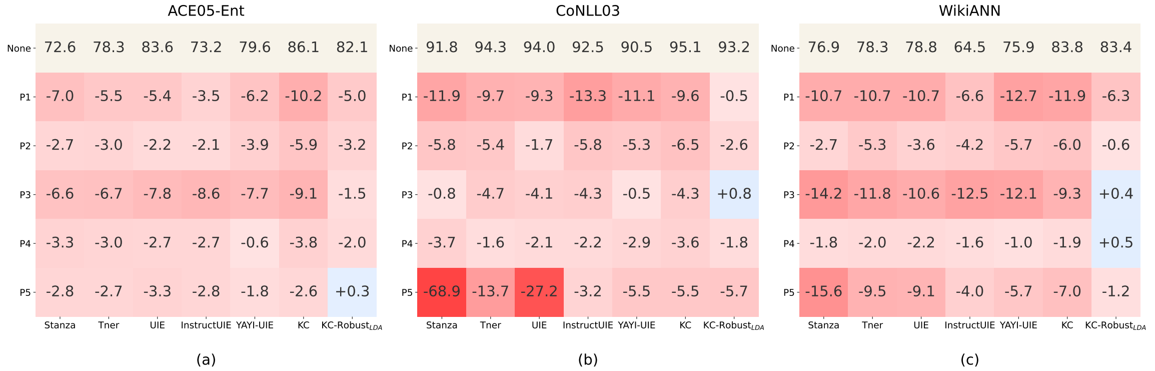


Figure 2: Performance comparison of different models under various perturbations on NER datasets. Red and blue indicate performance drop and improvement, respectively. KC is short for the KnowCoder model.

Model	**Unseen Dataset**			**Average**
	NER	**RE**	**ED**	
GLM4-Plus	56.6	41.4	56.1	51.4
DeepSeek-V3	59.2	39.8	55.5	51.5
GPT-4-turbo	58.9	38.8	59.7	52.5
Qwen2.5-Max	62.1	43.4	59.8	55.1
DeepSeek-R1	63.0	43.6	60.0	55.5
KnowCoder-7B$_{\text{partial}}$	64.9	40.4	52.2	52.5
KnowCoder-7B-Robust	62.8	47.4	56.6	55.6
KnowCoder-7B-Robust$_{\text{LDA}}$	**67.2**	**54.6**	**60.1**	**60.6**

Table 3: Results on the unseen dataset.

tively better generalization ability compared to the other LLMs. 2) Compared with the models without data augmentation training, the average performance of the models of KnowCoder-7B-Robust and KnowCoder-7B-Robust$_{\text{LDA}}$ in the UIE tasks is significantly improved. It is justified that training with the adversarial examples can enhance the model's generalization ability. 3) It is worth noting that the KnowCoder-7B-Robust$_{\text{LDA}}$ trained with only 15% of the augmented data using the LDA strategy achieves an average 8.9% performance improvement compared with the KnowCoder-7B-Robust using the complete set of augmented data fine-tuning. We guess that full training will lead to overfitting and thus show poor prediction ability on the unseen dataset.

6.3 Detailed Analysis

To further verify the effect of different perturbations on the traditional models and UIE models, we report the performance drop and improvement under various perturbations on the seven datasets in RUIE-Bench. For the NER, RE, and ED datasets, the detailed results are shown in Figure 2, Figure 3, and Figure 4, respectively. Through analysis, we can summarize the following observations.

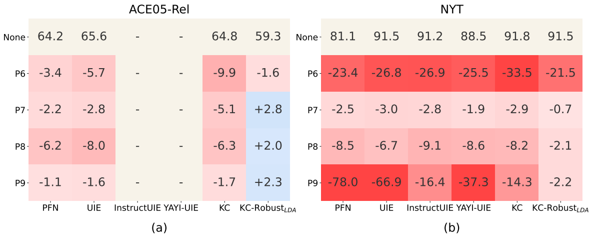


Figure 3: Performance comparison of different models under various perturbations on RE datasets.

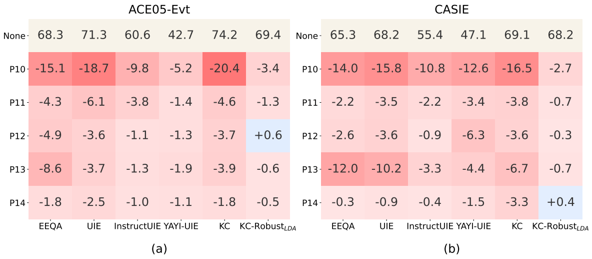


Figure 4: Performance comparison of different models under various perturbations on ED datasets.

(1) Under the three different perturbations of Replace Entity (P1), Replace Triple (P6), and Replace Trigger (P10) to the original test set results in a significant drop in model performance. This suggests that during training, the models may have memorized specific patterns of entities, relations, or events, rather than learning to reason based on contextual information. We provide representative case studies for all perturbations in Appendix F, including Replace Entity, Triple, and Trigger.

(2) Under certain perturbation settings, such as Lowercase Conversion, we observe that the model's performance drops significantly on some datasets while other datasets remain unaffected. This is because the annotations in the affected datasets contain a relatively high number of uppercase characters. Additionally, we find that the models using LLMs show better performance on these datasets. This suggests that LLMs already have strong generalization abilities to handle such

simple noise.

(3) KnowCoder-7B-Robust$_{\text{LDA}}$ shows a remarkable improvement across most of the datasets under nearly all perturbations. This observation strongly supports the effectiveness and feasibility of the LDA strategy. Furthermore, an interesting finding is observed in ACE05-Ent, WikiANN, and ACE05-Evt datasets: some models without data augmentation training (such as InstructUIE and YAYI-UIE) show similar robustness to KnowCoder-7B-Robust$_{\text{LDA}}$. This is because the performance of these models on the original test set is relatively poor, and thus, the perturbations have little effect on their performance.

7 Conclusion

In this paper, we introduced RUIE-Bench, a benchmark dataset designed to evaluate the robustness of UIE models. The dataset includes 14 adversarial perturbations for three core IE tasks, i.e., NER, RE, and ED. Through comprehensively benchmarking of existing models, the results reveal that these models struggle to maintain robustness when faced with these adversarial perturbations, highlighting the urgent need for robustness improvement for UIE. Motivated by this, we proposed a Loss-guided Data Augmentation (LDA) method that iteratively selects challenging samples for training. The results demonstrate that LDA achieves performance comparable to fully trained models on RUIE-Bench and even exhibits superior generalization capabilities on unseen datasets. This work aims to provide a valuable benchmark for evaluating robustness in UIE tasks and offer a practical methodology for enhancing model robustness.

Limitations

Generating more realistic perturbations remains an exploratory direction for future work. Although we propose various perturbation generation methods for UIE, they still fail to cover the diverse noise present in real-world scenarios. Meanwhile, due to cost and resource constraints, we have not conducted robustness evaluations on more LLMs. Moreover, the performance improvement achieved by the loss-guided data augmentation method may be constrained by the quality of the augmented data. In addition to these technical limitations, more robust UIE systems may also introduce societal risks, such as misuse for misinformation, surveillance, or amplification of bias. Addressing both the technical and ethical challenges will be a priority for future work, including more realistic perturbation design, broader model evaluation, and responsible deployment practices.

Acknowledgements

This work is funded by the National Natural Science Foundation of China under grants 62306299, 62441229 and 62406308, the CAS Special Research Assistant Fund, Lenovo-CAS Joint Lab Youth Scientist Project, the project under Grants No. JCKY2022130C039, and the Strategic Priority Research Program of the CAS under Grants No. XDB0680102. We thank anonymous reviewers for their insightful comments and suggestions.

References

Eliav Buchnik and Edith Cohen. 2020. Graph learning with loss-guided training. In *Proceedings of the 3rd Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA)*, pages 1–13.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*.

Saiping Guan, Xiaolong Jin, Jiafeng Guo, Yuanzhuo Wang, and Xueqi Cheng. 2020. NeuInfer: Knowledge inference on N-ary facts. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6141–6151, Online. Association for Computational Linguistics.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Eduard H. Hovy, Mitchell P. Marcus, Martha Palmer, Lance A. Ramshaw, and Ralph M. Weischedel. 2006. Ontonotes: The 90% solution. In *North American Chapter of the Association for Computational Linguistics*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Sharmistha Jat, Siddhesh Khandelwal, and Partha Talukdar. 2018. Improving distantly supervised relation extraction using word and entity based attention. *Preprint*, arXiv:1804.06987.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514.

Xiaomeng Jin, Bhanukiran Vinzamuri, Sriram Venkatapathy, Heng Ji, and Pradeep Natarajan. 2023. Adversarial robustness for large language NER models using disentanglement and word attributions. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12437–12450, Singapore. Association for Computational Linguistics.

Huaxia Li, Haoyun Gao, Chengzhang Wu, and Miklos A Vasarhelyi. 2024a. Extracting financial data from unstructured sources: Leveraging large language models. *Journal of Information Systems*, pages 1–22.

Lin Li and Michael Spratling. 2023. Data augmentation alone can improve adversarial training. *arXiv preprint arXiv:2301.09879*.

Luoqiu Li, Xiang Chen, Hongbin Ye, Zhen Bi, Shumin Deng, Ningyu Zhang, and Huajun Chen. 2021. On robustness and bias analysis of bert-based relation extraction. In *Knowledge Graph and Semantic Computing: Knowledge Graph Empowers New Infrastructure Construction: 6th China Conference, CCKS 2021, Guangzhou, China, November 4-7, 2021, Proceedings 6*, pages 43–59. Springer.

Zixuan Li, Yutao Zeng, Yuxin Zuo, Weicheng Ren, Wenxuan Liu, Miao Su, Yucan Guo, Yantao Liu, Lixiang Lixiang, Zhilei Hu, Long Bai, Wei Li, Yidan Liu, Pan Yang, Xiaolong Jin, Jiafeng Guo, and Xueqi Cheng. 2024b. KnowCoder: Coding structured knowledge into LLMs for universal information extraction. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8758–8779, Bangkok, Thailand. Association for Computational Linguistics.

Bill Yuchen Lin, Wenyang Gao, Jun Yan, Ryan Moreno, and Xiang Ren. 2021. RockNER: A simple method to create adversarial examples for evaluating the robustness of named entity recognition models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3728–3737, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020a. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020b. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Jian Liu, Yubo Chen, Kang Liu, Yantao Jia, and Zhicheng Sheng. 2020. How does context matter? on the robustness of event detection with context-selective mask generalization. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2523–2532, Online. Association for Computational Linguistics.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

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.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022. Learning cross-task dependencies for joint extraction of entities, events, event arguments, and relations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9349–9360, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Gennaro Nolano, Moritz Blum, Basil Ell, and Philipp Cimiano. 2024. Pointing out the shortcomings of relation extraction models with semantically motivated adversarials. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 12809–12820, Torino, Italia. ELRA and ICCL.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1946–1958, Vancouver, Canada. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Libo Qin, Qiguang Chen, Xiachong Feng, Yang Wu, Yongheng Zhang, Yinghui Li, Min Li, Wanxiang Che, and Philip S Yu. 2024. Large language models meet nlp: A survey. *arXiv preprint arXiv:2405.12819*.

Sylvestre-Alvise Rebuffi, Sven Gowal, Dan Andrei Calian, Florian Stimberg, Olivia Wiles, and Timothy A Mann. 2021. Data augmentation can improve robustness. *Advances in Neural Information Processing Systems*, 34:29935–29948.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *ECML/PKDD*.

Dan Roth and Wen tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Conference on Computational Natural Language Learning*.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code llama: Open foundation models for code. *Preprint*, arXiv:2308.12950.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *Preprint*, arXiv:cs/0306050.

Taneeya Satyapanich, Francis Ferraro, and Tim Finin. 2020. Casie: Extracting cybersecurity event information from text. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8749–8757.

Jiaming Song, Qinsheng Zhang, Hongxu Yin, Morteza Mardani, Ming-Yu Liu, Jan Kautz, Yongxin Chen, and Arash Vahdat. 2023. Loss-guided diffusion models for plug-and-play controllable generation. In *International Conference on Machine Learning*, pages 32483–32498. PMLR.

Akshay Srinivasan and Sowmya Vajjala. 2023. A multilingual evaluation of NER robustness to adversarial inputs. In *Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023)*, pages 40–53, Toronto, Canada. Association for Computational Linguistics.

InternLM Team. 2023. Internlm: A multilingual language model with progressively enhanced capabilities.

Asahi Ushio and Jose Camacho-Collados. 2021. T-NER: An all-round python library for transformer-based named entity recognition. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 53–62, Online. Association for Computational Linguistics.

C. Walker and Linguistic Data Consortium. 2005. *ACE 2005 Multilingual Training Corpus*. LDC corpora. Linguistic Data Consortium.

Haohan Wang, Zeyi Huang, Xindi Wu, and Eric Xing. 2022. Toward learning robust and invariant representations with alignment regularization and data augmentation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1846–1856.

Xiao Wang, Wei Zhou, Can Zu, Han Xia, Tianze Chen, Yuan Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, Jihua Kang, J. Yang, Siyuan Li, and Chunsai Du. 2023. Instructuie: Multi-task instruction tuning for unified information extraction. *Preprint*, arXiv:2304.08085.

Tino Werner. 2023. Loss-guided stability selection. *Advances in Data Analysis and Classification*, pages 1–26.

Xinglin Xiao, Yijie Wang, Nan Xu, Yuqi Wang, Hanxuan Yang, Minzheng Wang, Yin Luo, Lei Wang, Wenji Mao, and Daniel Zeng. 2024. Yayi-uie: A chat-enhanced instruction tuning framework for universal information extraction. *Preprint*, arXiv:2312.15548.

Zhiheng Yan, Chong Zhang, Jinlan Fu, Qi Zhang, and Zhongyu Wei. 2021. A partition filter network for joint entity and relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 185–197, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.

A Prompts for Adversarial Example Generation

Replace Entity, Triple, and Trigger To generate adversarial examples, we replace the entities, relation triples, and event triggers in the samples following specific rules designed for LLMs. The prompts used are shown in Figure 5.

Change Context We change the context of NER and ED samples based on the rules mentioned in Section 3.1 to generate high-quality adversarial examples. The prompt used for this context modification is presented in Figure 6.

Extend Sentence Using the prompt in Figure 7, we generate extended versions of NER, RE, and ED samples.

B RUIE-Bench Details

Sampling Details Considering that generating adversarial examples for all test sets of NER, RE and ED tasks would result in a large number of samples and costs, as well as significant evaluation expenses, we conduct sampling on the test sets of the selected datasets to balance evaluation costs and accuracy. The main principle we follow is to ensure that the sampled subsets maintain the same distribution. Specifically, for NER, we select 1,000 samples, including 134 from ACE05-Ent, 294 from CoNLL03, and 572 from WikiANN. For RE, we choose 800 samples, with 230 from ACE05-Rel and 570 from NYT. For ED, we choose 900 samples, with 676 from ACE05-Evt and 224 from CASIE.

Statistics Based on the sampled data, we utilize the perturbation methods described in Section 3.1 to generate adversarial examples, which are then used to construct the RUIE-Bench dataset. The detailed statistics of the resulting dataset are presented in Table 4.

Task	Dataset	Original Data		RUIE-Bench Data		
		Type	Test size	Type	Sampling size	Data size
NER	ACE05	7	1,060	7	134	670
	CoNLL03	4	3,453	4	294	1,470
	WikiANN	3	10,000	3	572	2,860
RE	ACE05	6	2,050	6	230	920
	NYT	24	5,000	24	570	2,280
ED	ACE05	33	676	33	676	3,380
	CASIE	5	3,208	5	224	1,120

Table 4: Statistics of the RUIE-Bench dataset.

Computational complexity We use GPT-4 to generate adversarial examples. The primary computational cost during generation is concentrated on the self-attention mechanism, whose computational complexity is typically $O(l^2)$ (where l denotes the length of the generated sequence). However, due to caching optimizations and other strategies employed during generation, the time required to generate a single token can be approximated as constant.

To facilitate quantification of the generation time, we conduct the following analysis: Assume the dataset contains a total of n examples. During adversarial example generation, an average of l tokens are generated per example. LLM takes a constant time k to generate each token. Then, the total time T can be expressed as: $T = n \cdot l \cdot k$. During the data generation phase, we did not use any GPU-based computational resources. Instead, we relied on web-based API services to perform the generation. To construct RUIE-Bench, the LLM generated a total of 7,300 adversarial examples. Excluding time consumed due to response failures and other factors, the total generation time was approximately 341 minutes. On average, generating a single example took approximately 2.8 seconds, with each example containing an average of 38 generated tokens.

C Augment Data Generation

We utilize the original training sets from all datasets included in RUIE-Bench as the foundation for constructing augmented data. Considering the large scale of these training sets and the diverse types of perturbations involved, we perform random sampling to select 30% of the training data from each dataset. On this sampled subset, we apply all perturbation injection methods defined in RUIE-Bench to generate a variety of augmented samples. To efficiently generate a substantial volume of augmented data while controlling computational costs, we opt to use Deepseek-V3 (Liu et al., 2024) as our data generation model instead of the more resource-intensive GPT-4 (OpenAI., 2023). The detailed statistics for all datasets before and after augmentation are summarized in Table 5.

Task	Dataset	Data size	
		Original	Augment
NER	ACE05	7,299	10,948
	CoNLL03	14,041	21,061
	WikiANN	20,000	30,000
RE	ACE05	10,051	12,061
	NYT	56,196	67,435
ED	ACE05	19,216	28,824
	CASIE	11,189	16,784

Table 5: Statistics of Augmented Data.

D Few-shot Prompts for UIE

Taking the NER task as an example, we illustrate the prompt design used for extraction, as shown in Table 7. The prompt is composed of five main components: (1) Task Objective, which states the goal of the task; (2) Entity Types, which define the categories of entities to be extracted along with their descriptions; (3) Output Formatting, which specifies the expected structure of the output; (4) Examples, which present a few demonstration instances randomly sampled from the training set to guide the model; and (5) Current Task, which includes the input sentence to be processed. The RE and ED tasks follow a similar prompt structure, differing only in the defined types and output formats, as shown in Table 8 and Table 9, respectively.

E Comparison of Data Augmentation Training Models

To evaluate the effectiveness of the model trained with the LDA strategy, we conduct a comprehensive comparison against two baselines: the non-augmented training model, denoted as KnowCoder$_{\text{partial}}$, and the model trained with the full set of augmented data, referred to as KnowCoder-Robust. The results are summarized in Table 6, where the best performance under each specific perturbation for every dataset is highlighted in bold. As shown in the table, models trained with any form of data augmentation consistently outperform the non-augmented baseline across all perturbation types and datasets. Notably, KnowCoder-Robust$_{\text{LDA}}$ achieves performance on par with KnowCoder-Robust, indicating that the LDA strategy can effectively support data augmentation training with significantly reduced augmentation cost.

F Case Study

As shown in Figures 8 and 9, we present extraction cases covering all types of perturbations to analyze model behavior under various perturbation settings. Specifically, for the Mask Context, Extend Sentence, Typo Injection, and Lowercase Conversation perturbations, we illustrate examples under the NER task. Correctly entities, relation triples, and event triggers are highlighted in red.

In the cases of Replace Entity, Triple, and Trigger perturbations, it is relatively easy to identify the correct entities, relations, or events based on contextual information, as these cases contain clear contextual clues that point to the correct interpretation. However, in practice, most models fail to make correct predictions on these adversarial examples. Only KnowCoder-Robust$_{\text{LDA}}$ trained with data augmentation is able to generate accurate predictions. This suggests that these models tend to rely on memorization rather than using context for inference.

In the Change Context and Extend Sentence perturbation cases, the adversarial examples involve only simple modifications, such as replacing a single word or slightly extending the sentence. Nevertheless, models without data augmentation training produce incorrect or redundant predictions. Similarly, in the Typo Injection and Lowercase Conversion cases, the adversarial examples introduce only minor typographical errors or convert parts of the text to lowercase, yet models without augmentation still yield incorrect or missing predictions. These observations highlight the models' sensitivity to input variations.

These cases collectively demonstrate that current LLMs and information extraction models suffer from insufficient robustness.

Task	Dataset	Perturbation Type	Model		
			KnowCoder$_{\text{partial}}$	KnowCoder-Robust	KnowCoder-Robust$_{\text{LDA}}$
NER	ACE05	None	79.2	**82.5**	82.1
		Replace Entity	72.3	**78.9**	76.3
		Change Context	74.2	**80.6**	78.9
		Extend Sentence	72.1	**82.7**	81.4
		Typo Injection	76.8	**82.7**	80.1
		Lowercase Conversion	77.0	82.0	**82.4**
	Conll03	None	91.7	93.0	**93.2**
		Replace Entity	80.5	90.9	**92.7**
		Change Context	87.2	90.2	**90.6**
		Extend Sentence	91.0	93.0	**94.0**
		Typo Injection	90.0	**92.5**	91.4
		Lowercase Conversion	85.1	**89.7**	87.5
	WikiANN	None	81.9	83.1	**83.4**
		Replace Entity	70.8	**76.9**	76.5
		Change Context	77.8	80.8	**83.4**
		Extend Sentence	78.2	**83.9**	83.8
		Typo Injection	79.4	83.6	**83.9**
		Lowercase Conversion	76.4	**82.6**	82.2
RE	ACE05	None	58.6	**63.5**	59.3
		Replace Triple	54.9	**61.6**	57.7
		Extend Sentence	54.1	**62.3**	62.1
		Typo Injection	53.9	59.7	**61.3**
		Lowercase Conversion	56.5	56.9	**61.6**
	NYT	None	90.6	91.1	**91.5**
		Replace Triple	62.9	67.9	**70.0**
		Extend Sentence	89.2	**91.2**	90.8
		Typo Injection	82.8	**89.8**	89.4
		Lowercase Conversion	51.3	89.0	**89.3**
ED	ACE05	None	69.1	**70.2**	69.4
		Replace Trigger	55.5	65.7	**66.0**
		Change Context	66.1	67.1	**68.1**
		Extend Sentence	65.8	**70.5**	70.0
		Typo Injection	66.8	68.3	**68.8**
		Lowercase Conversion	67.8	**69.6**	68.9
	CASIE	None	67.0	**68.7**	68.2
		Replace Trigger	53.5	64.9	**65.5**
		Change Context	63.7	**67.5**	**67.5**
		Extend Sentence	62.9	**68.2**	67.9
		Typo Injection	62.0	67.2	**67.5**
		Lowercase Conversion	64.0	**69.1**	68.6
All	All	**Average**	71.3	**77.1**	**77.1**

Table 6: Comparison of KnowCoder$_{\text{partial}}$ and different data augmentation training models on RUIE-Bench.

NER Task

{example data}
The above is a Named Entity Recognition data entry, where "sentence" contains the sentence information, and "entities" contains the entity label information. Now, based on the following rules, the entities in the sentence need to be changed:
1. Change the entity while keeping the original entity type.
2. The changed entities should be difficult and uncommon, and the number of entity words can vary.
3. Only change the entity content, do not change other content.
4. Changed entities should be updated in both "sentence" and "entities". Please output in the following format and do not output any extra content. {"sentence": "", "entities": []}

RE Task

{example data}
The above is a Relation Extraction data entry, where "sentence" contains the sentence information, and "relations" contains the relational triple information. Now, based on the following rules, the head and tail entities in the relational triples need to be changed:
1. Change the head and tail entities while keeping the original entity type.
2. The changed entity should be significantly different from the original entity and can vary in length.
3. Only change the head and tail entities, do not change other content.
4. Changed content should be updated in both "sentence" and "relations". Please output in the following format and do not output any extra content. {"sentence": "", "relations": []}

ED Task

{example data}
The above is a Event Detection data entry, where "sentence" contains the sentence information, and "events" contains the event trigger information. Now, based on the following rules, the event triggers in the sentence need to be changed:
1. Change the trigger while keeping the original event type.
2. The changed trigger should be significantly different from the original trigger and can vary in length.
3. Only change the trigger content, do not change other content.
4. Changed triggers should be updated in both "sentence" and "events". Please output in the following format and do not output any extra content. {"sentence": "", "events": []}

Figure 5: Prompts for Replace Entity, Triple, and Trigger.

NER Task

{example data}
The above is a Named Entity Recognition data entry, where the "sentence" contains the sentence information, and the "entities" contains the entity label information. The sentence containing 1-4 [MASK] tokens. These [MASK] tokens mask certain words, with each [MASK] token potentially masking one or more words. You need to generate some challenging words to replace these [MASK] tokens to create a difficult sentence. The generated words can not contain any entity information.
You need to provide 3 predictions for each [MASK] token. Please output in the following format, without any additional content:
[MASK]1: 1: " ", 2: " ", 3: " "
......
[MASK]n: 1: " ", 2: " ", 3: " "

ED Task

{example data}
The above is an Event Detection data entry, where the "sentence" contains the sentence information, and the "events" contains the event trigger information. The sentence containing 1-4 [MASK] tokens. These [MASK] tokens mask certain words, with each [MASK] token potentially masking one or more words. You need to generate some challenging words to replace these [MASK] tokens to create a difficult sentence. The generated words can not contain any event triger information.
You need to provide 3 predictions for each [MASK] token. Please output in the following format, without any additional content:
[MASK]1: 1: " ", 2: " ", 3: " "
......
[MASK]n: 1: " ", 2: " ", 3: " "

Figure 6: Prompts for Change Context.

NER Task

{example data}
The above is a Named Entity Recognition data entry, where the "sentence" contains the sentence information, and the "entities" contains the entity label information.
Now, you need to ensure that the original sentence remains unchanged while adding semantically related content at the beginning or end of the sentence, with a preference for the end. The additional content should not be too simplistic and not not introduce new entity information.
Please output in the following format, without any additional content: {"sentence": "", "entities": []}, where "sentence" should be the expanded sentence, and "entities" should contain the original entity label information.

RE Task

{example data}
The above is a Relation Extraction data entry, where the "sentence" contains the sentence information, and the "relations" contains the relational triple information.
Now, you need to ensure that the original sentence remains unchanged while adding semantically related content at the end or beginning of the sentence, with a preference for the end. The additional content should not be too simplistic, and should not introduce new relational triple information.
Please output in the following format, without any additional content: {"sentence": "", "relations": []}, where "sentence" should be the expanded sentence, and "relations" should contain the original relational triple information.

ED Task

{example data}
The above is an Event Detection data entry, where the "sentence" contains the sentence information, and the "events" contains the event trigger information.
Now, you need to ensure that the original sentence remains unchanged while adding semantically related content at the end or beginning of the sentence, with a preference for the end. The additional content should not be too simplistic and not contain new event information.
Please output in the following format, without any additional content: {"sentence": "", "events": []}, where "sentence" should be the expanded sentence, and "events" should contain the original event trigger information.

Figure 7: Prompts for Extend Sentence.

Original NER example

Sentence: France managed third place with 0.68 percent in the 16-nation world government bond index.
Entities:[{"name": "France", "type": "LOC"}]

Replace Entity

Adversarial example

Sentence: Atlantis managed third place with 0.68 percent in the 16-nation world government bond index.
Entities:[{"name": "Atlantis", "type": "LOC"}]

Model	Original example prediction	Adversarial example prediction
LLama3-8B-Instruct	[{"name": "France", "type": "LOC"}]	[{"name": "Atlantis", "type": "LOC"}, {"name": "world", "type": "LOC"}]
GLM4-Plus	[{"name": "France", "type": "LOC"}]	[{"name": "Atlantis", "type": "ORG"}]
Stanza	[{"name": "France", "type": "LOC"}]	[{"name": "Atlantis", "type": "ORG"}]
KnowCoder-7B	[{"name": "France", "type": "LOC"}]	[{"name": "Atlantis", "type": "ORG"}]
KnowCoder-7B-Robust$_{LDA}$	[{"name": "France", "type": "LOC"}]	[{"name": "Atlantis", "type": "LOC"}]

Original ED example

Sentence: Over an hour of talks, we asserted the will of both parties (Israel and the Arab world) to do everything to return to the negotiating table.
Events:[{"trigger": "talks", "type": "Contact"}]

Replace Trigger

Adversarial example

Sentence: Over an hour of discussion, we asserted the will of both parties (Israel and the Arab world) to do everything to return to the negotiating table.
Events: [{"trigger": "discussion", "type": "Contact"}]

Model	Original example prediction	Adversarial example prediction
LLama3-8B-Instruct	[{"trigger": "talks", "type": "Contact"}]	NULL
GLM4-Plus	[{"trigger": "talks", "type": "Contact"}, {"trigger": "negotiating", "type": "Contact"}]	[{"trigger": "discussion", "type": "Contact"}, {"trigger": "negotiating", "type": "Contact"}]
EEQA	NULL	NULL
KnowCoder-7B	[{"trigger": "talks", "type": "Contact"}]	NULL
KnowCoder-7B-Robust$_{LDA}$	[{"trigger": "talks", "type": "Contact"}]	[{"trigger": "discussion", "type": "Contact"}]

Original RE example

Sentence: Mr. Castano explains matter-of-factly that on the ninth of August, 1994, I traveled to Bogota and directed the commando unit that executed Senator Manuel Cepeda Vargas.
Relations: [{"head": "Manuel Cepeda Vargas", "relation": "PlaceOfDeath", "tail": "Bogota"}]

Replace Triple

Adversarial example

Sentence: Mr. Castano explains matter-of-factly that on the ninth of August, 1994, I traveled to Quito and directed the commando unit that executed Senator Julio Cesar Trujillo.
Relations: [{"head": "Julio Cesar Trujillo", "relation": "PlaceOfDeath", "tail": "Quito"}]

Model	Original example prediction	Adversarial example prediction
LLama3-8B-Instruct	[{"head": "Bogota", "relation": "ContainsTheAdministrativeTerritorialEntity", "tail": "Senator Manuel Cepeda Vargas"}]	[{"head": "Quito", "relation": "ContainsTheAdministrativeTerritorialEntity", "tail": "Senator Julio Cesar Trujillo"}]
GLM4-Plus	[{"head":"Mr. Castano", "relation": "Residence", "tail": "Bogota"}]	[{"head":"Mr. Castano","relation": " Residence", "tail": " Quito"}]
PFN	[{"head": "Manuel Cepeda Vargas", "relation": "Residence", "tail": "Bogota"}]	[{"head": "Julio Cesar Trujillo", "relation": "PlaceOfBrith", "tail": "Quito"}]
KnowCoder-7B	[{"head": "Manuel Cepeda Vargas", "relation": "Residence", "tail": "Bogota"}]	[{"head": "Julio Cesar Trujillo", "relation": "PlaceOfBrith", "tail": "Quito"}]
KnowCoder-7B-Robust$_{LDA}$	[{"head": "Manuel Cepeda Vargas", "relation": " PlaceOfDeath ", "tail": "Bogota"}]	[{"head": "Julio Cesar Trujillo", "relation": "PlaceOfDeath", "tail": "Quito"}]

Figure 8: Example cases for the Replace Entity, Triple and Trigger.

Original NER example

Sentence: It was the second costly blunder by Hyderabad in four hours.
Entities: [{"name": "Hyderabad", "type": "LOC"}]

Change Context

Adversarial example

Sentence: It was the second costly blunder by Hyderabad in four minutes.
Entities: [{"name": "Hyderabad", "type": "LOC"}]

Model	Original example prediction	Adversarial example prediction
LLama3-8B-Instruct	[{"name": "Hyderabad", "type": "LOC"}]	[{"name": "Hyderabad", "type": "ORG"}]
GLM4-Plus	[{"name": "Hyderabad", "type": "LOC"}]	[{"name": "Hyderabad", "type": "LOC"}]
Stanza	[{"name": "Hyderabad", "type": "LOC"}]	[{"name": "Hyderabad", "type": "ORG"}]
KnowCoder-7B	[{"name": "Hyderabad", "type": "LOC"}]	[{"name": "Hyderabad", "type": "ORG"}]
KnowCoder-7B-Robust$_{LDA}$	[{"name": "Hyderabad", "type": "LOC"}]	[{"name": "Hyderabad", "type": "LOC"}]

Original NER example

Sentence: Slaughter steers and heifers not tested, compared with Thursday 's close, USDA said.
Entities: [{"name": "USDA", "type": "ORG"}]

Typo Injection

Adversarial example

Sentence: Slalghter steers and heifeors not tested, compared with Thursday 's close, USDA said.
Entities: [{"name": "USDA", "type": "ORG"}]

Model	Original example prediction	Adversarial example prediction
LLama3-8B-Instruct	[{"name": "USDA", "type": "ORG"}]	NULL
GLM4-Plus	[{"name": "USDA", "type": "ORG"}]	NULL
Stanza	[{"name": "USDA", "type": "ORG"}]	[{"name": "USDA", "type": "ORG"}]
KnowCoder-7B	[{"name": "USDA", "type": "ORG"}]	[{"name": "USDA", "type": "ORG"}, {"name": "Slalghter", "type": "PER"}]
KnowCoder-7B-Robust$_{LDA}$	[{"name": "USDA", "type": "ORG"}]	[{"name": "USDA", "type": "ORG"}]

Original NER example

Sentence: Today the album is distributed by KMP Holdings.
Entities: [{"name": "KMP Holdings", "type": "ORG"}]

Extend Sentence

Adversarial example

Sentence: Today the album is distributed by KMP Holdings, a well-known music distribution company.
Entities: [{"name": "KMP Holdings", "type": "ORG"}]

Model	Original example prediction	Adversarial example prediction
LLama3-8B-Instruct	[{"name": "KMP Holdings", "type": "ORG"}]	[{"name": "KMP Holdings", "type": "ORG"}, {"name": "a well-known music distribution company", "type": "ORG"}]
GLM4-Plus	[{"name": "KMP Holdings", "type": "ORG"}]	[{"name": "KMP Holdings", "type": "ORG"}]
Stanza	[{"name": "KMP Holdings", "type": "ORG"}]	[{"name": "KMP Holdings", "type": "ORG"}]
KnowCoder-7B	[{"name": "KMP Holdings", "type": "ORG"}]	[{"name": "KMP Holdings, a well-known music distribution company.", "type": "ORG"}]
KnowCoder-7B-Robust$_{LDA}$	[{"name": "KMP Holdings", "type": "ORG"}]	[{"name": "KMP Holdings", "type": "ORG"}]

Original NER example

Sentence: He was born on November 30, 1582, most likely in Speyer, where his father was an official in the Reichskammergericht.
Entities: [{"name": "Speyer", "type": "LOC"}, {"name": "Reichskammergericht", "type": "ORG"}]

Lowercase Conversation

Adversarial example

Sentence: He was born on november 30, 1582, most likely in speyer, where his father was an official in the reichskammergericht.
Entities: [{"name": "speyer", "type": "LOC"}, {"name": "reichskammergericht", "type": "ORG"}]

Model	Original example prediction	Adversarial example prediction
LLama3-8B-Instruct	[{"name": "Speyer", "type": "LOC"}, {"name": "Reichskammergericht", "type": "ORG"}]	[{"name": "Speyer", "type": "LOC"}, {"name": "Reichskammergericht", "type": "ORG"}]
GLM4-Plus	[{"name": "Speyer", "type": "LOC"}, {"name": "Reichskammergericht", "type": "ORG"}]	[{"name": "Speyer", "type": "LOC"}, {"name": "Reichskammergericht", "type": "ORG"}]
Stanza	[{"name": "Speyer", "type": "LOC"}, {"name": "Reichskammergericht", "type": "LOC"}]	[{"name": "speyer", "type": "LOC"}]
KnowCoder-7B	[{"name": "Speyer", "type": "LOC"}, {"name": "Reichskammergericht", "type": "ORG"}]	[{"name": "reichskammergericht", "type": "ORG"}]
KnowCoder-7B-Robust$_{LDA}$	[{"name": "Speyer", "type": "LOC"}, {"name": "Reichskammergericht", "type": "ORG"}]	[{"name": "speyer", "type": "LOC"}, {"name": "reichskammergericht", "type": "ORG"}]

Figure 9: Example cases for the Mask Context, Extend Sentence, Typo Injection and Lowercase Conversation.

PROMPT FOR FEW-SHOT NER.

```
## Task Objective
Perform Named Entity Recognition (NER) on input sentences to extract entities of these types:

##Entity Types:
{entity_type 1}: {description 1}
{entity_type 2}: {description 2}
......
{entity_type n}: {description n}
The entity type here refer to the entity types specific to a given dataset, where the description represents the entity type information. When used, it should be replaced with the actual entity types and corresponding descriptions based on the specific dataset.

## Output Formatting
1. Return a JSON list of entities
2. Each entity must include:
- **Type**: Entity category (exact uppercase labels)
- **Name**: Original text span
3. Return empty list if no entities found

## Examples (10-shot)
1. Input: {example sentence 1}
   Output: {recognition result 1}
2. Input: {example sentence 2}
   Output: {recognition result 2}
......
10. Input: {example sentence 10}
   Output: {recognition result 10}
The example sentences are selected from the training set, and the recognition results should fully comply with the defined Output Formatting.

## Current Task
Input: {test sentence}
Output:
The input here should be the sentences to be tested, and the output should be the model's recognition results.
```

Table 7: Prompt for Few-Shot NER.

PROMPT FOR FEW-SHOT RE.

```
## Task Objective
Perform Relation Extraction (RE) on input sentences to extract relational triples of these types:

##Relation Types:
{relation_type 1}: {description 1}
{relation_type 2}: {description 2}
......
{relation_type n}: {description n}
The relation type here refer to the relation types specific to a given dataset, where the description represents the relation type information. When used, it should be replaced with the actual relation types and corresponding descriptions based on the specific dataset.

## Output Formatting
1. Return a JSON list of relational triples
2. Each relational triple must include:
- **Head**: Original head entity span
- **Type**: Relation category (exact uppercase labels)
- **Tail**: Original tail entity span
3. Return empty list if no relational triples found

## Examples (10-shot)
1. Input: {example sentence 1}
   Output: {recognition result 1}
2. Input: {example sentence 2}
   Output: {recognition result 2}
......
10. Input: {example sentence 10}
   Output: {recognition result 10}
The example sentences are selected from the training set, and the recognition results should fully comply with the defined Output Formatting.

## Current Task
Input: {test sentence}
Output:
The input here should be the sentences to be tested, and the output should be the model's recognition results.
```

Table 8: Prompt for Few-Shot RE.

PROMPT FOR FEW-SHOT ED.

```
## Task Objective
Perform Event Detection (ED) on input sentences to extract events of these types:

##Event Types:
{event_type 1}: {description 1}
{event_type 2}: {description 2}
......
{event_type n}: {description n}
The event type here refer to the event types specific to a given dataset, where the description represents the event type information. When used, it should be replaced with the actual event types and corresponding descriptions based on the specific dataset.

## Output Formatting
1. Return a JSON list of events
2. Each event must include:
- **Type**: Event category (exact uppercase labels)
- **Trigger**: Event trigger span
3. Return empty list if no event found

## Examples (10-shot)
1. Input: {example sentence 1}
   Output: {recognition result 1}
2. Input: {example sentence 2}
   Output: {recognition result 2}
......
10. Input: {example sentence 10}
   Output: {recognition result 10}
The example sentences are selected from the training set, and the recognition results should fully comply with the defined Output Formatting.

## Current Task
Input: {test sentence}
Output:
The input here should be the sentences to be tested, and the output should be the model's recognition results.
```

Table 9: Prompt for Few-Shot ED.