LLM's Weakness in NER Doesn't Stop It from Enhancing a Stronger SLM

Weilu Xu, Renfei Dang, Shujian Huang
National Key Laboratory for Novel Software Technology, Nanjing University, China
{weiluxu,dangrf}@smail.nju.edu.cn, huangsj@nju.edu.cn

Abstract

Large Language Models (LLMs) demonstrate strong semantic understanding ability and extensive knowledge, but struggle with Named Entity Recognition (NER) due to hallucination and high training costs. Meanwhile, supervised Small Language Models (SLMs) efficiently provide structured predictions but lack adaptability to unseen entities and complex contexts. In this study, we investigate how a relatively weaker LLM can effectively support a supervised model in NER tasks. We first improve the LLM using LoRA-based fine-tuning and similarity-based prompting, achieving performance comparable to a SLM baseline. To further improve results, we propose a fusion strategy that integrates both models: prioritising SLM's predictions while using LLM guidance in low confidence cases. Our hybrid approach outperforms both baselines on three classic Chinese NER datasets.

1 Introduction

Large Language Models (LLMs) (Smith et al., 2022; Du et al., 2022; Rae et al., 2021) have shown remarkable abilities on various NLP applications. LLMs can understand complex semantic information and have extensive knowledge.

However, LLMs often suffer from the hallucination problem, where they confidently classify non-entity words as entities (Wang et al., 2023). In addition, they require significantly higher training costs to achieve performance comparable to supervised Small Language Models (SLMs) (Zhou et al., 2024). In contrast, SLMs can achieve reasonable levels of performance with lower training costs, but struggles with unseen entities and lacks strong semantic understanding in complex contexts.

This raises an important question: Can a relatively weaker LLM in a particular task still provide useful guidance to a smaller but supervised model? If so, integrating LLMs' broad knowledge with SLM's structured learning could boost NER performance while keeping training costs manageable.

In this study, we first trained SLM and LLM baselines with reasonable computational cost. To enhance LLM performance, we applied LoRA-based SFT and retrieved similar examples as prompts for task-specific guidance. These improvements brought LLM closer to the SLM baseline. We then proposed a fusion strategy: SLM's prediction was preferred unless its confidence was low, in which case the LLM output guided the final result. Figure 1 illustrates this process.

Our final hybrid model outperformed both individual baselines, demonstrating that leveraging LLM knowledge can effectively enhance SLM's structured predictions while maintaining efficiency.

2 Related Work

2.1 Named Entity Recognition

Named Entity Recognition (NER) is a tagging task where each word in a sentence is labeled to indicate whether it is part of a named entity and its corresponding type. A common approach to NER is to model it as a sequence labeling problem, where a multi-layer perceptron with a softmax layer serves as the tag decoder, framing the task as multi-class classification. Additionally, Conditional Random Fields (CRFs) (Liu et al., 2021), which conditionally model dependencies between labels, have been widely used in feature-based supervised learning methods. In addition, deep learning has become a widely used approach for NER, and advances in related upstream and downstream tasks, such as sequence tagging and entity linking, have further improved NER performance.(Roy, 2021)

2.2 Collaboration of Large and Small Models

Recent advances in pre-trained large-scale models have enabled training on vast amounts of data, making them adaptable to diverse downstream tasks

Proceedings of the Second Ancient Language Processing Workshop associated with NAACL 2025, pages 117–122

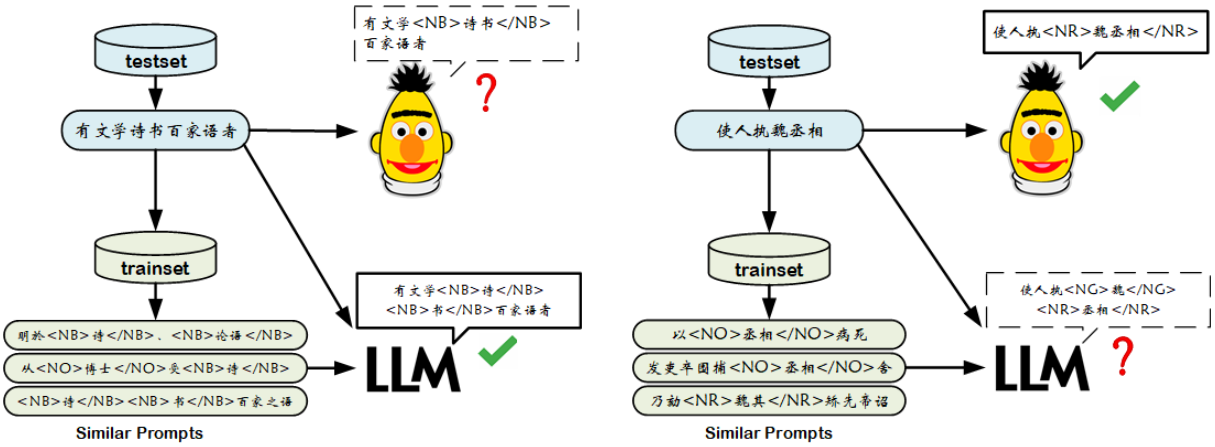

Figure 1: Two cases of the hybrid system. The green tick in the picture represents that the model's annotation for this sentence is confident and correct, while the red question mark represents that its perplexity is high, and it is actually incorrect. Therefore, based on the rules (detailed in Section 4), we select the results with high confidence from both sides. More discussions about these two examples are in Section 5.2.

(Bommasani et al., 2021). However, studies (Ma et al., 2023) suggest that while LLMs excel in extremely low-resource scenarios, they are not always effective for few-shot information extraction. In particular, combining LLMs with SLMs significantly improves performance in difficult cases, demonstrating the potential of hybrid approaches in NER.

3 Baseline Approaches for NER

3.1 SLM-Based Token Classification

To set a baseline, in the closed modality, we use *GujiRoBERTa_jian_fan*[1], a BERT model pretrained on massive traditional Chinese corpus. For the training data set used to adapt to the downstream NER task, there are three different data sets: Dataset A (from Shiji), Dataset B (from the Twenty-Four Histories), and Dataset C (from Traditional Chinese Medicine Classics).

During data processing, we mainly faced two key issues: sentence segmentation and character tokenization. To better leverage the context information, we use periods, question marks, and exclamation marks instead of a fixed maximum length as the end of a sentence. In terms of tokenization, a character within a single label may be split into multiple tokens, which requires careful processing to keep the boundary information. Details are in table 1.

[1] https://github.com/hsc748NLP/GujiBERT-and-GujiGPT

Original label (for one char)	Assigned new labels (for multiple tokens)
$[B-]$	$[B-]\ [M-]_{*(i-1)}$
$[M-]$	$[M-]_{*i}$
$[E-]$	$[M-]_{*(i-1)}\ [E-]$
$[S-]$	$[B-]\ [M-]_{*(i-2)}\ [E-]$

Table 1: Details of changing labels for multiple-token characters. "B-", "M-", "E-", "S-" are prefixes of labels. i is the number of tokens for a single character.

3.2 Large Model-Assisted NER

Although LLMs show strong performance in a wide range of tasks, their performance on NER is still significantly below the supervised baselines. To improve the performance of the LLMs in the NER task, we explored two key strategies: LoRA fine-tuning for better model adaptation and similarity-based prompting for more effective few-shot learning. Both methods have a significant impact on performance improvement.

3.2.1 LoRA fine-tuning

Considering the balance between performance and cost, we employed Low-Rank Adaptation (LoRA) (Hu et al., 2022) to fine-tune Qwen2.5-14B-Instruct.

As for the data, we have transformed the token-level training set into a special format that can be understood by LLMs. The specific format is as follows:

<NG>楚</NG>使怒去，归告<NR>怀王</NR>。

In this case, "楚" "怀" "王" are officially annotated as "S-NG" "B-NR" and "E-NR" in the training data set, so we use pairs of "<X></X>" to denote an "X" category.

3.2.2 Similarity-based prompting

Beyond standard fine-tuning, we explored a retrieval-based prompting approach to enhance in-context learning. Specifically, we utilized SIKU-BERT/sikuroberta[2] to generate sentence embeddings and performed similarity matching to retrieve the top 5 most similar sentences from the training set for each test sample. Unlike traditional few-shot prompting, which relies on a fixed set, similarity prompting dynamically selects contextually relevant examples, ensuring better alignment with the input instance. This approach effectively solves the transition labelling problem common to large language models on NER tasks.

4 Model Fusion: Combining SLM and LLM Outputs

4.1 Method

To leverage the strengths of both SLM-based token classification and LLM-assisted NER, we propose a fusion strategy that integrates their outputs. This method selects the more confident answer when the answers given by the two models are found to be in disagreement.

4.1.1 LLM's Category & Boundary Probabilities

In order to better collaborate with and compare against the small model, we need to define the category probabilities and boundary probabilities for the annotation results of the LLM.

The category probability is defined as the probability obtained after performing softmax normalization on the logits of the positions where the tokens corresponding to that category first appear, while the boundary probability is defined as the average of the probabilities of the first "<" token (denoting the start of an annotation), the first "</" token (denoting the end of one annotation), and the token preceding each of them (denoting whether to start or end an annotation). We will still use the following example:

<NG>楚</NG>使怒去，归告<NR>怀王</NR>。

In this case, "楚" "怀" "王" are officially annotated as "S-NG" "B-NR" and "E-NR" in the training data set, so we use pairs of "<X></X>" to denote an "X" category. So the category probability for "楚" is $Softmax(Logit(NG))$; and the boundary probability for "怀王" is average of the softmaxed logits of tokens "告" "<" "王" and "</". We do need the token preceding "<", in that the LLM may hesitate whether to start a new entity from token "告" or "怀"; we also need the token preceding "</", in that the LLM may hesitate whether to end this entity in token "怀" or "王".

4.1.2 Hyperparameters and formulas

It was observed that the LLM confidence gap was minimal; therefore, the decision was made to scale it using the exponent of e. With regard to the values of the hyperparameters, a search was conducted across the training sets for the optimal results. If $S_S < S_L$, choose the results of SLM; otherwise, let the LLM's results provide guidance. Here is the final formulas.

$$\text{Compare}(S_S, S_L) = S_S < S_L \tag{1}$$

where:

$$S_S = 1 - \text{Entropy}(\text{SLM}) \tag{2}$$

$$S_L = \lambda \cdot e^{P(\text{LLM})} \tag{3}$$

5 Experiments

All of our experiments were conducted on at most 4 NVIDIA A6000 GPUs. We train the SLM on training set for 10 epochs, with a batch size of 64 and learning rate of 2e-5. For the SFT of the LLM, we adopted a batch size of 64, learning rate of 2e-5, a LoRA rank of 64 and an alpha of 128. All the evaluation results below are token-level macro scores (on a 10% test set splited from train set).

5.1 Results

We conducted ablation experiments for every approach and demonstrated that all approaches were helpful in improving the ability of LLM to perform the NER task on all three test sets.

First, we show that both methods of improving LLM performance are effective. Details are in Table 3.

[2] https://huggingface.co/SIKU-BERT/sikuroberta

Character	Real Label	SLM Output (Entropy)	LLM Output (Category/Boundary Prob.)
魏	B-NR	B-NR (0.036)	S-NG (0.7815)
丞	M-NR	M-NR (0.038)	B-NR (0.9986)
相	E-NR	E-NR (0.033)	E-NR (-)
诗	B-NB	B-NB (2.537)	S-NB (0.9062)
书	E-NB	E-NB (2.425)	S-NB (0.9998)

Table 2: Cases of SLM guiding LLM and LLM guiding SLM. The one above shows the category probabilities of the LLM, and the one below shows the boundary probabilities of the LLM. Since the LLM only outputs category probabilities for the entire entity once, there is no corresponding category probability for the Chinese character "相" in line 3.

model	**A**	**B**	**C**
few shot	0.2626	0.4072	0.2492
Sim. prompt	0.4400	0.6367	0.5182
SFT + few shot	0.6737	0.8329	0.5878
SFT + Sim. prompt	**0.8350**	**0.8694**	**0.6381**

Table 3: Results of two approaches to improve the ability of LLM on NER task. Based on Qwen2.5-14B-Instruct. This 'Sim.' means 'Similarity', and 'SFT' refers to 'Supervised Fine-Tuning'.

Then, despite the close results of SLM and LLM, we merged the two with custom rules and got better results in comparison to both on all three test sets. Details are in Table 4.

Model	**A**	**B**	**C**
SLM	0.8305	0.8738	0.7207
Sim. Prompt	0.8350	0.8694	0.6381
Sim. Prompt + SLM	**0.8855**	**0.8824**	**0.7733**

Table 4: Results from the fusion of the LLM and the SLM model. "Sim. Prompt" refers to a fine-tuned LoRA SFT model with similarity-based prompting.

5.2 Case Study

To better demonstrate the effectiveness of our method, we will present two examples below. They are respectively the case where the SLM successfully guides the LLM and the case where LLM successfully guides the SLM. Table 2 presents two typical examples.

Case 1: SLM guiding LLM. The phrase "魏丞相" (the Prime Minister of the state of Wei) can be annotated as "Personal Name (NR)" or "Country (NS) + Personal Name (NR)". Although the former is better, it also depends on the annotation style. In this case, due to its professionalism, the SLM grasped the annotation style of the training set more accurately. Therefore, it provided the correct answer with a relatively low entropy (low uncertainty). In contrast, the LLM gave a wrong answer with a relatively low probability (which also reflects its lack of confidence in itself). In such situation, we follow the rules and adopt the result provided by the small-scale model.

Case 2: LLM guiding SLM. "诗"(*The Book of Songs*) and "书" (*The Book of History*) are two of the "Six Classics" in ancient China, which is our common cultural knowledge. Since the large-scale model has incorporated a vast amount of knowledge during the pre-training stage, it is highly likely that it has learned this common sense and can accurately annotate the Chinese characters "诗" and "书" as "S-NB" respectively. In contrast, due to the lack of this common sense, the small-scale model tends to mis-annotate things it doesn't recognize with a very high entropy (high uncertainty). According to our rules, this situation can also be successfully corrected.

6 Conclusions and Future Work

In this paper, we presents an efficient approach to enhancing LLM performance in NER to match a supervised SLM. Using LoRA fine-tuning and similarity-based prompting, we improved the LLM's entity recognition. We also introduced a fusion strategy that prioritizes SLM's predictions while leveraging LLM guidance when SLM's confidence is low. This hybrid approach consistently outperformed both baselines.

However, our method did not fully utilize the LLM's reasoning and analytical capabilities. In particular, enabling a 14B parameter model with limited domain knowledge of classical Chinese to self-correct remains challenging. Future work may explore ways to enhance LLM's domain adaptation, allowing it to better leverage contextual understanding and reasoning for collaborative NER frameworks.

References

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, and Emma Brunskill. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, and Orhan Firat. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*, pages 5547–5569. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Shuai Liu, Tenghui He, and Jianhua Dai. 2021. A survey of crf algorithm based knowledge extraction of elementary mathematics in chinese. *Mobile Networks and Applications*, 26(5):1891–1903.

Yubo Ma, Yixin Cao, YongChing Hong, and Aixin Sun. 2023. Large language model is not a good few-shot information extractor, but a good reranker for hard samples! *arXiv preprint arXiv:2303.08559*.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, and Susannah Young. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Arya Roy. 2021. Recent trends in named entity recognition (ner). *arXiv preprint arXiv:2101.11420*.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, and Vijay Korthikanti. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, and Guoyin Wang. 2023. Gpt-ner: Named entity recognition via large language models. *arXiv preprint arXiv:2304.10428*.

Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, and Lifang He. 2024. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *International Journal of Machine Learning and Cybernetics*. Advance online publication.

A Detailed Experimental Results

In this section we show detailed experimental results individually. Table 5, table 6 and table 7 are results of LLM Qwen2.5-14B-Instruct on NER tasks **A**, **B** and **C**. Here 'Sim.' means 'Similarity', 'SFT' refers to 'Supervised Fine-Tuning', 'P' refers to 'Precision', 'R' refers to 'Recall', and 'F1' refers to macro F1 scores.

Model	P	R	F1
few shot	0.2529	0.3167	0.2626
Sim. prompt	0.4369	0.4645	0.4400
SFT + few shot	0.6566	0.7161	0.6737
SFT + Sim. prompt	0.8143	0.8746	0.8350

Table 5: Results for **A**.

Model	P	R	F1
few shot	0.4477	0.4087	0.4072
Sim. prompt	0.6528	0.6364	0.6367
SFT + few shot	0.8601	0.8124	0.8329
SFT + Sim. prompt	0.8823	0.8592	0.8694

Table 6: Results for **B**.

Model	P	R	F1
few shot	0.2719	0.2642	0.2492
Sim. prompt	0.5280	0.5483	0.5182
SFT + few shot	0.6776	0.5412	0.5878
SFT + Sim. prompt	0.6874	0.6172	0.6381

Table 7: Results for **C**.

Table 8, table 9 and table 10 are results of the fusion of LLM and SLM model. Here "Sim. Prompt" refers to a fine-tuned LoRA SFT model with similarity-based prompting, 'P' refers to 'Precision', 'R' refers to 'Recall', and 'F1' refers to macro F1 scores.

Model	P	R	F1
SLM	0.8223	0.8646	0.8305
Sim. prompt	0.8143	0.8746	0.8350
Sim. prompt +SLM	0.8841	0.8901	0.8855

Table 8: Results for **A**.

Model	P	R	F1
SLM	0.8844	0.8648	0.8738
Sim. prompt	0.8823	0.8592	0.8694
Sim. prompt +SLM	0.8986	0.8685	0.8824

Table 9: Results for **B**.

Model	P	R	F1
SLM	0.6954	0.7522	0.7207
Sim. prompt	0.6874	0.6172	0.6381
Sim. prompt +SLM	0.7641	0.7904	0.7733

Table 10: Results for **C**.