

Assessment of Generative Named Entity Recognition in the Era of Large Language Models

Qi Zhan, Yile Wang*, Hui Huang

Guangdong Provincial Key Laboratory of Visual Media and Multidimensional Intelligence
CSSE, Shenzhen University
qzhan65@gmail.com, wangyile@szu.edu.cn, hhzhiyan@gmail.com

Abstract

Named entity recognition (NER) is evolving from a sequence labeling task into a generative paradigm with the rise of large language models (LLMs). We conduct a systematic evaluation of open-source LLMs on both flat and nested NER tasks. We investigate several research questions including the performance gap between generative NER and traditional NER models, the impact of output formats, whether LLMs rely on memorization, and the preservation of general capabilities after fine-tuning. Through experiments across eight LLMs of varying scales and four standard NER datasets, we find that: (1) With parameter-efficient fine-tuning and structured formats like inline bracketed or XML, open-source LLMs achieve performance competitive with traditional encoder-based models and surpass decoder-based LLMs with in-context learning techniques; (2) The NER capability of LLMs stems from instruction-following and generative power, not mere memorization of entity-label pairs; and (3) Applying NER instruction tuning has minimal impact on general capabilities of LLMs, even improving performance on datasets like DROP by 25.50 to 45.32 F1 points due to enhanced entity understanding. These findings demonstrate that generative NER with LLMs is a promising, user-friendly alternative to traditional methods. We release the data and code at <https://github.com/szuter/LLMs4NER>.

1 Introduction

Named entity recognition (NER) is a fundamental task in natural language processing, widely used in downstream applications such as relation extraction, knowledge graph construction, information retrieval, and question answering systems. As a typical sequence labeling task, approaches based on traditional auto-encoding pre-trained models

*Corresponding author.

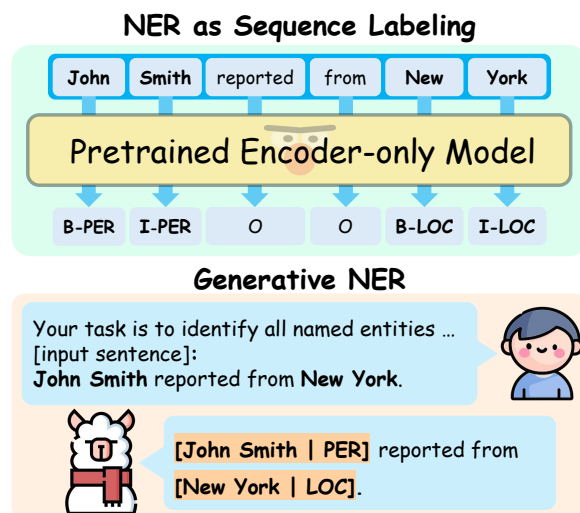


Figure 1: Comparison between traditional NER based on pre-trained models (top) and generative NER based on large language models (bottom).

have achieved strong results (Devlin et al., 2019; Li et al., 2020a; Lu et al., 2022; Yan et al., 2023).

The development of generative pre-trained models (Radford et al., 2019; Lewis et al., 2020) and large language models (LLMs; Brown et al., 2020; OpenAI, 2022, 2023; DeepSeek-AI, 2024) is gradually shifting the traditional NER task toward a unified generative paradigm. As shown in Figure 1, LLMs sequentially generate user-provided sentences along with their corresponding entity labels in textual form. This approach demonstrates strong interactivity, generalization, and user-friendliness. Recent studies have conducted preliminary investigations into NER based on LLMs by crafting specific prompts (Xie et al., 2023; Wei et al., 2023; Kim et al., 2024; Wang et al., 2025a), also achieving promising results.

In this work, we aim to further explore the following research questions: **In terms of performance, is generative NER with open-source LLMs a reliable approach compared with advanced pre-trained NER models?** Most of

the aforementioned studies test the performance of closed-source models like ChatGPT (OpenAI, 2022) for zero-shot NER, and there is currently a lack of systematic investigation into the effectiveness of fine-tuned open-source LLMs, such as advanced LLaMA (Grattafiori et al., 2024), Qwen (Yang et al., 2025), Granite (IBM Research, 2025) and Gemma (Gemma Team, 2025) models. Specifically, we seek to examine whether generative LLMs represent a reliable approach and whether there is a performance gap compared to traditional pre-trained model methods on both flat and nested NER tasks.

How do different output formats affect the performance of LLMs in generative NER? The performance of LLMs is highly sensitive to the specific instructions given by users. While recent studies have touched upon format designs (Kim et al., 2024; Lv et al., 2025), we systematically explore a more diverse set of output templates as a supplement to the aforementioned work and investigate their specific impact on NER performance. Specifically, we design five distinct output formats (Inline Bracketed, Inline XML, Category-grouped JSON, Occurrence-based JSON, and Offset-based JSON) to compare their impact and investigate whether the performance of different models remains consistent across these various formats.

Do LLMs rely on memorization of entity labels to solve NER tasks? Named entities are prevalent in corpora and are incorporated into the training process of LLMs. Moreover, studies indicate that portions of certain datasets have been leaked during LLM training, casting doubt on the reliability of evaluation results (Sainz et al., 2023; Dong et al., 2024; Balloccu et al., 2024). Therefore, we aim to investigate whether the generative NER capability of LLMs stems merely from memorizing entity-label mappings, rather than from genuinely learning the features of entities and accurately identifying and labeling them. Specifically, we evaluate LLMs under few-shot prompting without fine-tuning and replace standard text labels with arbitrary symbols to test whether models rely on memorized entity-label correlations.

Do LLMs preserve the original capabilities after instruction tuning for NER? One advantage of LLMs is their strong generalization capability, which enables them to perform well across a wide range of tasks. We investigate whether the original abilities of LLMs are affected after instruction tuning on NER tasks. To

this end, we compare the performance of the LLM before and after fine-tuning on multiple datasets, including HellaSwag (Zellers et al., 2019), DROP (Dua et al., 2019), GSM8K (Cobbe et al., 2021), MMLU (Hendrycks et al., 2021), and TruthfulQA (Lin et al., 2022).

In this study, we conduct experiments using eight LLMs with varying sizes and versions of LLaMA, Qwen, Granite, and Gemma across four standard datasets, including two flat NER datasets CoNLL2003 (Tjong Kim Sang and De Meulder, 2003) and OntoNotes5.0 (Pradhan et al., 2013), and two nested NER datasets ACE2005 (Walker et al., 2006) and GENIA (Kim et al., 2003). Empirically, the answers to the above questions are as follows:

1) With parameter-efficient fine-tuning and structured output formats like inline bracketed or inline XML, LLMs match the performance of traditional encoder-based NER models and exceed that of closed-source LLMs like GPT-3. They excel particularly in flat NER within general domains, though further progress is needed for nested NER in domains like biomedicine.

2) The strong performance of LLMs in generative NER tasks stems primarily from their powerful instruction-following and text generation capabilities, rather than from merely memorizing entity-label correlations from training corpora. Error analysis indicates that LLMs tend to generate wrong types, which stems from a misalignment between the rich intrinsic knowledge and the detailed annotation guidelines of datasets.

3) LLMs can retain most of their original capabilities under parameter-efficient fine-tuning for NER tasks, with performance on tasks staying stable or slightly decreasing while improving significantly on benchmarks like DROP (Dua et al., 2019) by 25.50 to 45.32 F1 points. Considering their user-friendly natural language interface, which lowers the barrier for non-experts, generative NER systems hold the potential to gradually supplant traditional sequence labeling based NER models.

2 Related Work

NER with Traditional and Pre-trained Models. Traditional NER methods utilize CNN or LSTM approaches for NER (Ma and Hovy, 2016; Yang et al., 2018), where no prior knowledge is used before model training. Subsequently, pre-trained models represented by BERT (Devlin et al., 2019) further enhanced the performance of NER, gradu-

Types	Examples of Labeled Output
Inline Bracketed	This is the first demonstration of a specific interaction with [PU.1 PROTEIN] on a [myeloid [PU.1 PROTEIN] binding site DNA].
Inline XML	This is the first demonstration of a specific interaction with < PROTEIN >PU.1</ PROTEIN > on a < DNA >myeloid < PROTEIN >PU.1</ PROTEIN > binding site</ DNA >.
Category-grouped JSON	{ "DNA" : ["myeloid PU.1 binding site"], "PROTEIN" : ["PU.1", "PU.1"], "CELL_LINE" : [], "CELL_TYPE" : [], "RNA" : []}
Occurrence-based JSON	[{"text": "PU.1", "label": "PROTEIN" , "occurrence_index": 1}, {"text": "myeloid PU.1 binding site", "label": "DNA" , "occurrence_index": 1}, {"text": "PU.1", "label": "PROTEIN" , "occurrence_index": 2}]
Offset-based JSON	[{"text": "PU.1", "label": "PROTEIN" , "start": 63, "end": 67}, {"text": "myeloid PU.1 binding site", "label": "DNA" , "start": 73, "end": 98}, {"text": "PU.1", "label": "PROTEIN" , "start": 81, "end": 85}]

Table 1: Five types of output formats and examples for both flat and nested named entities.

ally becoming solid backbone models and reaching state-of-the-art levels (Li et al., 2020a, 2022; Lu et al., 2022; Yan et al., 2023). Overall, these methods treat NER as a token-level sequence labeling task, utilizing auto-encoding models for text modeling. Cui et al. (2021) and Ma et al. (2022) explore the use of generative BART (Lewis et al., 2020) model for NER, gradually transforming NER from sequence labeling into a sequence-to-sequence generation task, and shifting NER models from auto-encoding representation models to auto-regressive generative models. As a result, the generative NER approach has also demonstrated promising results, while offering good flexibility and generalization.

Generative NER with Large Language Models. Recent studies employ advanced LLMs for generative NER. Xie et al. (2023) and Wei et al. (2023) preliminarily explore the performance of ChatGPT on zero-shot NER. Kim et al. (2024) test the performance of ChatGPT and LLaMA on nested NER by designing fixed templates. Zhou et al. (2024) distill knowledge from ChatGPT for NER and Wang et al. (2025a) investigate the effectiveness of GPT-3 in NER through in-context learning (Brown et al., 2020). These works primarily focus on evaluating zero-shot or few-shot performance of LLMs. However, a comprehensive evaluation and comparison of LLMs for NER tasks remains lacking. We systematically compare the performance of fine-tuned LLMs with traditional state-of-the-art NER models, analyze the impact of different output formats, and explore the memorization and learning capabilities of LLMs for NER tasks, thereby addressing this gap and serving as a complement to the aforementioned research efforts.

Datasets	Types	Domain	#Labels	#Train	#Dev	#Test
CoNLL2003	Flat	General	4	14,041	3,250	3,453
OntoNotes5.0	Flat	General	18	59,924	8,528	8,262
ACE2005	Nested	General	7	7,299	971	1,060
GENIA	Nested	Biomedical	5	14,835	1,854	1,854

Table 2: Statistics of four NER datasets we used.

Recent Exploration on NER. The advancement of LLMs has given rise to new paradigms for NER. Huang et al. (2026) employ reinforcement learning to implement NER based on long chain-of-thought reasoning. Wang et al. (2025b) propose a multi-agent paradigm for zero-shot NER. Mu et al. (2026) also propose an agent-style framework for multi-domain low-resource NER. These approaches offer new directions for future NER systems.

3 Instructions, Output Format, and Performance of Generative NER

3.1 Settings

Instructions and Output Format. Prompt design is crucial for LLM fine-tuning. It mainly includes the design of instructions that specify the tasks the model needs to complete, and the output format used for answer generation, extraction, and validation. The complete instructions are provided in Appendix B for reproducibility.

As listed in Table 1, we compare five different output formats. **Inline Bracketed** and **Inline XML** represent an embedded paradigm where entity boundaries and types are inserted directly into the original sentence using brackets or XML tags, yielding natural-language-like outputs that preserve the original text structure. The JSON-based formats treat NER as a structured information extrac-

tion task. **Category-grouped JSON** organizes entities by their label keys, which can only handle the entity-label relationship while ambiguity exists due to the lack of positional information. **Occurrence-based JSON** records an occurrence order index for each entity, providing a certain level of positional reference, while **Offset-based JSON** demands precise character-level start and end positions, whose fine-grained positional alignment poses a greater challenge for generative models.

Datasets. For in-domain evaluation, to fairly compare with pre-trained model based methods, we employ standard datasets and do not use external instruction data such as in UniversalNER (Zhou et al., 2024). For traditional flat NER, we use CoNLL2003 (Tjong Kim Sang and De Meulder, 2003) and OntoNotes5.0 (Pradhan et al., 2013) in general domain. For more challenging nested NER, we use ACE2005¹ (Walker et al., 2006) in general domain and GENIA (Kim et al., 2003) in molecular biology domain. The statistics of datasets are shown in Table 2.

Baselines. For pre-trained NER models, we compare BERT-Tagger (Devlin et al., 2019), GNN-SL (Wang et al., 2023), ACE + Fine-tune (Wang et al., 2021), BERT-MRC+DSC (Li et al., 2020b), BERT-MRC (Li et al., 2020a), BINDER (Zhang et al., 2023), W2NER (Li et al., 2022), and Bi-affine+CNN (Yan et al., 2023). See Appendix C for brief introductions of these methods.

We also compare GPT-NER (Wang et al., 2025a), which uses closed-source GPT-3 for generative NER. Three settings are used including zero-shot (ZS), zero-shot with entity verification (ZS w/ Ver.), and few-shot with entity verification (FS w/ Ver.).

Models. We use representative open-source models from four LLM families: LLaMA (Grattafiori et al., 2024), Qwen (Yang et al., 2025), Granite (IBM Research, 2025), and Gemma (Gemma Team, 2025). Specifically, we evaluate versions including LLaMA3.1, LLaMA3.2, Qwen2.5, Qwen3, Granite3.3, Granite4, Gemma2, and Gemma3, covering lightweight (3B~4B) and medium-sized (7B~9B) parameter scales.

Implementation Details. We fine-tuned all models using the LLaMA-Factory (Zheng et al., 2024) with LoRA (Hu et al., 2022). We compare LoRA and full fine-tuning in Appendix D, showing that LoRA achieves competitive performance with fewer parameters. We set the LoRA rank r to 256 and the

Models	CoNLL2003	OntoNotes5.0	Average
<i>Pre-Trained NER Models</i>			
BERT-Tagger (Devlin et al., 2019)	91.70	89.16	90.43
GNN-SL (Wang et al., 2023)	93.20	91.39	92.30
ACE + Fine-tune (Wang et al., 2021)	93.64	<u>91.39</u>	92.52
BERT-MRC+DSC (Li et al., 2020b)	93.33	92.07	92.70
BERT-MRC (Li et al., 2020a)	93.04	91.11	92.08
<i>GPT-NER (Wang et al., 2025a)</i>			
🌀 GPT-3 (ZS)	89.97	81.73	85.85
🌀 GPT-3 (ZS w/ Ver.)	90.62	82.08	86.35
🌀 GPT-3 (FS w/ Ver.)	90.91	82.20	86.56
<i>Open-Source LLMs (Inline Bracketed)</i>			
🌀 LLaMA3.1-8B	93.85	91.28	92.57
🌀 LLaMA3.2-3B	93.54	89.19	91.37
🌀 Qwen2.5-7B	93.73	90.97	92.35
🌀 Qwen3-4B	93.37	89.30	91.34
🌀 Granite3.3-8B	93.76	90.31	92.04
🌀 Granite4-3B	93.15	89.82	91.49
🌀 Gemma2-9B	93.64	90.91	92.28
🌀 Gemma3-4B	93.68	90.09	91.89
Average	93.59	90.23	91.91
<i>Open-Source LLMs (Inline XML)</i>			
🌀 LLaMA3.1-8B	<u>93.83</u>	90.94	92.39
🌀 LLaMA3.2-3B	93.06	89.30	91.18
🌀 Qwen2.5-7B	93.59	90.79	92.19
🌀 Qwen3-4B	93.15	88.96	91.06
🌀 Granite3.3-8B	93.33	89.95	91.64
🌀 Granite4-3B	93.07	89.46	91.27
🌀 Gemma2-9B	93.59	90.99	92.29
🌀 Gemma3-4B	93.22	90.15	91.69
Average	93.36	90.07	91.71
<i>Open-Source LLMs (Category-grouped JSON)</i>			
🌀 LLaMA3.1-8B	92.56	89.84	91.20
🌀 LLaMA3.2-3B	92.41	87.34	89.88
🌀 Qwen2.5-7B	92.74	89.48	91.11
🌀 Qwen3-4B	92.47	87.67	90.07
🌀 Granite3.3-8B	92.97	88.98	90.98
🌀 Granite4-3B	91.78	88.44	90.11
🌀 Gemma2-9B	93.48	90.54	92.01
🌀 Gemma3-4B	92.53	89.17	90.85
Average	92.62	88.93	90.78
<i>Open-Source LLMs (Occurrence-based JSON)</i>			
🌀 LLaMA3.1-8B	93.54	90.52	92.03
🌀 LLaMA3.2-3B	93.58	87.64	90.61
🌀 Qwen2.5-7B	93.52	90.04	91.78
🌀 Qwen3-4B	92.91	87.98	90.45
🌀 Granite3.3-8B	92.94	89.20	91.07
🌀 Granite4-3B	92.76	88.63	90.70
🌀 Gemma2-9B	93.55	90.29	91.92
🌀 Gemma3-4B	93.17	88.49	90.83
Average	93.25	89.10	91.17
<i>Open-Source LLMs (Offset-based JSON)</i>			
🌀 LLaMA3.1-8B	63.22	53.31	58.27
🌀 LLaMA3.2-3B	56.13	31.20	43.67
🌀 Qwen2.5-7B	60.84	50.92	55.88
🌀 Qwen3-4B	61.18	38.34	49.76
🌀 Granite3.3-8B	64.79	44.08	54.44
🌀 Granite4-3B	61.40	38.91	50.16
🌀 Gemma2-9B	65.53	61.45	63.49
🌀 Gemma3-4B	59.85	51.35	55.60
Average	61.62	46.20	53.91

Table 3: Results of pre-trained models, GPT-3, and open-source LLMs on flat NER datasets. The best results are shown in **bold** and the second-best results are shown in underlined. Averages are highlighted in **blue**.

scaling factor α to 512, with adapters inserted into all linear modules. The models were trained with a total batch size of 8 and a learning rate of $2e-5$, utilizing a cosine scheduler and a warmup ratio of 0.01. The maximum input length was set to 2048, and all models were trained for two epochs.

¹<https://catalog.ldc.upenn.edu/LDC2006T06>

3.2 Results

The results for flat and nested NER are shown in Table 3 and Table 4, respectively. We report single-run Micro-F1 scores in the main text and provide the complete Precision, Recall, and F1 results in Appendix H. The main findings are as follows:

The output format has a decisive impact on the performance of generative NER. Among the five output formats, the Inline Bracketed and Inline XML formats achieve the best average performance across all LLMs, with average F1 scores of 91.91/82.85 and 91.71/82.67 on flat and nested NER. Both inline formats consistently outperform the JSON-based formats (90.78/81.69, 91.17/80.07 and 53.91/34.53). These results indicate that clear, structured, natural-language-like inline outputs are more conducive to helping LLMs understand and generate accurate entity information, whereas formats requiring concluding the whole information all at once and precise character-offset calculations are poorly suited to the text-generation nature of LLMs, leading to a sharp decline in performance.

Well-designed output formats enable LLMs to match or surpass some traditional fine-tuning methods and closed-source models. With the Inline Bracketed and XML format, LLaMA3.1-8B achieves 93.85 and 93.83 on CoNLL2003, surpassing all pre-trained NER baselines. On ACE2005, Gemma2-9B achieves 88.02, surpassing BERT-MRC (86.88) and Biaffine+CNN (87.25). Both inline formats also substantially outperform in-context learning by closed-source GPT-3 models. This demonstrates that with properly constructed prompts, generative NER with open-source LLMs can achieve highly competitive performance, leveraging powerful native language understanding capabilities. Moreover, as shown in Section 6, after instruction tuning for NER, the model’s general capability is preserved, with performance on some tasks remaining unaffected or even improved.

Compared with the general domain, LLMs lag behind traditional specialized models in low-resource domains. Although LLMs achieve competitive results on general-domain datasets, their performance is less competitive on GENIA in the biomedical domain. The best LLM result is 4.02 F1 points lower than the best specialized model BERT-MRC and falls below other traditional baselines. This indicates that due to a lack of prior knowledge in low-resource domains, LLMs’ capabilities there are significantly weaker than in the general

Models	ACE2005	GENIA	Average
<i>Pre-Trained NER Models</i>			
BERT-MRC (Li et al., 2020a)	86.88	83.75	85.32
BINDER (Zhang et al., 2023)	90.00	80.80	85.40
W2NER (Li et al., 2022)	86.79	80.32	83.56
Biaffine+CNN (Yan et al., 2023)	87.25	80.33	83.79
<i>GPT-NER (Wang et al., 2025a)</i>			
🌀 GPT-3 (ZS)	72.96	64.06	68.51
🌀 GPT-3 (ZS w/ Ver.)	73.46	64.29	68.88
🌀 GPT-3 (FS w/ Ver.)	73.59	64.42	69.01
<i>Open-Source LLMs (Inline Bracketed)</i>			
🌀 LLaMA3.1-8B	87.51	79.18	83.35
🌀 LLaMA3.2-3B	85.82	79.15	82.49
🌀 Qwen2.5-7B	86.70	78.67	82.69
🌀 Qwen3-4B	86.48	79.16	82.82
🌀 Granite3.3-8B	86.81	78.87	82.84
🌀 Granite4-3B	86.78	78.42	82.60
🌀 Gemma2-9B	87.44	79.73	83.59
🌀 Gemma3-4B	86.20	78.64	82.42
Average	86.72	78.98	82.85
<i>Open-Source LLMs (Inline XML)</i>			
🌀 LLaMA3.1-8B	86.13	78.41	82.27
🌀 LLaMA3.2-3B	86.26	78.22	82.24
🌀 Qwen2.5-7B	85.97	79.27	82.62
🌀 Qwen3-4B	85.66	79.60	82.63
🌀 Granite3.3-8B	87.27	79.09	83.18
🌀 Granite4-3B	86.84	78.61	82.73
🌀 Gemma2-9B	<u>88.02</u>	79.32	83.67
🌀 Gemma3-4B	85.62	78.46	82.04
Average	86.47	78.87	82.67
<i>Open-Source LLMs (Category-grouped JSON)</i>			
🌀 LLaMA3.1-8B	86.13	78.29	82.21
🌀 LLaMA3.2-3B	83.69	77.99	80.84
🌀 Qwen2.5-7B	84.84	78.15	81.50
🌀 Qwen3-4B	84.35	78.48	81.42
🌀 Granite3.3-8B	86.23	78.47	82.35
🌀 Granite4-3B	84.77	78.38	81.58
🌀 Gemma2-9B	86.64	78.93	82.79
🌀 Gemma3-4B	84.51	77.18	80.85
Average	85.14	78.23	81.69
<i>Open-Source LLMs (Occurrence-based JSON)</i>			
🌀 LLaMA3.1-8B	83.41	78.66	81.04
🌀 LLaMA3.2-3B	80.93	77.41	79.17
🌀 Qwen2.5-7B	81.69	77.58	79.64
🌀 Qwen3-4B	81.58	78.47	80.03
🌀 Granite3.3-8B	83.02	78.27	80.65
🌀 Granite4-3B	81.88	77.25	79.57
🌀 Gemma2-9B	84.29	79.12	81.71
🌀 Gemma3-4B	81.58	76.03	78.81
Average	82.30	77.85	80.07
<i>Open-Source LLMs (Offset-based JSON)</i>			
🌀 LLaMA3.1-8B	35.85	34.62	35.24
🌀 LLaMA3.2-3B	27.31	25.79	26.55
🌀 Qwen2.5-7B	32.59	32.02	32.31
🌀 Qwen3-4B	35.41	36.81	36.11
🌀 Granite3.3-8B	36.90	39.54	38.22
🌀 Granite4-3B	33.30	36.80	35.05
🌀 Gemma2-9B	40.80	41.15	40.98
🌀 Gemma3-4B	31.60	32.00	31.80
Average	34.22	34.84	34.53

Table 4: Results of pre-trained models, GPT-3, and open-source LLMs on nested NER datasets.

domains. This highlights the generalization limitations of LLMs in low-resource domains, where their performance is inferior to that in general domains. This gap is again evidenced in NER tasks. Despite LLMs being trained on the same data, their deficiency in low-resource domain knowledge prevents them from surpassing traditional models that have undergone targeted training.

<i>Instruction I. Symbol-based Label with Explanations (CoNLL2003-SE)</i>	
Instruction: Your task is to do sequence labeling with labels A, B, C, and D.	
A: A collective entity such as a company, institution, brand, political or governmental body...	(Explanation of ORG)
B: A named individual, including humans, animals, fictional characters, and their aliases.	(Explanation of PER)
C: A geographical or spatial entity, including natural features, built structures, regions...	(Explanation of LOC)
D: Named entities that are not persons, organizations, or locations, including derived...	(Explanation of MISC)
<i>Instruction II. Symbol-based Label Only (CoNLL2003-SO)</i>	
Instruction: Your task is to do sequence labeling with labels A, B, C, and D.	
<i>Input Sentence and Output with Symbol-based Labels Appended to the Instructions</i>	
Input Sentence: Havel praises Czech native Albright as friend.	
Output: Havel praises <D>Czech</D> native Albright as friend.	

Table 5: Examples of two variants of CoNLL2003 with symbol-based labels. The first contains explanations for each symbolic label (CoNLL2003-SE), while the second does not offer explanations in each sample (CoNLL2003-SO).

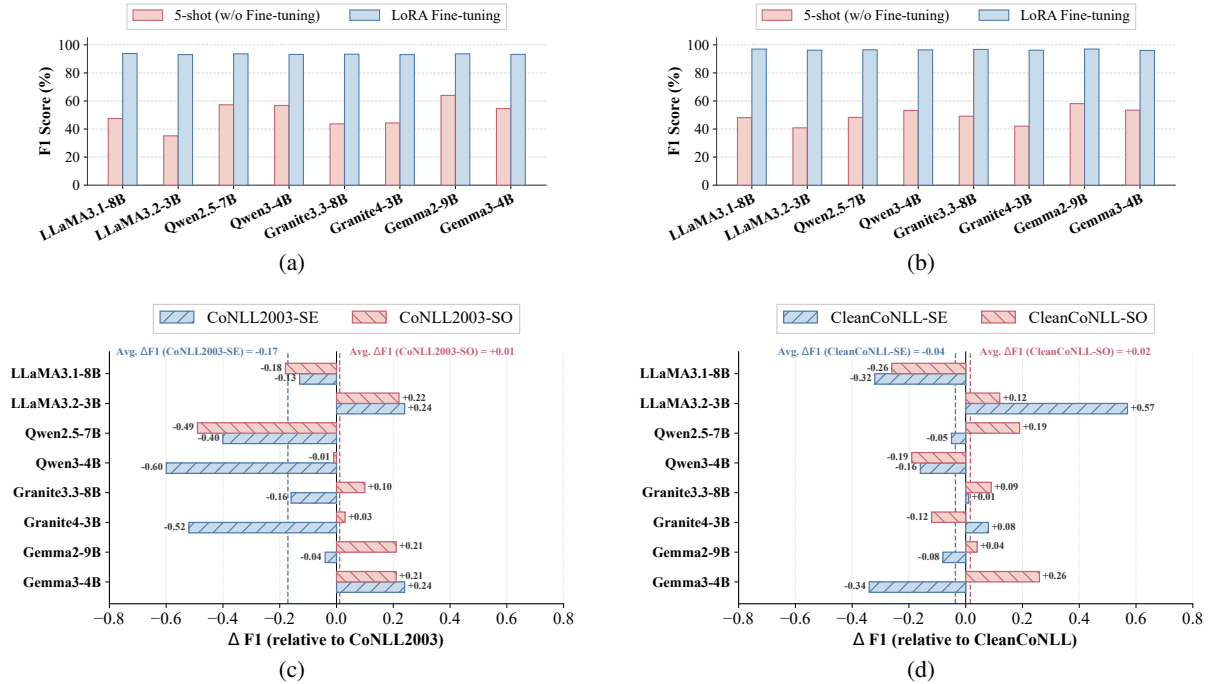


Figure 2: Investigation of entity-label memorization. (a) and (b) compare few-shot prompting with LoRA SFT on CoNLL2003 (left) and CleanCoNLL (right). (c) and (d) report F1 changes under two variants (SE: with explanations; SO: symbol only) relative to the original dataset.

4 Entity Recognition or Memorization?

LLMs are trained on extensive datasets during the pre-training process. In this section, we investigate whether LLMs genuinely possess entity *recognition* capabilities or just rely on entity-label *memorization* when performing generative NER tasks.

Settings. We conduct two experiments on both CoNLL2003 and CleanCoNLL (Rücker and Akbik, 2023) with the Inline XML format. CleanCoNLL is a nearly noise-free revision of CoNLL2003 with corrected annotation errors. Because it must be reconstructed locally from the original corpus using the released correction scripts, its fully processed

form is unlikely to have appeared in model pre-training data, making it a more reliable dataset for investigating memorization. **Few-shot prompting without fine-tuning:** We evaluate all eight LLMs under a 5-shot in-context learning setting. If LLMs had already memorized entity-label mappings during pre-training, they should achieve competitive performance without fine-tuning. **Symbol-based label replacement:** We replace the ground-truth text labels (e.g., LOC, ORG) with arbitrary symbols (e.g., A, B) in each sample. We further distinguish two settings based on whether label meaning explanations are provided. In the symbolic label with explanations (SE) setting, we pro-

	Error Types	Examples of Incorrectly Labeled Output
Type	<i>OOD Types</i>	<u>[Inter]</u> _{ORG} will be without suspended <u>[French]</u> _{MISC} defender [Joceyln Angloma] _{PLAYER}
	<i>Wrong Types</i>	[Inter] _{LOC} will be without suspended <u>[French]</u> _{MISC} defender <u>[Joceyln Angloma]</u> _{PER}
Boundary	<i>Contain Gold</i>	<u>[Inter]</u> _{ORG} will be without suspended <u>[French]</u> _{MISC} [defender Joceyln Angloma] _{PER}
	<i>Contained by Gold</i>	<u>[Inter]</u> _{ORG} will be without suspended <u>[French]</u> _{MISC} defender Joceyln [Angloma] _{PER}
	<i>Overlap with Gold</i>	<u>[Inter]</u> _{ORG} will be without suspended French [defender Joceyln] _{PER} Angloma
Others	<i>Completely-O</i>	<u>[Inter]</u> _{ORG} will be without [suspended] _{MISC} <u>[French]</u> _{MISC} defender <u>[Joceyln Angloma]</u> _{PER}
	<i>OOD Mentions</i>	<u>[Inter]</u> _{ORG} will be without suspended <u>[French]</u> _{MISC} defender <u>[Joceyln Angloma]</u> _{PER} [Milan] _{ORG}
	<i>Omitted Mentions</i>	<u>[Inter]</u> _{ORG} will be without suspended French defender <u>[Joceyln Angloma]</u> _{PER}

Table 6: Error types based on the sentence: “[Inter]_{ORG} will be without suspended [French]_{MISC} defender [Joceyln Angloma]_{PER}.” Underlined entities denote correct predictions, while **blue bold** spans highlight specific errors.

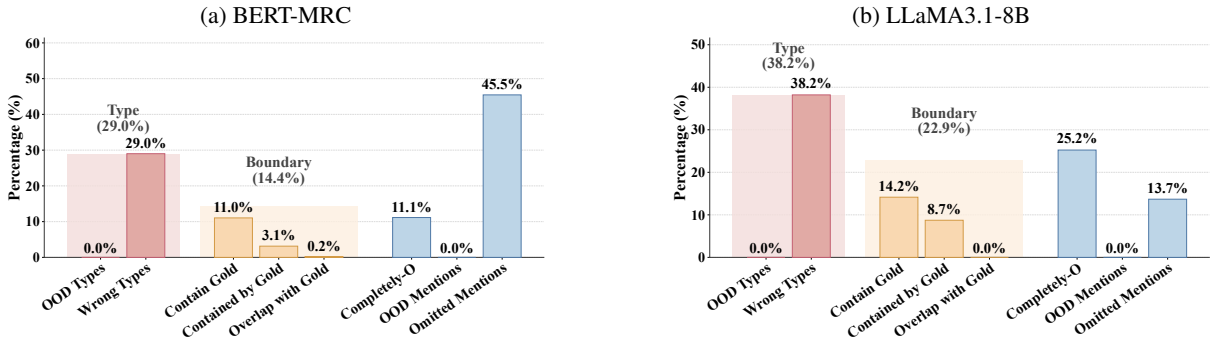


Figure 3: Distribution of error types for BERT-MRC and LLaMA3.1-8B on CoNLL2003.

vide standard entity meaning interpretations for the symbols and denote the transformed datasets as CoNLL2003-SE and CleanCoNLL-SE. In the symbol-only (SO) setting, we omit such interpretations, allowing the model to learn the meaning of each symbol solely from the training examples. The transformed datasets are denoted as CoNLL2003-SO and CleanCoNLL-SO.

Results. As shown in Figure 2(a) and (b), all LLMs achieve moderate F1 scores under 5-shot prompting on both CoNLL2003 and CleanCoNLL, far below fine-tuning performance. This large gap indicates that even if LLMs were exposed to the sentences during pre-training, their generative NER ability remains limited. The NER capability is mainly acquired through instruction tuning.

From the symbol-based label experiment in Figure 2(c) and (d), we find that: 1) LLMs exhibit only marginal F1 changes on both datasets (average $\Delta F1$ of -0.17 on CoNLL2003-SE and $+0.01$ on CoNLL2003-SO; -0.04 on CleanCoNLL-SE and $+0.02$ on CleanCoNLL-SO); 2) The symbol-only setting even yields slight average increases on both datasets. This may be because symbol expla-

nations introduce additional context that confuses the model, whereas pure symbolic labels reduce inference-time ambiguity.

Overall, we reach the same conclusion on both datasets: **LLMs do not rely on entity-label memorization in generative NER tasks, but rather demonstrate a considerable capability in entity learning**, thereby effectively accomplishing entity recognition tasks.

5 Error Analysis on Generative NER

To understand what types of errors LLMs are prone to make in NER and their specific gaps compared to traditional models, we conduct a statistical analysis of error samples. We follow Xie et al. (2023) and categorize errors into eight types, as shown in Table 6. Detailed definitions are in Appendix I.

Statistical Results. We present the error distribution statistics for the representative pre-trained model BERT-MRC, and the fine-tuned LLaMA3.1-8B with Inline Bracketed format in Figure 3.

Overall, we observe clear differences in error patterns between the two paradigms. BERT-MRC’s errors are dominated by *Omitted Mentions* (45.5%),

Error Types	Input Sentence	Ground Truth	BERT-MRC	LLaMA3.1-8B
<i>Omitted Mentions</i>	Italy recalled <u>Marcello Cuttitta</u> .	[M. Cuttitta] _{PER}	O (Missed) (45.5%)	[M. Cuttitta] _{PER} (13.7%)
<i>Wrong Types</i>	In congruence with futures, <u>Permian</u> cash prices for the weekend fell more...	[Permian] _{MISC}	[Permian] _{MISC} (29.0%)	[Permian] _{LOC} (38.2%)
<i>Completely-O</i>	I’ve also got a contract to play for New South Wales in the <u>Super 12</u> next year.	O	O (11.1%)	[Super 12] _{MISC} (25.2%)

Table 7: Case study of errors from BERT-MRC and LLaMA3.1-8B. Text in blue indicates incorrect predictions. The number in parentheses indicates the share of each error type among all errors.

Models	TruthfulQA MC1	MMLU Acc	GSM8K EM	HellaSwag Acc_norm	DROP F1	Avg.
LLaMA3.1-8B	40.15	68.76	84.69	72.53	33.20	59.87
w/ NER LoRA Fine-tuning (XML)	36.96 (↓3.19)	67.08 (↓1.68)	80.36 (↓4.33)	70.66 (↓1.87)	58.70 (↑25.50)	62.75 (↑2.88)
Qwen2.5-7B	47.49	73.42	76.57	65.43	15.74	55.73
w/ NER LoRA Fine-tuning (XML)	45.04 (↓2.45)	72.75 (↓0.67)	79.61 (↑3.04)	68.47 (↑3.04)	61.06 (↑45.32)	65.39 (↑9.66)

Table 8: Performance comparison of LLMs on benchmarks before and after generative NER fine-tuning.

indicating a conservative prediction strategy that tends to miss entities. In contrast, LLaMA3.1-8B exhibits a substantially lower omission rate, reflecting a more aggressive extraction behavior enabled by its extensive pre-training knowledge. However, this higher sensitivity to potential entities leads to a trade-off: the model is prone to over-extraction, resulting in a significantly higher rate of *Completely-O* errors than BERT-MRC.

The primary error for LLMs is *Wrong Types*, comprising 38.2% of all errors. This shows that while generative models correctly identify entity boundaries, they often misclassify types. This likely arises from a conflict between the model’s strong pre-trained prior knowledge and the specific dataset annotation guidelines. Notably, both models have 0% in *OOD Types* and *OOD Mentions*, confirming their adherence to the predefined label set and that all predictions are grounded in the input text.

Case Study. To intuitively understand the statistical differences discussed above, we provide typical error examples in Table 7.

The first example illustrates a common omission error made by traditional NER models. BERT-MRC omits “*Marcello Cuttitta*” due to limited pre-training exposure, whereas the LLM correctly extracts it, leveraging extensive world knowledge to recognize unseen entities.

The second example highlights a typical type misclassification made by LLMs. In the phrase “*Permian cash price*”, the gold standard labels

“*Permian*” as MISC, following the CoNLL2003 guideline that adjectival or derived forms from named entities (e.g., locations, organizations, or persons) should be categorized as miscellaneous. LLaMA3.1-8B predicts “*Permian*” as LOC, which can be attributed to prior knowledge during large-scale pre-training that associates “*Permian*” with a geographical region, instead of following the CoNLL2003 annotation guidelines.

The third example demonstrates an over-extraction behavior of LLMs. In this sentence, the gold annotation does not label “*Super 12*”. However, LLaMA3.1-8B incorrectly labels it as MISC. According to the CoNLL2003 annotation guidelines, the MISC category encompasses a broad and diverse range of entities. This lack of precise semantic boundaries creates ambiguity, making LLMs prone to over-predicting entities.

6 Preservation of General Capability

While traditional pre-trained models become single-purpose specialists after task-specific fine-tuning, LLMs retain their inherent multi-task capabilities as general generative models. To examine the impact of generative NER fine-tuning on these general capabilities, we conduct an evaluation across a range of tasks, including reading comprehension (DROP; Dua et al., 2019), commonsense reasoning (HellaSwag; Zellers et al., 2019), multitask language understanding (MMLU; Hendrycks et al., 2021), mathematical reasoning (GSM8K; Cobbe et al., 2021), and truthfulness (TruthfulQA; Lin

et al., 2022). We use the lm-evaluation-harness framework (Gao et al., 2024) for comparison.

The results are shown in Table 8. We find that the fine-tuned LLMs performed comparably to the original model across most of the tasks, with only minor fluctuations ranging from a 4.33% decrease to a 3.04% increase. Notably, on the reading comprehension benchmark DROP, the fine-tuned model **achieves a substantial performance gain of 25.50 to 45.32 F1 scores**. The improvement could be attributed to the fact that DROP answers are often text spans containing named entities (e.g., persons, locations, organizations). Fine-tuning for NER strengthens the model’s ability to detect these entities, leading to more precise answer extraction. Overall, the results suggest that the core capabilities of LLMs remain largely intact after undergoing lightweight fine-tuning for the NER task.

7 Conclusion

We present a comprehensive assessment of generative NER using open-source LLMs. Our experiments show that LLMs, when fine-tuned with parameter-efficient methods and appropriate output formats, can match or even exceed the performance of strong traditional NER models, while maintaining robust multi-task capabilities. Furthermore, we find that LLMs keep genuine learning on NER task instead of memorizing entity-label correlations. These findings confirm the viability of generative NER as a flexible and effective paradigm, highlighting its potential to gradually supplement traditional sequence labeling approaches.

Limitations

This study primarily focuses on evaluating the NER capabilities of LLaMA, Qwen, Granite, and Gemma model families under instruction tuning. Future improvements could be pursued along two dimensions: First, at the model level, the investigation could be extended to reasoning models with long chain-of-thought output, which offer the advantage of tracing entity labels through the reasoning chain, thereby enhancing interpretability. The current work emphasizes instruction compliance more than the model’s thinking and reasoning abilities. Second, regarding training methodology, reinforcement learning-based approaches such as PPO, DPO, and GRPO have been shown to surpass traditional supervised fine-tuning in terms of eliciting model capability and generalization, yet

were not explored in this work. These limitations represent key directions for our future research.

Acknowledgments

This work was supported in part by ICFCRT (W2441020), Guangdong Basic and Applied Basic Research Foundation (2026A1515011358), Youth S&T Talent Support Programme of Guangdong Provincial Association for Science and Technology (SKXRC2026862), Shenzhen Natural Science Foundation (JCYJ20250604181610014), Intelligent Computing Center of Shenzhen University, and Scientific Development Fund from Guangdong Provincial Key Laboratory of Visual Media and Multidimensional Intelligence.

References

- Simone Balloccu, Patrícia Schmidová, Mateusz Lango, and Ondrej Dusek. 2024. [Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source LLMs](#). In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 67–93, St. Julian’s, Malta. Association for Computational Linguistics.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. [Language models are few-shot learners](#). In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. [Training verifiers to solve math word problems](#). *arXiv preprint arXiv:2110.14168*.
- Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. 2021. [Template-based named entity recognition using BART](#). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1835–1845, Online. Association for Computational Linguistics.
- DeepSeek-AI. 2024. [Deepseek-V3 technical report](#). *arXiv preprint arXiv:2412.19437*.
- 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.
- Yihong Dong, Xue Jiang, Huanyu Liu, Zhi Jin, Bin Gu, Mengfei Yang, and Ge Li. 2024. [Generalization or memorization: Data contamination and trustworthy evaluation for large language models](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12039–12050, Bangkok, Thailand. Association for Computational Linguistics.
- Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. [DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs](#). 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 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.
- Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac’h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. [The language model evaluation harness](#).
- Gemma Team. 2025. [Gemma 3 technical report](#). *arXiv preprint arXiv:2503.19786*.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. [The llama 3 herd of models](#). *arXiv preprint arXiv:2407.21783*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). In *International Conference on Learning Representations*.
- Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *International Conference on Learning Representations*.
- Hui Huang, Yanping Chen, Ruizhang Huang, Chuan Lin, and Yongbin Qin. 2026. [A reasoning paradigm for named entity recognition](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(37):31140–31148.
- IBM Research. 2025. Granite 4.0 language models. <https://github.com/ibm-granite/granite-4.0-language-models>. Accessed: 2025-10-01.
- Hongjin Kim, Jai-Eun Kim, and Harksoo Kim. 2024. [Exploring nested named entity recognition with large language models: Methods, challenges, and insights](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8653–8670, Miami, Florida, USA. Association for Computational Linguistics.
- J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii. 2003. [Genia corpus—a semantically annotated corpus for biotextmining](#). *Bioinformatics*, 19:i180–i182.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. [BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.
- Jingye Li, Hao Fei, Jiang Liu, Shengqiong Wu, Meishan Zhang, Chong Teng, Donghong Ji, and Fei Li. 2022. [Unified named entity recognition as word-word relation classification](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10965–10973.
- Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020a. [A unified MRC framework for named entity recognition](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859, Online. Association for Computational Linguistics.
- Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2020b. [Dice loss for data-imbalanced NLP tasks](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 465–476, Online. Association for Computational Linguistics.
- Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. [TruthfulQA: Measuring how models mimic human falsehoods](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.
- 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.
- Tengxiao Lv, Ling Luo, Juntao Li, Yanhua Wang, Yuchen Pan, Chao Liu, Yanan Wang, Yan Jiang, Huiyi Lv, Yuanyuan Sun, et al. 2025. [A unified biomedical named entity recognition framework with large language models](#). *arXiv preprint arXiv:2510.08902*.

- Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Linyang Li, Qi Zhang, and Xuanjing Huang. 2022. [Template-free prompt tuning for few-shot NER](#). In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5721–5732, Seattle, United States. Association for Computational Linguistics.
- Xuezhe Ma and Eduard Hovy. 2016. [End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.
- Wenxuan Mu, Jinzhong Ning, Di Zhao, and Yijia Zhang. 2026. [A multi-agent llm framework for multi-domain low-resource in-context ner via knowledge retrieval, disambiguation and reflective analysis](#). *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(38):32528–32536.
- OpenAI. 2022. [GPT-3.5 turbo model](#).
- OpenAI. 2023. [GPT-4 technical report](#). *arXiv preprint arXiv:2303.08774*.
- Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Hwee Tou Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. [Towards robust linguistic analysis using OntoNotes](#). In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152, Sofia, Bulgaria. Association for Computational Linguistics.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. [Language models are unsupervised multitask learners](#). *OpenAI blog*, 1(8):9.
- Susanna Rücker and Alan Akbik. 2023. [CleanCoNLL: A nearly noise-free named entity recognition dataset](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8628–8645, Singapore. Association for Computational Linguistics.
- Oscar Sainz, Jon Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. [NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10776–10787, Singapore. Association for Computational Linguistics.
- Erik F. Tjong Kim Sang and Fien De Meulder. 2003. [Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition](#). In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.
- Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. [ACE 2005 multilingual training corpus](#). *Linguistic Data Consortium, Philadelphia*.
- Shuhe Wang, Yuxian Meng, Rongbin Ouyang, Jiwei Li, Tianwei Zhang, Lingjuan Lyu, and Guoyin Wang. 2023. [GNN-SL: Sequence labeling based on nearest examples via GNN](#). In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12679–12692, Toronto, Canada. Association for Computational Linguistics.
- Shuhe Wang, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li, Guoyin Wang, and Chen Guo. 2025a. [GPT-NER: Named entity recognition via large language models](#). In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4257–4275, Albuquerque, New Mexico. Association for Computational Linguistics.
- Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2021. [Automated concatenation of embeddings for structured prediction](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2643–2660, Online. Association for Computational Linguistics.
- Zihan Wang, Ziqi Zhao, Yougang Lyu, Zhumin Chen, Maarten de Rijke, and Zhaochun Ren. 2025b. [A co-operative multi-agent framework for zero-shot named entity recognition](#). In *Proceedings of the ACM on Web Conference 2025*, WWW ’25, page 4183–4195, New York, NY, USA. Association for Computing Machinery.
- Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, et al. 2023. [Zero-shot information extraction via chatting with chatgpt](#). *arXiv preprint arXiv:2302.10205*.
- Tingyu Xie, Qi Li, Jian Zhang, Yan Zhang, Zuozhu Liu, and Hongwei Wang. 2023. [Empirical study of zero-shot NER with ChatGPT](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7935–7956, Singapore. Association for Computational Linguistics.
- Hang Yan, Yu Sun, Xiaonan Li, and Xipeng Qiu. 2023. [An embarrassingly easy but strong baseline for nested named entity recognition](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1442–1452, Toronto, Canada. Association for Computational Linguistics.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. [Qwen3 technical report](#). *arXiv preprint arXiv:2505.09388*.
- Jie Yang, Shuailong Liang, and Yue Zhang. 2018. [Design challenges and misconceptions in neural sequence labeling](#). In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3879–3889, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. [HellaSwag: Can a machine really finish your sentence?](#) In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Sheng Zhang, Hao Cheng, Jianfeng Gao, and Hoifung Poon. 2023. [Optimizing bi-encoder for named entity recognition via contrastive learning](#). In *The Eleventh International Conference on Learning Representations*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyang Luo. 2024. [LlamaFactory: Unified efficient fine-tuning of 100+ language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, Bangkok, Thailand. Association for Computational Linguistics.

Wenxuan Zhou, Sheng Zhang, Yu Gu, Muhao Chen, and Hoifung Poon. 2024. [UniversalNER: Targeted distillation from large language models for open named entity recognition](#). In *The Twelfth International Conference on Learning Representations*.

A Checkpoints of Models

Table 9 lists the checkpoints of the open-source large language models used in our experiments.

Model	Resource Link
	 LLaMA Family
LLaMA3.1-8B	meta-llama/Llama-3.1-8B-Instruct
LLaMA3.2-3B	meta-llama/Llama-3.2-3B-Instruct
	 Qwen Family
Qwen2.5-7B	Qwen/Qwen2.5-7B-Instruct
Qwen3-4B	Qwen/Qwen3-4B-Instruct-2507
	 Granite Family
Granite3.3-8B	ibm-granite/granite-3.3-8b-instruct
Granite4-3B	ibm-granite/granite-4.0-micro
	 Gemma Family
Gemma2-9B	google/gemma-2-9b-it
Gemma3-4B	google/gemma-3-4b-it

Table 9: Checkpoints of open-source models used in our experiments.

B Full Instructions for Different Datasets

We provide the complete instruction design used for LLMs on generative NER tasks below.

B.1 Instruction Construction Framework

Each instruction follows a unified structural organization. As illustrated in Figure 4, the constructed input is composed of three main components: **Task Description**, which defines the extraction goal and

Instruction Framework

[Task Description]

Your task is to identify all named entities in the input sentence and rewrite the sentence by enclosing each entity using the format `<LABEL>Entity Text</LABEL>`. Use only the label tags defined in the Label Set below.

[Label Definition Schema]

Label Set:

ORG(organization): A collective entity such as a company, institution, brand, political or governmental body, publication, or any organized group of people acting as a unit.

PER(person): A named individual, including humans, animals, fictional characters, and their aliases.

LOC(location): A geographical or spatial entity, including natural features, built structures, regions, public or commercial places, assorted buildings, and abstract or metaphorical places.

MISC(miscellaneous): Named entities that are not persons, organizations, or locations, including derived adjectives, religions, ideologies, nationalities, languages, events, programs, wars, titles of works, slogans, eras, and types of objects.

[Input Sentence]

Now process the input sentence:

Figure 4: The template and example of constructing instructions for generative NER.

output format; **Label Definition Schema**, which provides fine-grained semantic constraints for each entity type; **Input Sentence**, the target text for labeling. These components collectively guide the model to generate the target output according to the specified format.

B.2 Task Descriptions for Different Output Formats

We designed five distinct task descriptions corresponding to the output formats. These descriptions explicitly constrain the generation behavior of the LLMs. Table 10 details the specific prompt templates used for each format.

Output Format	Task Description
Inline Bracketed	Your task is to identify all named entities in the input sentence and rewrite the sentence by enclosing each entity using the format [Entity Text LABEL]. Use only the label tags defined in the Label Set below.
Inline XML	Your task is to identify all named entities in the input sentence and rewrite the sentence by enclosing each entity using the format <LABEL>Entity Text</LABEL>. Use only the label tags defined in the Label Set below.
Category-grouped JSON	Your task is to identify all named entities in the input sentence and extract them into a JSON object grouped by label. The keys of the JSON object should be the label tags defined in the Label Set below, and the values should be lists of extracted entities. If a label has no corresponding entities, use an empty list.
Occurrence-based JSON	Your task is to identify all named entities in the input sentence and extract them into a JSON list. Output a JSON list where each object contains: <ol style="list-style-type: none"> 1. “text”: the entity string. 2. “label”: the label tags defined in the Label Set below. 3. “occurrence_index”: a 1-based integer indicating the order of appearance of this text in the sentence (e.g., 1 for the first appearance, 2 for the second). If no entities are found, return an empty list.
Offset-based JSON	Your task is to identify all named entities in the input sentence and extract them with precise character offsets. Output a JSON list where each object contains: <ol style="list-style-type: none"> 1. “text”: the entity string. 2. “label”: the label tags defined in the Label Set below. 3. “start”: the 0-based index of the entity’s first character. 4. “end”: the 0-based index of the first character after the entity (exclusive).

Table 10: Task descriptions for different output formats.

B.3 Label Definition Schema

To enhance the semantic understanding of the LLM, particularly for ambiguous or domain-specific entities, we provide explicit definitions for each label in the instructions. These definitions are derived from the official annotation guidelines of each dataset, which serve as the authoritative standard in the dataset’s labeling process. For the CoNLL2003 dataset, the official annotation guidelines contain extensive explanations for the labels, which we have summarized using a large language model to provide concise, standard linguistic definitions. For the GENIA dataset, the official annotation guidelines were not available; therefore, we employed a large language model to generate standard biomedical definitions. The detailed definitions for CoNLL2003, OntoNotes5.0, GENIA, and ACE2005 are provided in Table 14, Table 15, Ta-

ble 16, and Table 17, respectively.

C Brief Introduction of Baselines

BERT-Tagger (Devlin et al., 2019) uses a BERT model and trains classification heads for tagging tokens. **GNN-SL** (Wang et al., 2023) augments sequence labeling by retrieving nearest neighbor examples and modeling their interactions with the input via GNNs. **ACE + Fine-tune** (Wang et al., 2021) utilizes a reinforcement learning controller to automatically search for the optimal concatenation of various pre-trained embeddings. For a fair comparison, we adopt its sentence-level variant, achieving SOTA performance on the CoNLL2003 dataset. **BERT-MRC+DSC** (Li et al., 2020b) addresses data imbalance via Dice loss, achieving SOTA performance on the OntoNotes5.0 dataset. **BERT-MRC** (Li et al., 2020a) formulates NER

Model	Fine-tuning	CoNLL2003	GENIA
LLaMA3.1-8B	LoRA	93.85	79.18
LLaMA3.1-8B	Full	93.61 (↓0.24)	78.70 (↓0.48)
Qwen2.5-7B	LoRA	93.73	78.67
Qwen2.5-7B	Full	93.58 (↓0.15)	78.53 (↓0.14)

Table 11: Comparison between LoRA fine-tuning and full fine-tuning on CoNLL2003 and GENIA. Numbers in parentheses denote the difference compared with LoRA fine-tuning.

as a machine reading comprehension task, achieving SOTA performance on the GENIA dataset. **BINDER** (Zhang et al., 2023) aligns spans and types in a shared space via a bi-encoder and contrastive learning, distinguishing entities with dynamic thresholding. **W2NER** (Li et al., 2022) unifies flat, nested, and discontinuous NER by modeling word-word relations on a 2D grid to capture token interactions. **Biaffine+CNN** (Yan et al., 2023) treats the span feature grid generated by a biaffine decoder as an image, utilizing CNNs to model spatial correlations between neighboring spans for nested entity recognition.

D Comparison between LoRA and Full Fine-Tuning

We compare LoRA with full fine-tuning to determine which strategy is more effective for generative NER. The two well-performing models, LLaMA3.1-8B and Qwen2.5-7B, are fine-tuned on the flat NER dataset (CoNLL2003) and the nested NER dataset (GENIA). Both fine-tuning methods use the same training configuration. The F1 results are shown in Table 11.

Across all settings, LoRA fine-tuning obtains slightly higher F1 scores than full fine-tuning. Moreover, because the number of trainable parameters in LoRA is much closer to that of traditional NER models, LoRA also provides a fairer basis for comparing generative and traditional approaches. Therefore, we adopt LoRA as the fine-tuning strategy for all LLMs in our main experiments.

E Breakdown of Flat and Nested Entity Performance

We further break down performance on the two nested NER datasets, ACE2005 and GENIA, according to the structural relation between entity spans. We define a *nested entity* as a gold entity whose span strictly contains or is strictly contained by another gold entity in the same sentence. A *flat*

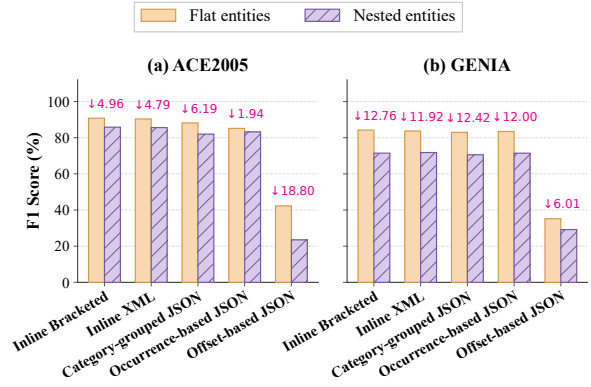


Figure 5: Performance on flat and nested entity mentions in the two nested NER datasets, averaged across eight LLMs for each output format. Downward arrows report the F1 decrease from flat to nested entities.

entity neither contains nor is contained by another gold entity. We compute entity-level Micro-F1 separately for these two disjoint subsets and average the scores across all eight LLMs.

As shown in Figure 5, nested entities consistently achieve lower F1 scores than flat entities across all output formats and on both datasets, indicating that they are more difficult to recognize. Excluding the Offset-based JSON format, whose overall scores are too low for meaningful comparison, the flat-to-nested F1 decrease on GENIA is consistently larger than on ACE2005, suggesting that nested entity recognition is particularly challenging in the biomedical domain.

F Error Analysis of the Offset-based JSON Format

The Offset-based JSON format yields substantially lower F1 scores than the other formats, as reported in Tables 3 and 4. To further investigate the reasons for the low Offset-based JSON performance, we present two complementary analyses below.

Entity recognition vs. offset generation. We re-evaluate predictions under two criteria. Standard F1 requires an exact match of the entity text, label, and character offsets. $F_1^{w/o \text{ offset}}$ matches only the entity text and label. As shown in Table 12, ignoring offsets leads to large improvements across all models and datasets, with performance approaching that of the inline formats. This confirms that the models usually accurately identify entities, while the primary difficulty lies in precise character-level offset alignment.

Model	CoNLL2003		OntoNotes5.0		ACE2005		GENIA	
	F1	$F_1^{w/o \text{ offset}}$	F1	$F_1^{w/o \text{ offset}}$	F1	$F_1^{w/o \text{ offset}}$	F1	$F_1^{w/o \text{ offset}}$
LLaMA3.1-8B	63.22	92.64 ($\uparrow 29.42$)	53.31	87.60 ($\uparrow 34.29$)	35.85	81.60 ($\uparrow 45.75$)	34.62	77.96 ($\uparrow 43.34$)
LLaMA3.2-3B	56.13	91.91 ($\uparrow 35.78$)	31.20	84.71 ($\uparrow 53.51$)	27.31	79.54 ($\uparrow 52.23$)	25.79	76.63 ($\uparrow 50.84$)
Qwen2.5-7B	60.84	92.74 ($\uparrow 31.90$)	50.92	87.59 ($\uparrow 36.67$)	32.59	80.61 ($\uparrow 48.02$)	32.02	76.45 ($\uparrow 44.43$)
Qwen3-4B	61.18	91.97 ($\uparrow 30.79$)	38.34	84.76 ($\uparrow 46.42$)	35.41	80.07 ($\uparrow 44.66$)	36.81	77.33 ($\uparrow 40.52$)
Granite3.3-8B	64.79	92.90 ($\uparrow 28.11$)	44.08	87.05 ($\uparrow 42.97$)	36.90	81.86 ($\uparrow 44.96$)	39.54	77.50 ($\uparrow 37.96$)
Granite4-3B	61.40	92.40 ($\uparrow 31.00$)	38.91	86.85 ($\uparrow 47.94$)	33.30	80.95 ($\uparrow 47.65$)	36.80	77.01 ($\uparrow 40.21$)
Gemma2-9B	65.53	92.47 ($\uparrow 26.94$)	61.45	89.35 ($\uparrow 27.90$)	40.80	83.30 ($\uparrow 42.50$)	41.15	77.93 ($\uparrow 36.78$)
Gemma3-4B	59.85	92.24 ($\uparrow 32.39$)	51.35	86.28 ($\uparrow 34.93$)	31.60	79.11 ($\uparrow 47.51$)	32.00	75.65 ($\uparrow 43.65$)
Average	61.62	92.41 ($\uparrow 30.79$)	46.20	86.77 ($\uparrow 40.58$)	34.22	80.88 ($\uparrow 46.66$)	34.84	77.06 ($\uparrow 42.22$)

Table 12: Impact of character-offset errors on Offset-based JSON performance. $F_1^{w/o \text{ offset}}$ removes the offset-matching requirement, revealing that models identify entities accurately but struggle with precise positional alignment. Green values indicate the absolute F1 improvement from ignoring offsets.

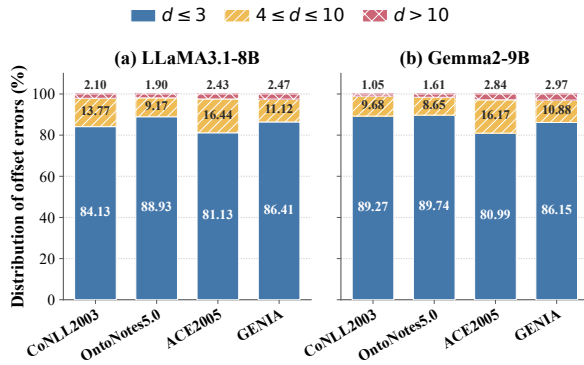


Figure 6: Distribution of character-offset errors by boundary displacement d for two representative models across four datasets.

Magnitude of character-offset errors. We further examine predictions whose entity text and label are correct but whose offsets are incorrect. For each prediction, we define the boundary displacement as $d = \max(|\hat{s} - s|, |\hat{e} - e|)$, where $(\hat{s}, \hat{e})$ and (s, e) are the predicted and gold start and end offsets, respectively. As shown in Figure 6, across all models and datasets, 81–90% of offset errors fall within 3 characters, and large deviations (>10 characters) account for only 1–3%.

Together, these analyses confirm that **the low Offset-based JSON scores are caused by character-level offset misalignment, not by failures in entity recognition**, and that most errors involve only small boundary deviations. These results indicate that exact character indexing is difficult for autoregressive language models. Clearer instructions and additional examples may reduce some errors, but they are unlikely to remove the problem entirely.

Metric	LLaMA3.1-8B	BERT-MRC
Training Time (h)	2.40	2.31
Trainable Parameters (M)	671	335
Peak GPU Memory (GiB)	27.89	21.62
F1	93.85	93.04

Table 13: Cost-benefit comparison between LLaMA3.1-8B and BERT-MRC on CoNLL2003.

G Cost-Benefit Analysis

We further conduct a cost-benefit experiment between the best-performing generative NER model, LLaMA3.1-8B with the Inline Bracketed format, and a representative traditional NER model, BERT-MRC, on CoNLL2003. Both models were trained on a single NVIDIA L40 GPU (48GB). The results are shown in Table 13.

While LLaMA3.1-8B requires slightly more resources than BERT-MRC, the gap is modest considering its strong performance (93.85 vs. 93.04). This trade-off demonstrates the viability of generative NER under resource-efficient fine-tuning.

H Full Precision, Recall, and F1 Results

To provide a complete view of model performance, we report Precision, Recall, and F1 scores for the flat and nested NER experiments in Table 18.

I Error Type Definitions

Prediction errors are categorized into eight distinct types. The detailed definitions are as follows:

Type Errors. These errors involve correct boundaries but incorrect labeling. *OOD Types* refer to predicted entity types that do not exist in the pro-

vided label set. *Wrong Types* denote cases where the predicted type is incorrect but belongs to the given label set.

Boundary Errors. These occur when the predicted span is imprecise. *Contain Gold* describes predicted mentions that enclose the gold mentions. *Contained by Gold* refers to predicted mentions that are inside the gold mentions. *Overlap with Gold* covers predictions that overlap with gold mentions but do not fall into the above two cases.

Other Errors. *Completely-O* represents predicted mentions that have no overlap with any gold mentions. *OOD Mentions* correspond to hallucinated predictions that do not appear in the input text. *Omitted Mentions* refer to gold entities that the model fails to identify.

J AI Assistance Statement

We utilized LLMs (e.g., ChatGPT) to assist in polishing the text, refining grammar, and improving the clarity of the manuscript. All content was reviewed and verified by the authors.

As detailed in Appendix B.3, we employed LLMs to summarize official annotation guidelines for CoNLL2003 dataset and generate standard biomedical definitions for GENIA dataset to construct the label definition schema used in our instruction tuning prompts.

Labels	Definition
ORG	A collective entity such as a company, institution, brand, political or governmental body, publication, or any organized group of people acting as a unit.
PER	A named individual, including humans, animals, fictional characters, and their aliases.
LOC	A geographical or spatial entity, including natural features, built structures, regions, public or commercial places, assorted buildings, and abstract or metaphorical places.
MISC	Named entities that are not persons, organizations, or locations, including derived adjectives, religions, ideologies, nationalities, languages, events, programs, wars, titles of works, slogans, eras, and types of objects.

Table 14: Label definitions for CoNLL2003 dataset.

Labels	Definition
PERSON	People, including fictional.
NORP	Nationalities or religious or political groups.
FAC	Buildings, airports, highways, bridges, etc..
ORG	Companies, agencies, institutions, etc..
GPE	Countries, cities, states.
LOC	Non-GPE locations, mountain ranges, bodies of water.
PRODUCT	Vehicles, weapons, foods, etc. (Not services).
EVENT	Named hurricanes, battles, wars, sports events, etc..
WORK_OF_ART	Titles of books, songs, etc..
LAW	Named documents made into laws.
LANGUAGE	Any named language.
DATE	Absolute or relative dates or periods.
TIME	Times smaller than a day.
PERCENT	Percentage (including “%”).
MONEY	Monetary values, including unit.
QUANTITY	Measurements, as of weight or distance.
ORDINAL	“first”, “second”.
CARDINAL	Numerals that do not fall under another type.

Table 15: Label definitions for OntoNotes5.0 dataset.

Labels	Definition
DNA	Deoxyribonucleic acid sequences or molecules, including genes, gene fragments, promoters, enhancers, and other genomic regions that encode or regulate biological information.
RNA	Ribonucleic acid molecules transcribed from DNA, encompassing messenger RNA, non-coding RNAs, and RNA transcripts involved in gene expression and regulation.
PROTEIN	Functional biological macromolecules composed of amino acids, including enzymes, receptors, transcription factors, and protein complexes produced by translation of RNA.
CELL_LINE	A population of cells derived from a single source and maintained in vitro through continuous culture, often genetically stable and used for experimental research.
CELL_TYPE	A class of cells defined by shared morphological, functional, and molecular characteristics within an organism, such as immune cells, epithelial cells, or neurons.

Table 16: Label definitions for GENIA dataset.

Labels	Definition
PER	Person entities are limited to humans. A person may be a single individual or a group.
ORG	Organization entities are limited to corporations, agencies, and other groups of people defined by an established organizational structure.
GPE	GPE entities are geographical regions defined by political and/or social groups. A GPE entity subsumes and does not distinguish between a nation, its region, its government, or its people.
LOC	Location entities are limited to geographical entities such as geographical areas and land-masses, bodies of water, and geological formations.
FAC	Facility entities are limited to buildings and other permanent man-made structures and real estate improvements.
VEH	A vehicle entity is a physical device primarily designed to move an object from one location to another, by (for example) carrying, pulling, or pushing the transported object. Vehicle entities may or may not have their own power source.
WEA	Weapon entities are limited to physical devices primarily used as instruments for physically harming or destroying other entities.

Table 17: Label definitions for ACE2005 dataset.

Models	CoNLL2003			OntoNotes5.0			Avg ^{Flat} _{F1}	ACE2005			GENIA			Avg ^{Nested} _{F1}
	P	R	F1	P	R	F1		P	R	F1	P	R	F1	
Pre-Trained NER Models														
BERT-Tagger	91.20	92.30	91.70	90.01	88.35	89.16	90.43	-	-	-	-	-	-	-
GNN-SL	93.02	93.40	93.20	91.48	91.29	91.39	92.30	-	-	-	-	-	-	-
ACE + Fine-tune	93.07	94.21	93.64	91.48	91.29	91.39	92.52	-	-	-	-	-	-	-
BERT-MRC+DSC	93.41	93.25	93.33	91.59	92.56	92.07	92.70	-	-	-	-	-	-	-
BERT-MRC	92.33	94.61	93.04	92.98	89.95	91.11	92.08	87.16	86.59	86.88	85.18	81.12	83.75	85.32
BINDER	-	-	-	-	-	-	-	89.60	90.50	90.00	83.40	78.30	80.80	85.40
W2NER	-	-	-	-	-	-	-	85.03	88.62	86.79	81.58	79.11	80.32	83.56
Biaffine+CNN	-	-	-	-	-	-	-	86.78	87.72	87.25	81.52	79.17	80.33	83.79
GPT-NER														
GPT-3 (ZS)	88.54	91.40	89.97	79.17	84.29	81.73	85.85	71.72	74.20	72.96	61.38	66.74	64.06	68.51
GPT-3 (ZS w/ Ver.)	89.47	91.77	90.62	79.64	84.52	82.08	86.35	72.63	75.39	73.46	61.77	66.81	64.29	68.88
GPT-3 (FS w/ Ver.)	89.76	92.06	90.91	79.89	84.51	82.20	86.56	72.77	75.51	73.59	61.89	66.95	64.42	69.01
Large Language Models (Inline Bracketed Format)														
LLaMA3.1-8B	93.50	94.21	93.85	91.23	91.32	91.28	92.57	87.16	87.85	87.51	81.93	76.62	79.18	83.35
LLaMA3.2-3B	93.17	93.91	93.54	89.18	89.21	89.19	91.37	85.18	86.47	85.82	81.77	76.69	79.15	82.49
Qwen2.5-7B	93.51	93.94	93.73	90.98	90.97	90.97	92.35	86.47	86.93	86.70	81.76	75.82	78.67	82.69
Qwen3-4B	93.05	93.68	93.37	89.25	89.35	89.30	91.34	85.56	87.43	86.48	81.94	76.56	79.16	82.82
Granite3.3-8B	93.38	94.14	93.76	90.37	90.26	90.31	92.04	86.08	87.56	86.81	81.24	76.64	78.87	82.84
Granite4-3B	92.75	93.56	93.15	89.59	90.05	89.82	91.49	85.93	87.66	86.78	81.09	75.91	78.42	82.60
Gemma2-9B	93.24	94.03	93.64	90.72	91.11	90.91	92.28	86.55	88.35	87.44	82.41	77.22	79.73	83.59
Gemma3-4B	93.32	94.03	93.68	90.01	90.17	90.09	91.89	85.40	87.03	86.20	81.75	75.76	78.64	82.42
Average	93.24	93.94	93.59	90.17	90.30	90.23	91.91	86.04	87.41	86.72	81.74	76.40	78.98	82.85
Large Language Models (Inline XML Format)														
LLaMA3.1-8B	93.57	94.09	93.83	90.84	91.05	90.94	92.39	85.30	86.98	86.13	80.78	76.18	78.41	82.27
LLaMA3.2-3B	92.80	93.33	93.06	89.14	89.46	89.30	91.18	85.31	87.24	86.26	80.21	76.32	78.22	82.24
Qwen2.5-7B	93.31	93.87	93.59	90.88	90.71	90.79	92.19	85.50	86.45	85.97	81.31	77.34	79.27	82.62
Qwen3-4B	92.95	93.36	93.15	88.99	88.93	88.96	91.06	84.89	86.45	85.66	81.39	77.88	79.60	82.63
Granite3.3-8B	92.91	93.75	93.33	89.87	90.04	89.95	91.64	86.87	87.67	87.27	81.42	76.89	79.09	83.18
Granite4-3B	92.75	93.38	93.07	89.33	89.58	89.46	91.27	85.50	88.23	86.84	81.07	76.30	78.61	82.73
Gemma2-9B	93.14	94.03	93.59	90.98	91.01	90.99	92.29	87.24	88.82	88.02	81.93	76.87	79.32	83.67
Gemma3-4B	92.73	93.71	93.22	90.08	90.23	90.15	91.69	85.41	85.83	85.62	81.22	75.87	78.46	82.04
Average	93.02	93.69	93.36	90.01	90.13	90.07	91.71	85.75	87.21	86.47	81.17	76.71	78.87	82.67
Large Language Models (Category-grouped JSON Format)														
LLaMA3.1-8B	92.08	93.04	92.56	89.87	89.82	89.84	91.20	85.69	86.59	86.13	79.77	76.87	78.29	82.21
LLaMA3.2-3B	92.18	92.63	92.41	87.12	87.56	87.34	89.88	83.39	83.99	83.69	79.36	76.68	77.99	80.84
Qwen2.5-7B	92.44	93.04	92.74	89.49	89.46	89.48	91.11	84.79	84.88	84.84	79.37	76.97	78.15	81.50
Qwen3-4B	92.28	92.67	92.47	87.18	88.18	87.67	90.07	83.74	84.98	84.35	79.86	77.14	78.48	81.42
Granite3.3-8B	92.73	93.22	92.97	88.74	89.22	88.98	90.98	85.61	86.85	86.23	80.14	76.87	78.47	82.35
Granite4-3B	91.48	92.07	91.78	88.51	88.37	88.44	90.11	84.02	85.54	84.77	79.79	77.03	78.38	81.58
Gemma2-9B	93.18	93.79	93.48	90.55	90.53	90.54	92.01	86.40	86.88	86.64	80.09	77.81	78.93	82.79
Gemma3-4B	92.36	92.71	92.53	89.19	89.16	89.17	90.85	84.54	84.48	84.51	78.83	75.60	77.18	80.85
Average	92.34	92.90	92.62	88.83	89.04	88.93	90.78	84.77	85.52	85.14	79.65	76.87	78.23	81.69
Large Language Models (Occurrence-based JSON Format)														
LLaMA3.1-8B	92.89	94.19	93.54	90.27	90.76	90.52	92.03	82.78	84.06	83.41	80.07	77.30	78.66	81.04
LLaMA3.2-3B	92.96	94.21	93.58	87.23	88.05	87.64	90.61	80.27	81.59	80.93	78.68	76.19	77.41	79.17
Qwen2.5-7B	93.10	93.94	93.52	89.77	90.32	90.04	91.78	81.46	81.92	81.69	78.60	76.58	77.58	79.64
Qwen3-4B	92.26	93.56	92.91	87.51	88.45	87.98	90.45	80.85	82.31	81.58	79.32	77.63	78.47	80.03
Granite3.3-8B	92.30	93.59	92.94	88.84	89.57	89.20	91.07	82.08	83.99	83.02	79.67	76.93	78.27	80.65
Granite4-3B	92.29	93.24	92.76	88.31	88.94	88.63	90.70	80.84	82.94	81.88	78.55	75.99	77.25	79.57
Gemma2-9B	92.99	94.12	93.55	90.08	90.50	90.29	91.92	83.49	85.11	84.29	80.30	77.96	79.12	81.71
Gemma3-4B	92.74	93.61	93.17	88.60	88.37	88.49	90.83	81.43	81.72	81.58	79.25	73.06	76.03	78.81
Average	92.69	93.81	93.25	88.83	89.37	89.10	91.17	81.65	82.95	82.30	79.30	76.45	77.85	80.07
Large Language Models (Offset-based JSON Format)														
LLaMA3.1-8B	62.91	63.53	63.22	52.99	53.63	53.31	58.27	35.53	36.17	35.85	35.16	34.09	34.62	35.24
LLaMA3.2-3B	56.01	56.25	56.13	31.06	31.34	31.20	43.67	27.03	27.59	27.31	26.17	25.43	25.79	26.55
Qwen2.5-7B	60.52	61.17	60.84	50.67	51.17	50.92	55.88	32.54	32.64	32.59	32.52	31.53	32.02	32.31
Qwen3-4B	60.86	61.51	61.18	38.03	38.66	38.34	49.76	35.06	35.78	35.41	37.51	36.14	36.81	36.11
Granite3.3-8B	64.49	65.09	64.79	43.78	44.38	44.08	54.44	36.48	37.33	36.90	40.06	39.03	39.54	38.22
Granite4-3B	60.99	61.81	61.40	38.71	39.10	38.91	50.16	33.00	33.60	33.30	37.55	36.08	36.80	35.05
Gemma2-9B	65.21	65.86	65.53	61.15	61.75	61.45	63.49	40.45	41.16	40.80	41.75	40.55	41.15	40.98
Gemma3-4B	59.67	60.02	59.85	51.50	51.19	51.35	55.60	31.67	31.52	31.60	33.56	30.57	32.00	31.80
Average	61.33	61.91	61.62	45.99	46.40	46.20	53.91	33.97	34.47	34.22	35.53	34.18	34.84	34.53

Table 18: Complete Precision, Recall, and F1 results of pre-trained models, GPT-3, and open-source LLMs on flat and nested NER datasets.