DynamicNER: A Dynamic, Multilingual, and Fine-Grained Dataset for LLM-based Named Entity Recognition

Hanjun Luo[1*], **Yingbin Jin**[3], **Yiran Wang**[5], **Xinfeng Li**[4], **Tong Shang**[6], **Xuecheng Liu**[2], **Ruizhe Chen**[2], **Kun Wang** [4], **Hanan Salam**[1], **Qingsong Wen**[7], **Zuozhu Liu**[2†]
[1]New York University Abu Dhabi, [2]Zhejiang University,
[3]The Hong Kong Polytechnic University, [4]Nanyang Technology University,
[5]University of Electronic Science and Technology of China,
[6]Texas A&M University, [7]Squirrel AI

Abstract

The advancements of Large Language Models (LLMs) have spurred a growing interest in their application to Named Entity Recognition (NER) methods. However, existing datasets are primarily designed for traditional machine learning methods and are inadequate for LLM-based methods, in terms of corpus selection and overall dataset design logic. Moreover, the prevalent fixed and relatively coarse-grained entity categorization in existing datasets fails to adequately assess the superior generalization and contextual understanding capabilities of LLM-based methods, thereby hindering a comprehensive demonstration of their broad application prospects. To address these limitations, we propose DynamicNER, the first NER dataset designed for LLM-based methods with dynamic categorization, introducing various entity types and entity type lists for the same entity in different context, leveraging the generalization of LLM-based NER better. The dataset is also multilingual and multi-granular, covering **8** languages and **155** entity types, with corpora spanning a diverse range of domains. Furthermore, we introduce CascadeNER, a novel NER method based on a two-stage strategy and lightweight LLMs, achieving higher accuracy on fine-grained tasks while requiring fewer computational resources. Experiments show that DynamicNER serves as a robust and effective benchmark for LLM-based NER methods. Furthermore, we also conduct analysis for traditional methods and LLM-based methods on our dataset. Our code and dataset are openly available at https://github.com/Astarojth/DynamicNER.

1 Introduction

Recent advances in Large Language Models (LLMs) have transformed the landscape of NLP (Naveed et al., 2023). Among the impacted tasks, Named Entity Recognition (NER) has seen notable methodological shifts (Xie et al., 2023). Leveraging LLMs' strong generalization and contextual understanding capabilities, existing LLM-based methods (Shao et al., 2023; Li and Zhang, 2023) show superior performance compared to traditional machine learning (ML) methods (Wang et al., 2020; Yan et al., 2021; Curran and Clark, 2003) in low-resource, multilingual, or few-/zero-shot settings. As a result, LLM-based NER has garnered growing interest in these settings (Xiao et al., 2024), offering a promising path toward more scalable and adaptable information retrieval.

Despite recent progress, there are no existing NER datasets specifically optimized for the characteristics of LLMs, thereby limiting both their effective evaluation and the development of optimized methods. Existing NER datasets employ static categorization with a fixed set of entity types, preventing the evaluation of LLMs' ability to generalize to novel entity types and varying levels of granularity, especially in few-shot or zero-shot settings. Moreover, while some datasets address domain-specific corpora with specialized entity types (Kim et al., 2003; Liu et al., 2021), others target multi-grained classifications (Ding et al., 2021), or multilingualism (Malmasi et al., 2022), no existing dataset simultaneously incorporates all three aspects. This fragmentation hinders comprehensive evaluation of LLM-based methods, which are particularly well-suited to handling such challenges. As a result, current datasets fall short in revealing performance differences between LLM-based methods, fail to capture their full potential and limitations, and ultimately impede the advancement of more effective NER solutions.

To address these gaps, we develop DynamicNER, the first NER dataset optimized for LLM-based methods and the first to support dynamic categorization. It employs multiple strategies to dynamically adjust entity labels, type lists, and

*hl6266@nyu.edu
†Corresponding Author, zuozhuliu@intl.zju.edu.cn

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 16523–16547
November 4-9, 2025

granularity levels during annotation. This design simulates the complexity and uncertainty of entity types in real-world, general-purpose scenarios—a challenge that traditional ML methods with fixed entity categorization struggle to address, enabling a more rigorous evaluation of LLM-based NER methods' ability to generalize across diverse and evolving scenarios. We introduce cohesion and distribution balance metrics to guide the evaluation and optimization of the annotation process. The entire procedure is algorithmically automated and openly available, ensuring both reliability and reproducibility.

In addition, DynamicNER is a multilingual and multi-granular dataset, featuring **8** languages, **8** coarse-grained types, **31** medium-grained types, and **155** fine-grained types. The multilingual and fine-grained nature of DynamicNER not only provides necessary support for the dynamic categorization, but also places higher demands on the NER methods. Many existing benchmarks are no longer difficult enough to reveal the performance boundaries of advanced methods, and the challenging DynamicNER will help future researchers to explore performance frontiers. Its entity types and corpora span a wide range of professional domains, including science, computer engineering, medicine, history, and arts. These features offer an unprecedented level of semantic and linguistic coverage for complex NER evaluation.

Furthermore, our evaluation on DynamicNER reveals significant limitations in existing LLM-based methods, particularly when migrating to lightweight LLMs (models with 1.5B to 7B parameters) for local deployment. While approaches leveraging commercial models like ChatGPT (Brown, 2020) achieve high performance, this reliance introduces practical challenges related to API costs and privacy risks. API-based usage is often prohibitively expensive for real-world NER applications, and privacy remains a critical concern (Zhang et al., 2024; Das et al., 2024; Deng et al., 2025).

To address this issue, we propose CascadeNER, a universal and multilingual NER framework that achieves competitive performance with lightweight LLMs, comparable to existing LLM-based methods that rely on costly commercial models. CascadeNER employs a two-stage strategy by dividing NER as two in-context text generation sub-tasks, extraction and classification, instead of treating it as a traditional sequential labeling task. To reduce task complexity and better capture in-context dependencies, CascadeNER assigns each stage of the NER process–extraction and classification–to separate fine-tuned lightweight LLMs within a model cascading framework (Varshney and Baral, 2022). This modular architecture, combined with the integration of prior knowledge, enables effective multilingual performance in low-resource settings.

We evaluate a BERT-based (Devlin et al., 2018) supervised method, two LLM-based methods, and our proposed CascadeNER on DynamicNER. We also conduct evaluations of CascadeNER against existing methods on existing datasets. Results demonstrate that DynamicNER effectively evaluates the performance of LLM-based methods in low-resource and complex NER tasks, while CascadeNER outperforms existing LLM-based methods significantly with smaller models. Moreover, this work offers the first comprehensive comparison and analysis of existing LLM-based NER methods, with an emphasis on multilingual and fine-grained scenarios.

Our contributions are summarized as follows:

- We develop DynamicNER, the first NER dataset optimized for LLM-based NER methods, featuring a novel **dynamic categorization** system. The dataset which supports **8** languages, **155** entity types, and **3** levels of granularity, enabling comprehensive evaluation across diverse linguistic and semantic settings.
- We propose CascadeNER, a universal NER framework, which outperforms existing LLM-based methods using only lightweight LLMs and a two-stage strategy.
- We conduct a comprehensive evaluation of LLM-based NER methods on our challenging dataset and identify key challenges and future directions for the field.

2 Related Works

Named Entity Recognition. NER is the task of identifying named entities in text and classifying them into predefined categories. Supervised methods, such as BiLSTM (Yu et al., 2020) and BERT-MRC (Li et al., 2019a), currently dominate this task. They generally rely on large amounts of training data to achieve strong performance, which limits their application in low-resource scenarios. Some researchers apply LLMs to address this issue. GPT-NER (Wang et al., 2023) employs the GPT-3

model and re-frames the task as single-entity labeling, supporting few-shot/zero-shot learning. It achieves comparable performance to supervised methods in traditional scenarios and excels in low-resource scenarios. PromptNER (Ashok and Lipton, 2023) achieves state-of-the-art (SOTA) accuracy in datasets with complex classification (Liu et al., 2021; Ding et al., 2021) with GPT-4 and Chain-of-Thought (CoT) (Wei et al., 2022), yet performs significantly worse than GPT-NER and supervised methods in classical NER datasets like CoNLL2003 (Tjong Kim Sang and De Meulder, 2003). Furthermore, several studies apply LLM-based NER in domain-specific tasks (Li and Zhang, 2023; Shao et al., 2023; Keloth et al., 2024), focusing on science and medicine. Their performances surpass supervised methods in those domains, further highlighting the potential of LLMs in low-resource and complex NER tasks.

Dataset	#Language	#Coarse	#Fine	Domain
CoNLL2002	2	4	no	News
CoNLL2003	2	4	no	News
ACE2005	2	7	41	News
OntoNotes 5.0	3	18	no	General
CrossNER	1	9-17	no	Multi Domain
FEW-NERD	1	8	66	General
PAN-X	282	3	no	General
MultiCoNER	11	6	33	General
I2B2	1	22	no	Medical
DynamicNER (ours)	**8**	**8**	**155**	**Multi Domain**

Table 1: Overview of NER datasets. Notably, DynamicNER covers a wide range of cross-domain categories, such as art, medicine, and biology, thus offering better generalization compared to other general datasets.

NER Datasets. There have been a considerable number of NER datasets in various domains (Tjong Kim Sang, 2002; Kim et al., 2003; Doddington et al., 2004; Walker et al., 2006; Weischedel et al., 2011; Pradhan et al., 2013; Derczynski et al., 2017; Katz et al., 2023). However, these existing datasets exhibit several limitations, making them unsuitable for LLM-based NER. Most previous multilingual NER datasets adopt coarse-grained classification, no longer meeting the fine-grained requirements of contemporary flat NER applications. Even existing fine-grained datasets demonstrate clear limitations in category coverage and granularity, falling short of being truly "universal." For instance, FewNERD, despite having 66 entity types, suffers from highly imbalanced data distribution, which affects its reliability for evaluating few-shot learning capabilities. Furthermore, current datasets fail to adequately address the generalization capabilities of LLMs, hindering the comprehensive training and evaluation of LLM-based NER methods. Table 1 presents a simple comparison between DynamicNER and existing multilingual or fine-grained datasets.

3 DynamicNER Dataset

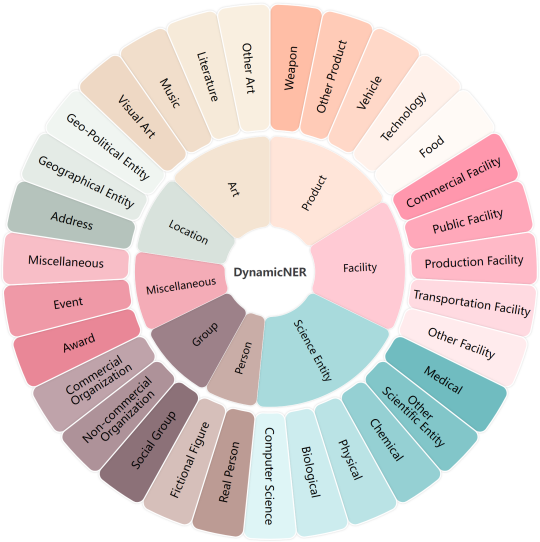

Figure 1: The coarse-grained and medium-grained categories of DynamicNER. Detailed categories are provided in Appendix P.

DynamicNER spans **8** languages: English, Chinese, Spanish, French, German, Japanese, Korean, and Russian. In terms of categorization, it is the first NER dataset with three-level granularity categorization, encompassing **8** coarse-grained types, **31** medium-grained types, and **155** fine-grained types, as shown in Figure 1. Like other NER datasets, DynamicNER is divided into train, dev, and test sets. Data volumes for different languages and parts shown in Appendix D. To develop DynamicNER, we first collect unlabeled corpus from Wikipedia. Then we manually extract sentences from corpora and conduct comprehensive manual annotation on the collected sentences, strictly following our 3-level, 155-type fine-grained entity schema. This initial annotated result, containing all entity types without any automated adjustments, is what we define as the `Base Version` of DynamicNER. It represents our most complete, static knowledge base. Subsequently, using the `Base Version` as input, we apply the automated dynamic categorization according to four metrics, generating the `Dynamic Version`. To ensure reproducibility and support future research on dynamic categorization,

we provide the full code for the pipeline, including all settings and parameters in our repository.

Corpora Collection and Annotation. Wikipedia provides multilingual, domain-specific corpora with clear hierarchical and indexing systems, serving as a rich resource for our research. We utilize legal Wikipedia-API to filter and download corpora across different languages and categories, followed by manual selection and annotation of sentences. We particularly focus on corpora containing long texts and complex contexts, with a special emphasis on encyclopedic content from professional domains such as science, medical, arts, engineering, and law, to ensure DynamicNER's coverage of specialized domains. In addition, we have incorporated multilingual corpora from the social media platforms X and Weibo to ensure that our dataset also includes a substantial portion of colloquial language. After completing 50% of the annotation process of each language, we annotate corpora from categories related to underrepresented entity types to achieve a balanced entity type distribution. For instance, when the entities of "Algorithm" are significantly less than others, we use more corpora from Computer Science category. Thus, DynamicNER ensures balanced entity distribution, and includes rare entities and emerging fields that are inadequate in existing datasets, ensuring comprehensive coverage across diverse domains. Details about the manual annotation process is provided in Appendix J.

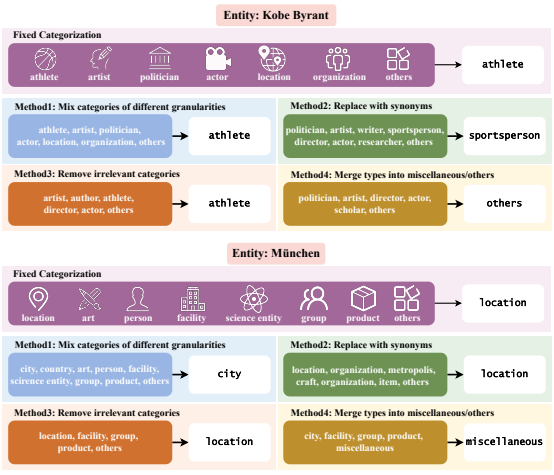

Figure 2: Examples of dynamic categorization. Details of the four strategies are provided in Appendix A.

Dynamic Categorization. The dynamic version improves model generalization and reduces overfitting risk by dynamically adjusting entity labels and corresponding entity type lists during annotation, including ❶ mixing types of different granularities, ❷ replacing types with synonyms, ❸ using type lists without irrelevant types, and ❹ merging certain types into miscellaneous/others, as shown in Figure 2. This method addresses the mismatch between existing datasets and few-shot/zero-shot training needs, better simulating real-world scenarios, and is particularly critical for evaluating methods relying on complex prompt designs (e.g., CoT). Unlike traditional few-shot learning, some LLM-based methods only use few-shot demos to help the model understand the task or format, without requiring knowledge of entity types. They can perform NER across different datasets with fixed few-shot demos, resembling zero-shot NER. Research shows this method is more effective than typical zero-shot NER (Zhang et al., 2022). In methods that uses complex prompt designs like CoT to guide the reasoning, even few-shot CoT only conveys the CoT process rather than task-relevant knowledge, the performance of prompt-guided zero-shot CoT is significantly worse than few-shot CoT, making zero-shot restrictions inadequate for reflecting their true capability. However, in NER, models inevitably learn about entity types through few-shot demos, which limits generalization evaluations on fixed-category datasets. Our method significantly mitigates this limitation by varying entity types and lists, isolating the impact of prior type knowledge. Notably, as dynamic categorization is a subtractive method applied to a comprehensive and detailed categorization system, this method relies heavily on DynamicNER's fine-grained taxonomy, which includes 3 levels of granularity and 155 entity types. Consequently, this method may not be suitable for all datasets. Based on our empirical findings, we recommend that future researchers adopting this categorization method use datasets that feature at least two levels of granularity and over 20 fine-grained entity types.

Categorization Metrics. Random dynamic categorization not only exhibits poor reproducibility and explainability, but may also lead to data quality degradation. For training, inappropriate categorization may result in inconsistent learning objectives and overfitting risks (Ren et al., 2016). For evaluation, certain categories may experience imbalanced sample distribution and boundary ambiguity, reducing the comprehensiveness and consistency of

evaluation (Obeidat et al., 2019). Thus, we design four metrics to regulate the dynamic categorization: cohesion, normalized entropy, Gini coefficient, and variation coefficient. The definition and calculation methods are provided in Appendix B.

Categorization Process. The dynamic categorization process consists of 4 rounds of re-categorization, each sequentially corresponding to an adjustment method, and different metrics are employed in each round to guide the optimization. This hierarchical design enables each stage to focus on distinct data characteristics and optimization objectives, preventing interference between metrics while ensuring proper optimization direction, thus achieving a progressive optimization. We do not use all metrics in each evaluation, considering that certain metrics may have overlapping or conflicting effects at specific stages. For instance, normalized entropy and Gini coefficient both measure distribution uniformity, while improving cohesion may lead to more concentrated distribution and consequently lower entropy values. Figure 4 illustrates the metrics and methods corresponding to each round. Appendix C explains the reasons of metric selection for each round.

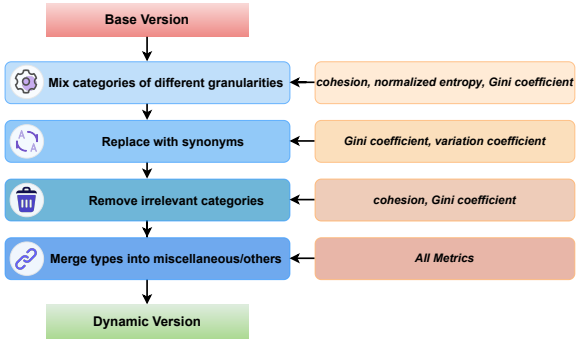

Figure 4: Pipeline of dynamic categorization.

4 CascadeNER

4.1 Framework

Background. Some existing supervised methods suggest that separating extraction and classification can improve NER performance as this two-stage strategy reduces the task complexity (Shen et al., 2021; Wu et al., 2022). However, these methods are limited by traditional models, failing to incorporate LLMs, and exhibit notable performance deficiencies that make them inferior to other methods treating NER as a single task. On the other hand, LLM-based methods demonstrate superior performance compared to traditional methods in Named Entity Extraction (Sancheti et al., 2024) and Text Classification (Gasparetto et al., 2022), indicating the potential of two-stage in LLM-based NER.

Framework Design. We propose the framework to implement two-stage strategy in LLM-based NER. CascadeNER divides NER into two sequential, independently executed, generation-based sub-tasks. In the first sub-task, extraction, the model generate a sentence where all named entities are marked with identifiers and individually re-embeds each entity back into its context, resulting in sentences with identifiers at the number of entities. In the second sub-task, classification, the model receives sentences with identifiers and a list of entity types, and labels one entity at a time.

Model Cascading. To optimize performance while reducing computational resources, CascadeNER employs model cascading, where the extraction and classification sub-tasks are handled separately by two specialized fine-tuned LLMs. This structure allows each model to focus on its specific sub-task, maximizing performance on simpler, more specialized tasks. The architecture enables CascadeNER to be particularly suitable for lightweight LLMs, as each model only focus on a simplified task. Existing research shows that fine-tuned lightweight LLMs can achieve performance close to normal LLMs on specific simple tasks (Hu et al., 2024a). Through the implementation of two-stage strategy and model cascading, CascadeNER effectively leverages the advantages of lightweight LLMs in simple tasks, maintaining high accuracy while reducing computational resource usage.

Pipeline. A simplified pipeline of CascadeNER is shown in Figure 3. Upon receiving the input sentence, CascadeNER first processes the sentence by the extractor to mark all entities with identifiers, and re-embeds each entity back, resulting in sentences with identifiers around the named entities. These sentences are then individually fed into the classifier, which classifies each entity based on the context and the input type list. For multi-granularity data, CascadeNER allows a progressive strategy, significantly improving CasacdeNER's performance in accurate fine-grained classification. The detailed steps of extraction and classification are discussed in following sections. Examples about the prompts are provided in Appendix M.

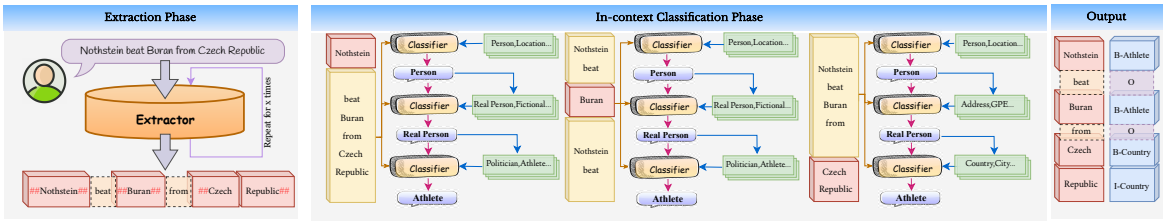

Figure 3: Use a sentence and the multi-granularity categories of DynamicNER as the example. The extractor and classifier are the two different lightweight LLMs used in CascadeNER. Azure boxes represent the specific type list for the input of the classifier. Blue boxes represent the sentence input.

4.2 Extraction

Prompt Design. In the extraction sub-task, we utilize a generation-based extraction method, where special tokens "##" are used to surround any entities identified in the sentence, regardless of the number of entities or their types. Figure 5 shows an instance for the prompt and corresponding response.

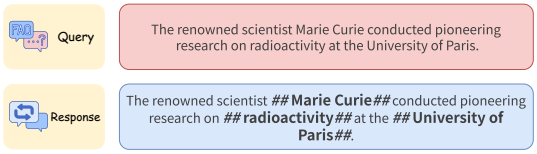

Figure 5: Example prompts of extraction.

This method, compared to conventional sequential extraction, avoids requiring LLMs to perform text alignment, thus reducing task complexity. Comparing similar methods (Wang et al., 2023; Hu et al., 2024b), CascadeNER's query contains only the sentence, without any task descriptions, demonstrations, or category information. The response exclusively uses "##" as the identifiers, and all entities are extracted without specifying categories. CascadeNER achieves low-cost NER by using simple prompts and better generalization by treating all entities uniformly. A detailed comparison with existing methods and further advantages of our method are shown in Appendix H.

Result Fusion. After conducting extensive experiments, we find that the extractor's precision consistently exceeds recall, regardless of the model or dataset, indicating that while correct entities are effectively identified, there is a tendency for under-detection. To mitigate this issue, we introduce a union strategy in result fusion (Ganaie et al., 2022), allowing multiple extraction for one sentence and taking the union of the results to maximize recall. For cases of entity nesting, where different extraction rounds produce overlapping or nested entities, we apply a length-first strategy, retaining the longer entity, as longer entities generally carry more semantic meaning (Nguyen and Cao, 2010). For example, "Boston University" is semantically more accurate than "Boston" in the context of "She studies in Boston University". The formula of our strategy is shown below:

$$E_{\text{final}} = \bigcup_{i=1}^{n} \left\{ \arg\max_{e \in E_i} \text{length}(e) \right\} \quad (1)$$

where E_i is the set of extracted entities from the i-th extraction, n is the number of extraction rounds, E_{overlap} is the set of overlapping or nested entities, and $\text{length}(e)$ is the length of entity e. The effects of the number of extraction repetitions and other details are provided in Appendix F.1.

4.3 Classification

Prompt Design. In the classification sub-task, we employ a generation-based in-context classification method, where we input the categories and the sentence with one entity surrounded by "##", and require the classifier to generate the label for that entity. This method re-embeds the entity into the sentence for classification, which utilizes the self-attention architecture of LLMs for contextual information and improves accuracy compared to entity-level classification. Figure 7 is an example:

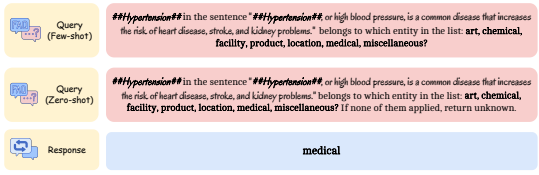

Figure 7: Example prompts of classification.

In zero-shot scenarios, we use a slightly different prompt. Due to differences in entity categorization

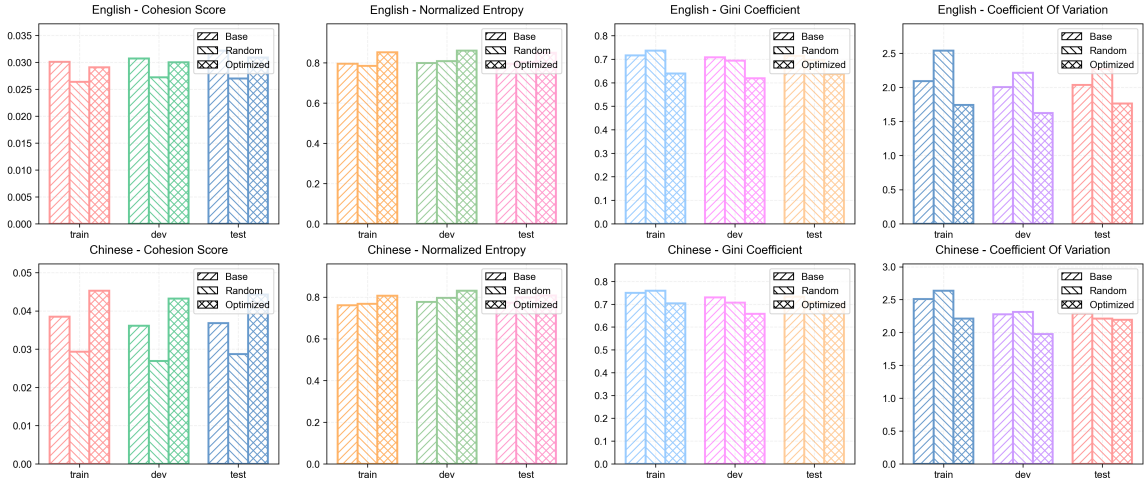

Figure 6: Quantitative categorization metric results for 3 versions DynamicNER in English and Chinese. Generally, higher cohesion and normalized entropy, or lower Gini coefficient and variation coefficient, indicate better quality.

across datasets, some entities in one dataset may be overlooked in others. We append the query with `If none of them applied, return unknown` to handle situations where the extracted entity cannot be classified into the provided categories, enhancing CascadeNER's generalization.

Multi-granularity. For multi-granularity data, we apply a progressive strategy. After obtaining the coarse-grained result, CascadeNER use the result to index the corresponding sub-categories and classify again, continuing this process until no further classification is possible:

$$L_i^{\text{fine}} = f_{\text{fine-classify}}(L_i^{\text{coarse}}, \text{subcategories}) \quad (2)$$

where L_i^{fine} is the fine-grained label, L_i^{coarse} is the generated coarse-grained label, and subcategories are the subcategories under the coarse-grained.

5 Experiment

In this section, we first present the categorization metrics of DynamicNER before and after dynamic categorization, followed by a comparative analysis of existing methods and CascadeNER's performance on different versions of DynamicNER. We also conduct experiments of CascadeNER and baselines on existing datasets in Appendix E, and ablation studies in Appendix F. In evaluations across existing datasets, CascadeNER, with base models fine-tuned using the dynamic version of DynamicNER, demonstrates consistent excellence in all datasets and achieves new SOTA performance in FewNERD and CrossNER. This confirms that DynamicNER not only provides exceptional effectiveness for evaluating LLM-based NER methods but also offers substantial value in training.

5.1 Categorization Quality Evaluation

To demonstrate that our dynamic categorization improves dataset generalization while maintaining dataset quality, we conduct comparative experiments across three versions of DynamicNER: the Base Version, a version with random parameters for dynamic categorization, and the optimized Dynamic Version. We still employ the 4 metrics for evaluating dataset quality, whose detailed definition are provided in Appendix B. For reliability of the random version's results, we conduct five independent tests and use the average results. Due to space limitations, we only present results for English and Chinese in Figure 6 here. Other quantitative results are provided in Appendix Q.

Language	**en**	**es**	**fr**	**ru**	**de**	**zh**	**ja**	**kr**
# Lists	725	455	501	377	465	786	553	478

Table 3: The numbers of entity type lists of each language after dynamic categorization. In some scenarios, this can be equivalent to having 700+ distinct datasets.

Experimental results demonstrate that our dynamic categorization significantly increases data diversity, as shown in Table 3, while maintaining or improving dataset quality compared to the base version. The quality of the dynamic version also considerably surpasses the random version. These results comprehensively validate the reliability and effectiveness of our method.

Model	Dynamic-Supervised								Dynamic-Fewshot							
	en	es	fr	ru	de	zh	ja	ko	en	es	fr	ru	de	zh	ja	ko
G-1.5B	47.6	39.7	38.0	37.6	37.3	41.2	35.7	36.1	36.9	32.2	31.9	30.5	30.3	35.8	31.9	32.6
G-7B	52.3	46.4	44.8	44.8	45.7	48.1	42.3	42.1	42.7	37.3	38.2	36.8	36.5	41.1	37.3	38.6
G-GPT	60.6	57.3	56.5	55.6	55.9	58.4	54.9	53.8	49.2	46.9	47.5	47.2	47.0	48.9	47.7	48.3
P-1.5B	23.2	20.8	18.5	16.3	17.5	22.7	18.0	17.3	20.5	17.9	16.2	15.9	16.1	19.9	16.0	15.9
P-7B	44.3	35.8	33.2	32.5	31.9	40.4	37.4	35.6	39.8	33.2	32.1	31.8	31.5	37.8	35.6	34.5
P-GPT	53.0	50.5	51.2	47.9	50.2	52.3	48.7	48.5	49.4	48.5	47.1	46.6	46.0	47.4	44.1	44.0
C-1.5B	62.8	55.7	52.8	51.1	48.8	58.9	54.1	52.7	49.7	44.1	44.0	43.4	42.9	48.5	43.1	43.8
C-7B	68.2	61.5	55.3	52.9	51.4	64.5	58.8	55.3	55.7	49.9	49.7	46.5	46.1	52.9	50.2	50.0
C-GPT	**73.1**	**67.1**	**67.8**	**66.9**	**67.6**	**68.3**	**67.4**	**67.9**	**61.3**	**57.4**	**56.9**	**56.2**	**56.0**	**59.7**	**56.8**	**56.4**

Table 2: The results of supervised learning with dynamic version and few-shot learning with dynamic version. G means GPT-NER, P means PromptNER, and C means CascadeNER. The results indicate that, due to its unprecedentedly detailed categorization and multilingual coverage, DynamicNER is an extremely challenging flat NER dataset, placing higher demands on methods' generalization capability.

5.2 DynamicNER Experiment

Baseline Selection. In our experiments for DynamicNER, we evaluate four NER methods: XLM-RoBERTa (Conneau et al., 2020), GPT-NER (Wang et al., 2023), PromptNER (Ashok and Lipton, 2023), and our CascadeNER. XLM-RoBERTa is a famous BERT-based multilingual model widely used as a baseline in multilingual NER research Malmasi et al. (2022); Fetahu et al. (2022), thus being selected as our baseline representing supervised methods. GPT-NER and PromptNER are two major general LLM-based NER methods that achieve performance significantly superior to supervised methods in low-resource scenarios through sophisticated prompt design and powerful GPT models, as discussed in Section 2 and Appendix H.

Model Selection. Given the lack of existing lightweight LLM-based NER methods and controlled variable principles, we evaluate two LLM-based methods and CascadeNER using three LLMs: Qwen2.5-1.5B (Yang et al., 2024), Qwen2.5-7B, and GPT-4o (Hurst et al., 2024). The lightweight LLMs of Qwen series perform exceptionally across benchmarks and gaining widespread recognition. According to HuggingFace (2024), Qwen2.5-1.5B is the most downloaded open-source model in 2024. Therefore, we select Qwen2.5-1.5B and 7B to represent the current best-performing lightweight LLMs. GPT-4o is the most widely-used current general commercial LLM, and its previous versions are employed in GPT-NER and PromptNER, making it our choice. In CascadeNER, the extractor and the classifier use the same base model.

Implementation. For the supervised method, XLM-RoBERTa is trained and only evaluated with the base version of DynamicNER. As its fixed classification output layer corresponds to a predefined set of entity types and any modification to the entity type list necessitates full model retraining, it cannot be evaluated with the dynamic version. For LLM-based methods, we conduct experiments under three scenarios: supervised learning with base version, supervised learning with dynamic version, and few-shot learning with dynamic version. Training data for GPT-NER and CascadeNER is obtained through format conversion. For PromptNER, as its prompt involves complex designs such as CoT, we utilize LLM-generated prompts by GPT-4o using prompts from its paper as few-shot demonstrations and manually verified the prompts. The repetition count i of CascadeNER for result fusion is set to 3. Potential data contamination are discussed in Appendix N. The finetuning settings and hyperparameters of CascadeNER are provided in Appendix L.

Model	Base-Supervised							
	en	es	fr	ru	de	zh	ja	ko
BERT	41.9	33.5	29.1	23.4	32.9	29.2	27.2	28.6
G-1.5B	50.2	43.5	40.4	39.8	39.3	44.1	38.9	38.7
G-7B	55.1	48.2	47.2	44.0	48.1	50.9	44.8	44.5
G-GPT	62.4	58.3	57.9	56.8	56.9	60.4	57.3	55.9
P-1.5B	21.6	18.6	17.1	14.9	15.8	20.7	16.4	15.9
P-7B	41.1	32.9	31.0	30.7	30.3	47.4	35.6	29.6
P-GPT	49.7	47.7	48.2	45.9	46.6	48.6	45.7	45.4
C-1.5B	67.6	59.9	57.9	55.7	53.5	64.0	58.5	55.1
C-7B	73.8	65.5	60.3	59.6	61.4	69.8	65.3	62.7
C-GPT	**77.1**	**71.7**	**69.9**	**70.3**	**70.8**	**74.3**	**72.4**	**70.9**

Table 4: The results of supervised learning with base version. BERT represents XLM-RoBERTa. The supervised method XLM-RoBERTa performs terribly.

Results. As presented in Table 2 and 4, CascadeNER achieves a significant advantage on DynamicNER, demonstrating its strong generaliza-

tion. The supervised method XLM-RoBERTa performs terribly, as DynamicNER's low-resource characteristics make it more suitable for evaluating LLM-based methods. For LLM-based methods, the 3 methods show significant performance variations across different datasets and models. When using GPT-4o and transitioning from supervised to few-shot, PromptNER exhibits notably smaller performance degradation, partially reflecting the generalization advantages of reasoning-focused approaches. However, when migrating to lightweight LLMs, these methods show significantly larger performance drops compared to the other two methods. GPT-NER and CascadeNER demonstrate generally similar performance patterns, but GPT-NER shows more pronounced degradation when migrated to lightweight LLMs, while CascadeNER achieves a greater performance advantage on the dynamic version compared to the base version, validating the effectiveness of the two-stage strategy in the complex classification. CascadeNER also exhibits notable computational efficiency from two key perspectives. First, it is parameter-efficient, leveraging two 7B LLMs to achieve performance superior to baselines that rely on a 100B-level commercial model. Second, its two-stage design enables the use of concise prompts that drastically reduce token consumption compared to competing methods, as discussed in Appendix H, as validated by our cost analysis in Appendix I. This efficiency makes CascadeNER a powerful yet practical solution.

5.3 Performance Across Granularities

To investigate how different methods handle increasing classification complexity, we conduct an analysis across the three granularity levels of DynamicNER. We evaluate all LLM-based methods on the Base version (as the Dynamic version doesn't have clear granularity levels) for English and Chinese, with results presented in Table 5.

The results lead to two key conclusions. First, as expected, the performance of all models declines as the number of entity types increases from coarse to fine, confirming that fine-grained classification is a significantly more challenging task. More importantly, the analysis reveals a stark difference in how the methods handle this increased complexity. Methods that rely on a single, complex prompt, such as PromptNER, suffer a dramatic performance collapse when scaled to the 155 categories. In contrast, our CascadeNER framework shows remarkable resilience, with a more graceful and controlled degradation in performance. This analysis empirically validates the core motivation for DynamicNER, demonstrating that a challenging, fine-grained benchmark is necessary to expose the limitations of existing methods and accurately evaluate the robustness of purpose-built frameworks.

Model	English			Chinese		
	Coarse	Medium	Fine	Coarse	Medium	Fine
G-1.5B	59.3	55.0	50.2	59.7	54.5	44.1
G-GPT	70.4	69.3	62.4	67.3	65.5	60.4
P-1.5B	58.5	49.2	21.6	57.7	48.8	20.7
P-GPT	75.1	62.5	49.7	73.5	57.1	48.6
C-1.5B	82.7	78.8	67.6	81.2	75.1	64.0
C-GPT	**90.6**	**86.4**	**77.1**	**86.5**	**80.3**	**74.3**

Table 5: F1 scores across granularities on the Base-Supervised setting.

6 Conclusion

This paper introduces DynamicNER, a multilingual and multi-granular NER dataset optimized for LLM-based NER, including a human-annotated base version and a dynamic-categorized version. We develop the first dynamic categorization method in NER datasets for DynamicNER, enhancing its generalization while keeping data quality. We also propose CascadeNER, a powerful NER framework which is exceptionally suitable for lightweight LLMs and local deployment, outperforming current LLM-based methods. Moreover, we conduct comprehensive experiments and analyses on DynamicNER and discuss the advantage and future direction of LLM-based NER. More experiments and discussions are provided in Appendix E and H. We hope that DynamicNER and CascadeNER will facilitate future research in LLM-based NER, revitalizing this classical NLP task.

Limitations

There are still some challenges in our work. Although CascadeNER is designed to be able to accommodate nested and discontinuous NER, we only conduct evaluation on CascadeNER about flat NER tasks. This limitation arises from the fact that the models in CascadeNER are pre-trained on the dynamic version of DynamicNER, and DynamicNER is a flat NER dataset. Our resources are insufficient to collect enough open-source data for this purpose, which lead to DynamicNER containing only flat NER labels, and thus constraining CascadeNER to flat NER. Furthermore, Due to resource constraints and our failure to find annotators proficient in other languages for manual annotation, DynamicNER currently supports only 8 languages, which somewhat restricts its applicability.

Acknowledgements

This work is supported by the National Key R&D Program of China (Grant No. 2024YFC3308304), the "Pioneer" and "Leading Goose" R&D Program of Zhejiang (Grant no. 2025C01128), the National Natural Science Foundation of China (Grant No. 62476241), the Natural Science Foundation of Zhejiang Province, China (Grant No. LZ23F020008), the State Key Laboratory (SKL) of Biobased Transportation Fuel Technology, and the Zhejiang Zhejiang Key Laboratory of Medical Imaging Artificial Intelligence.

This work is supported in part by the NYUAD Center for Artificial Intelligence and Robotics, funded by Tamkeen under the NYUAD Research Institute Award CG010 and in part by the NYUAD Center for Interdisciplinary Data Science & AI (CIDSAI), funded by Tamkeen under the NYUAD Research Institute Award CG016.

Finally, we are grateful to the three anonymous reviewers for their insightful feedback, which has significantly improved the quality of this manuscript.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anthropic. 2023. Claude: An ai assistant by anthropic. https://www.anthropic.com/index/claude. Accessed: [date of access].

Dhananjay Ashok and Zachary C Lipton. 2023. Promptner: Prompting for named entity recognition. *arXiv preprint arXiv:2305.15444*.

Emily M Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

James R Curran and Stephen Clark. 2003. Language independent ner using a maximum entropy tagger. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 164–167.

Badhan Chandra Das, M Hadi Amini, and Yanzhao Wu. 2024. Security and privacy challenges of large language models: A survey. *ACM Computing Surveys*.

Jiangyi Deng, Xinfeng Li, Yanjiao Chen, Yijie Bai, Haiqin Weng, Yan Liu, Tao Wei, and Wenyuan Xu. 2025. RACONTEUR: A knowledgeable, insightful, and portable LLM-powered shell command explainer. In *Proceedings of the 32nd Annual Network and Distributed System Security Symposium*, NDSS.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the wnut2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Haitao Zheng, Zhiyuan Liu, Juanzi Li, Maosong Sun, and Jing Zhou. 2021. Few-NERD: A few-shot named entity recognition dataset. 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 3198–3213, Online. Association for Computational Linguistics.

George R Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *LREC*, pages 837–840.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2022. Dynamic gazetteer integration in multilingual models for cross-lingual and cross-domain named entity recognition. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2777–2790.

Mohammad Aqib Ganaie, Minmin Hu, M I Tanveer, and Ponnuthurai Nagaratnam Suganthan. 2022. Ensemble deep learning: A review. *arXiv preprint arXiv:2104.02395*.

Andrea Gasparetto, Matteo Marcuzzo, Alessandro Zangari, and Andrea Albarelli. 2022. A survey on text classification algorithms: From text to predictions. *Information*, 13(2):83.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. 2021. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92.

Corrado Gini. 1921. Measurement of inequality of incomes. *The economic journal*, 31(121):124–125.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR.

Dirk Hovy and Shannon L Spruit. 2016. The social impact of natural language processing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 591–598.

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

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. 2024a. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*.

Yan Hu, Qingyu Chen, Jingcheng Du, Xueqing Peng, Vipina Kuttichi Keloth, Xu Zuo, Yujia Zhou, Zehan Li, Xiaoqian Jiang, Zhiyong Lu, et al. 2024b. Improving large language models for clinical named entity recognition via prompt engineering. *Journal of the American Medical Informatics Association*, page ocad259.

HuggingFace. 2024. Open source ai year in review 2024. https://huggingface.co/spaces/huggingface/open-source-ai-year-in-review-2024.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Uri Katz, Matan Vetzler, Amir DN Cohen, and Yoav Goldberg. 2023. Neretrieve: Dataset for next generation named entity recognition and retrieval. *arXiv preprint arXiv:2310.14282*.

Vipina K Keloth, Yan Hu, Qianqian Xie, Xueqing Peng, Yan Wang, Andrew Zheng, Melih Selek, Kalpana Raja, Chih Hsuan Wei, Qiao Jin, et al. 2024. Advancing entity recognition in biomedicine via instruction tuning of large language models. *Bioinformatics*, 40(4):btae163.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. In *Bioinformatics*, volume 19, pages i180–i182. Oxford University Press.

Mingchen Li and Rui Zhang. 2023. How far is language model from 100% few-shot named entity recognition in medical domain. *arXiv preprint arXiv:2307.00186*.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2019a. A unified mrc framework for named entity recognition. *arXiv preprint arXiv:1910.11476*.

Xiaoya Li, Xiaofei Sun, Yuxian Meng, Junjun Liang, Fei Wu, and Jiwei Li. 2019b. Dice loss for data-imbalanced nlp tasks. *arXiv preprint arXiv:1911.02855*.

Xuezhe Li and Xiaodan Sun. 2020. Dice loss for data-imbalanced nlp tasks: Application to named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4653–4661.

Hongbin Liu, Ruixuan Xu, and Wei Xu. 2021. Cross-NER: Evaluating cross-domain named entity recognition. 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 4984–4995, Online. Association for Computational Linguistics.

Shervin Malmasi, Ning Zhang, Daniella Semedo, Ryan Ip, Aitor Gonzalez Aguirre, Leon Derczynski, and Isabelle Augenstein. 2022. MultiCoNER: A large-scale multilingual dataset for complex named entity recognition. In *Proceedings of the 13th Language Resources and Evaluation Conference*, pages 5102–5112. European Language Resources Association (ELRA).

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gemnet: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2023. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*.

Thien Huu Nguyen and Hung Le Cao. 2010. Nested named entity recognition using maximum entropy models. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, pages 2010–2018. ACL.

Rasha Obeidat, Xiaoli Fern, Hamed Shahbazi, and Prasad Tadepalli. 2019. Description-based zero-shot fine-grained entity typing. 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 807–814.

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

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. Association for Computational Linguistics.

Xiang Ren, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, and Jiawei Han. 2016. Label noise reduction in entity typing by heterogeneous partial-label embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1825–1834.

Prateek Sancheti, Kamalakar Karlapalem, and Kavita Vemuri. 2024. Llm driven web profile extraction for identical names. In *Companion Proceedings of the ACM on Web Conference 2024*, pages 1616–1625.

Wujun Shao, Yaohua Hu, Pengli Ji, Xiaoran Yan, Dongwei Fan, and Rui Zhang. 2023. Prompt-ner: Zero-shot named entity recognition in astronomy literature via large language models. *arXiv preprint arXiv:2310.17892*.

Yongliang Shen, Xinyin Ma, Zeqi Tan, Shuai Zhang, Wen Wang, and Weiming Lu. 2021. Locate and label: A two-stage identifier for nested named entity recognition. *arXiv preprint arXiv:2105.06804*.

The ModelScope Team. 2024. Swift:scalable lightweight infrastructure for fine-tuning. `https://github.com/modelscope/swift`.

Erik F. Tjong Kim Sang. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of the 6th Conference on Natural Language Learning (CoNLL-2002)*, pages 155–158.

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. Association for Computational Linguistics.

Neeraj Varshney and Chitta Baral. 2022. Model cascading: Towards jointly improving efficiency and accuracy of nlp systems. *arXiv preprint arXiv:2210.05528*.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. In *Linguistic Data Consortium, Philadelphia*.

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*.

Xinyu Wang, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Kewei Tu. 2020. Automated concatenation of embeddings for structured prediction. *arXiv preprint arXiv:2010.05006*.

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

Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Claudette Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. Ontonotes: A large training corpus for enhanced processing. In *Handbook of Natural Language Processing and Machine Translation*, pages 54–63. Springer.

Shuhui Wu, Yongliang Shen, Zeqi Tan, and Weiming Lu. 2022. Propose-and-refine: A two-stage set prediction network for nested named entity recognition. *arXiv preprint arXiv:2204.12732*.

Le Xiao, Yunfei Xu, and Jing Zhao. 2024. Llm-der: A named entity recognition method based on large language models for chinese coal chemical domain. *arXiv preprint arXiv:2409.10077*.

Tingyu Xie, Qi Li, Jian Zhang, Yan Zhang, Zuozhu Liu, and Hongwei Wang. 2023. Empirical study of zero-shot ner with chatgpt. *arXiv preprint arXiv:2310.10035*.

Hang Yan, Tao Gui, Junqi Dai, Qipeng Guo, Zheng Zhang, and Xipeng Qiu. 2021. A unified generative framework for various ner subtasks. *arXiv preprint arXiv:2106.01223*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. *arXiv preprint arXiv:2005.07150*.

Dawen Zhang, Pamela Finckenberg-Broman, Thong Hoang, Shidong Pan, Zhenchang Xing, Mark Staples, and Xiwei Xu. 2024. Right to be forgotten in the era of large language models: Implications, challenges, and solutions. *AI and Ethics*, pages 1–10.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.

A Dynamic Categorization Strategies

This section elaborates on the four strategies used to transform the manually annotated Base Version of DynamicNER into the final Dynamic Version. This entire process is an algorithm-driven pipeline that systematically modifies the entity type lists and corresponding labels for each data instance to create a more challenging and realistic benchmark. The pipeline consists of four sequential stages. We detail the goal, a concrete example, and the execution process for each strategy below.

Method 1: Mixing Categories of Different Granularities

- **Goal:** To test a method's robustness and generalization capability when faced with entity type lists of varying specificity. This strategy simulates the real-world variability where the required level of detail for an entity can change depending on the context.
- **Example:**
  - *Sentence:* "Ada Lovelace is considered the first computer programmer."
  - *Original State:* The entity "Ada Lovelace" is labeled as `Person` from the coarse-grained list, e.g., `[Person, Location, Organization, Product, ...]`.
  - *After Dynamic Change:* The system replaces the coarse-grained type with its fine-grained children. The new, shorter type list could be `[Artist, Athlete, Scholar, Location, Organization, Product, ...]`, and the required label for "Ada Lovelace" becomes `Scholar`.

Method 2: Replacing Types with Synonyms

- **Goal:** To evaluate a method's ability to understand semantic equivalence and not just rely on keyword matching between the entity context and the type list.
- **Example:**
  - *Sentence:* "Bekele won a gold medal in the competition."
  - *Original State:* The entity "Bekele" is labeled as `Athlete` from a type list such as `[..., Artist, Author, Athlete, ...]`.

– *After Dynamic Change:* The system replaces the type name `Athlete` with a valid synonym. The new type list becomes `[..., Artist, Author, Sportsperson, ...]`, and the required label for "Bekele" is now `Sportsperson`.

Method 3: Removing Irrelevant Categories

- **Goal:** To simulate a more realistic, focused NER task where the type list is context-specific, and to test the model's ability to avoid hallucinating labels that are not in the provided list.
- **Example:**
  – *Sentence:* "In the luxurious suite of The Ritz Carlton Hotel, a traveler is enjoying the beautiful view."
  – *Original State:* The entity is "The Ritz Carlton Hotel", and the type list contains all fine-grained types of Commercial Facility, including many irrelevant ones like `Restaurant` and `Bank`.
  – *After Dynamic Change:* The system prunes unrelated categories. The new type list is `[Person, Location, Hotel, Market/Mall, Theater/Cinema, Other Commercial Facility]`.

Method 4: Merging Types into Miscellaneous/Others

- **Goal:** To test the model's ability to correctly identify and classify infrequent or contextually less important entities into a general catch-all category, a common requirement in practical applications.
- **Example:**
  – *Sentence:* "It was organized by Harry West and constituted a formal electoral pact."
  – *Original State:* The entity "Harry West" is labeled as `Person` from the full type list.
  – *After Dynamic Change:* The new type list could be `[Location, Product, Group, Facility, Art, Miscellaneous]`, and the required label would become `Miscellaneous`.

Through this four-stage automated pipeline, DynamicNER generates diverse and challenging evaluation scenarios that go beyond static categorization, enabling a more thorough evaluation of LLMs' true generalization and contextual understanding capabilities.

B Categorization Metric Definition

Cohesion. Category cohesion score (cohesion) measures categorical semantic consistency by calculating the average semantic similarity between all entities within the same category. We employ the BERT-base (Devlin et al., 2018) model to extract semantic representations of entities, obtain embeddings of each entity, then computing cosine similarity between embeddings to derive cohesion. This metric ranges from [-1,1], where 1 indicates complete similarity and -1 indicates complete opposition. Typically, we perform category merging when cohesion exceeds 0.9. The formula is shown below:

$$Cohesion = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \cos(\mathbf{v}_i, \mathbf{v}_j) \tag{3}$$

where n is the number of entities in this category, $\mathbf{v}_i$ and $\mathbf{v}_j$ are the vector representations of the i-th and j-th entities encoded by BERT-base, and $\cos(\mathbf{v}_i, \mathbf{v}_j)$ represents the cosine similarity between two vectors.
The detailed cosine similarity formula is shown below.

$$\cos(\mathbf{v}_i, \mathbf{v}_j) = \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_i\| \|\mathbf{v}_j\|} \tag{4}$$

Normalized Entropy. Normalized entropy measures the overall balance of category distribution. This metric is used for the influence of category quantity by calculating the information entropy of category frequency distribution and normalizing it to a score within the range [0,1]. A score of 1 indicates perfect balance, where all categories have equal sample sizes, while 0 indicates complete imbalance, where all samples are concentrated in a single category. When normalized entropy falls below 0.8, it indicates significant distributional imbalance and needs to be adjusted. The formula is shown below:

$$H = -\frac{\sum_{i=1}^{n} p_i \log_2(p_i)}{\log_2(n)} \tag{5}$$

where n is the total number of categories, p_i is the proportion of samples in the i-th category calculated as the number of samples in category i

divided by the total number of samples across all categories.

Gini Coefficient. The Gini Coefficient (Gini, 1921) measures the degree of inequality in category distribution. Compared to normalized entropy, the Gini coefficient demonstrates higher sensitivity to distributional inequalities, performing better at identifying extreme imbalances where minority categories contain large sample proportions. For instance, when sample distributions exhibit extreme imbalances like [0.8, 0.1, 0.05, 0.05], the Gini coefficient provides stronger warning signals, while normalized entropy is more suitable for monitoring progressive imbalances such as [0.4, 0.3, 0.2, 0.1]. This metric also ranges from [0,1], where 0 indicates perfect balance and 1 indicates complete imbalance. A Gini coefficient exceeding 0.4 signals significant categorical inequality and requires distribution improvement. By using Gini coefficient and normalized entropy together, we achieve both sensitive detection of extreme imbalances and effective monitoring of overall distribution trends. The formula is shown below:

$$G = \frac{n + 1 - 2\sum_{i=1}^{n}(n - i + 1)p_i}{n} \tag{6}$$

where n is the total number of categories, p_i is the proportion of samples in the i-th category after sorting proportions in ascending order ($p_1 \leq p_2 \leq ... \leq p_n$).

Variation Coefficient. The Coefficient of Variation measures data dispersion by calculating the ratio of standard deviation to mean of category sample sizes. Its advantage is its scale independence, enabling comparisons across different scenarios. The coefficient ranges from 0 to positive infinity, where 0 indicates perfect balance and larger values indicate greater distributional imbalance. When the coefficient exceeds 0.5, it indicates significant fluctuation in sample sizes between categories, necessitating balance adjustments. The formula is shown below:

$$CV = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}}{\bar{x}} = \frac{\sigma}{\mu} \tag{7}$$

where n is the total number of categories, x_i is the number of samples in the i-th category, $\bar{x}$ is the mean number of samples across categories, σ is the standard deviation of sample numbers, and μ is the mean.

C Categorization Metric Selection

Mixing Types of Different Granularities. In this round of re-categorization, we use cohesion, normalized entropy, and Gini coefficient as metrics for optimization. Cohesion is employed to assess relationships between entity types, where close categorical relationships reduce the need for mixing to avoid creating unreasonable combinations. Meanwhile, normalized entropy and Gini coefficient are utilized to comprehensively measure distribution uniformity, where uneven distributions guide the system to perform additional merging for balance or category redistribution.

Replace with Synonyms. In this round of re-categorization, we use Gini coefficient and variation coefficient as metrics for optimization. We employ the variation coefficient to measure data dispersion, increasing synonym substitutions for increasing data convergence when dispersion is high. The Gini coefficient is used to guide system to reduce operations to prevent exacerbating imbalances when distributions are uneven. Cohesion is not used as synonym substitution does not alter hierarchical relationships between categories. Entropy is also given up because synonym substitution primarily focuses on linguistic variation rather than distributional changes.

Remove Irrelevant Types. In this round of re-categorization, we use cohesion and normalized entropy as metrics for optimization. We employ normalized entropy as a reference for controlling removal probability, ensuring that deletion operations do not result in overly concentrated distributions. Additionally, the system adjusts removal probability when cohesion is low, regulating relationships between categories. The variation coefficient is not used as this stage primarily focuses on option quantity rather than distribution characteristics, while the Gini coefficient is omitted since distribution balance has been addressed in the previous two stages, thus temporarily foregoing the Gini coefficient to prevent interference with other metrics.

Merge Types into Miscellaneous. In this round of re-categorization, we use all four metrics for optimization. As the final optimization stage, it requires consideration across all dimensions. We use all metrics for final fine-tuning to ensure overall data quality and avoid biases that might arise from single metric optimization.

D Detail of DynamicNER

The specific data volumes for each language are shown in Table 6. It is important to note that for languages except English and Chinese, we partially use manually translated English corpora. This is necessary to balance category distribution, as some languages lack sufficient corpora in specific domains. For the colloquial portion of the corpora, the Chinese part is sourced from Weibo, while the data for all other languages originated from X platform. We provide conversion scripts that allow DynamicNER to be transformed into train, dev, and test sets with non-overlapping subsets based on coarse categories, making it easier to use for traditional few-shot learning methods.

Language	# Sentences	# Tokens	# Entities	# Train	# Dev	# Test
English	1500	36.7k	4664	300	300	900
Chinese	1500	98.1k	5198	300	300	900
Spanish	1000	22.8k	2454	197	201	602
French	1000	24.1k	2763	200	200	600
German	1000	21.7k	2800	200	197	603
Japanese	1000	81.7k	3032	201	199	600
Korean	1000	66.4k	2401	202	200	598
Russian	1000	18.5k	2092	201	198	601

Table 6: Statistics of DynamicNER across languages. We roughly follow a 1:1:3 ratio to divide the train, dev, and test sets, with slight adjustments based on the proportional distribution of entities within the corpus.

Additionally, two points about the Table 6 require clarification. First, as DynamicNER's design emphasizes the evaluation of generalization and low-resource learning capabilities, we set the test set capacity to the biggest one, rather than the train set. Second, for Chinese, Japanese, and Korean, due to linguistic characteristics where each character is treated as a token, the token count appears significantly higher, though the actual corpus volume is comparable to other languages.

Although DynamicNER's data volume is significantly smaller than existing multilingual NER datasets, it aligns with our primary goal: evaluating LLM-based NER in fine-grained, low-resource, and dynamic scenarios. In such contexts, data volume may not be decisive. We acknowledge that smaller data volumes may lead to higher variance and overfitting. To show the robustness of our work, we conduct additional experiments on existing datasets with the model finetuned on DynamicNER. As shown in Appendix E, especially Table 7 & 8, CascadeNER, trained on DynamicNER, demonstrates strong few-shot performance on other established benchmarks. This indicates that the models have learned robust, generalizable features, avoiding overfitting.

E More Experiment about CascadeNER

E.1 CascadeNER Setting

In the experiments of this section, CascadeNER always employs two Qwen2.5-7B base models, which are fine-tuned separately based on the corresponding part of the dynamic version of DynamicNER to obtain an extractor and a classifier. Potential data contamination about the fine-tuning is discussed in Appendix N. We evaluate CascadeNER's performance in both few-shot and zero-shot scenarios, comparing it with supervised SOTAs and LLM-based baselines. For few-shot scenarios, the number of few-shot demonstrations is set to 3, the same as the experiments on DynamicNER.

E.2 Baselines

For supervised methods, we adopt ACE+document-context by Wang et al. (2020) (SOTA of CoNLL2003) and BERT-MRC+DSC by Li et al. (2019b) (SOTA of Ontonotes 5.0 (Pradhan et al., 2013)) for English datasets, while XLM-RoBERTa (Conneau et al., 2020) and GEMNET by Meng et al. (2021); Fetahu et al. (2022) (SOTA of MultiCoNER) for multilingual datasets. For LLM-based methods, we adopt GPT-NER and PromptNER with GPT-4o.

E.3 Dataset

Few-shot Data Sampling. In existing datasets, only CrossNER (Liu et al., 2021), designed for low-resource scenarios, and FewNERD (Ding et al., 2021), designed for few-shot scenarios, meet our requirements for evaluating CascadeNER in few-shot scenarios. However, relying solely on them is insufficient for comprehensively evaluating CascadeNER, particularly its multilingual NER performance. To address this, we develop a sampling algorithm to construct datasets for few-shot evaluation. Considering that basic random sampling cannot ensure a balanced category distribution, we employ a stratified sampling algorithm, which divides the dataset into strata based on the labels. Each stratum corresponds to a distinct entity type, and we ensure an relatively equal number of samples per category by drawing from these strata, thereby maintaining balance across categories in the results. The size for each stratum is calculated with the

Model	CoNLL2003	AI	Literature	Music	Politics	Science	FewNERD-8	FewNERD-66
XLM-RoBERTa	92.3	59.0	65.9	72.1	70.8	66.9	80.5	64.1
ACE+document-context	**94.6**	17.2	22.6	23.8	35.1	32.3	83.3	70.4
BERT-MRC+DSC	93.5	63.2	67.8	74.5	76.1	68.7	**86.7**	74.1
PromptNER	84.2	64.8	74.44	**84.2**	78.6	72.6	76.5	35.6
GPT-NER	73.5	58.0	61.2	60.8	62.4	55.8	70.0	58.4
CascadeNER (zero-shot)	88.2	68.9	71.7	79.3	80.5	73.6	73.4	67.0
CascadeNER (few-shot)	92.8	**75.8**	**75.2**	83.2	**82.4**	**77.1**	84.5	**75.9**

Table 7: F1 score of different models on CoNLL2003, CrossNER, and FewNERD.

Model	PAN-X								MultiCoNER					
	en	es	fr	ru	de	zh	ja	ko	en	es	ru	de	zh	ko
XLM-RoBERTa	88.1	86.5	85.4	86.3	83.1	78.3	75.6	82.0	58.9	54.8	55.9	60.6	62.6	52.0
GEMNET	90.5	**91.1**	**87.6**	**87.4**	**86.6**	81.5	80.8	**85.5**	84.3	**85.3**	78.7	**89.5**	83.2	**85.7**
PromptNER	81.7	79.6	73.5	73.8	71.9	72.1	70.8	73.5	79.5	75.6	76.5	67.6	70.8	72.4
GPT-NER	75.2	72.8	71.6	63.5	72.0	72.4	71.5	72.1	71.7	67.9	58.2	63.1	61.2	62.5
CascadeNER (zero-shot)	87.8	85.0	83.2	80.7	77.4	78.7	74.7	72.0	71.9	71.5	71.2	63.5	70.3	69.8
CascadeNER (few-shot)	**91.0**	85.2	87.2	86.8	82.8	**87.0**	**83.2**	79.4	**85.9**	81.1	**79.5**	69.1	**85.1**	76.9

Table 8: F1 score of different models across languages on PAN-X and MultiCoNER.

formula:

$$s_i = \min\left(\left\lfloor \frac{S}{m} \right\rfloor, n_i\right) \tag{8}$$

where N is the total number of labels in the dataset, S is the total sample size, n_i is the total number of labels with value i, m is the number of categories, and s_i is the number of labels from stratum i.

Dataset Selection. We conduct supplementary experiments on existing datasets including CoNLL2003 (Tjong Kim Sang and De Meulder, 2003), CrossNER (Liu et al., 2021), FewNERD (Ding et al., 2021), PAN-X (Pan et al., 2017), and MultiCoNER (Malmasi et al., 2022). Since we decide to use and share the formatted versions of these datasets in our repository to facilitate the test and use of CascadeNER, we only choose open-sourced datasets to avoid copyright issues. For evaluation metrics, we primarily use F1 score, as it is widely recognized as the most robust and effective metric for NER tasks (Li and Sun, 2020). We detail below the reasons for selecting these datasets and their usage.

CoNLL2003. CoNLL2003 is the most widely used English NER dataset, featuring four types: PER, LOC, ORG, and MISC. Supervised methods achieve excellent F1 scores of 90%-95% on this dataset. We use this dataset to compare CascadeNER and other LLM-based methods with existing supervised SOTAs in classical scenarios.

CrossNER. CrossNER is a English cross-domain dataset primarily used to evaluate a model's cross-domain generalization and low-resource performance. It consists of five independent sub-datasets, each covering a specific domain (AI, Literature, Music, Politics, and Sciences) and containing 9-17 entity types. Since the train set for the datasets only contains 100-200 sentences, supervised methods underperform compared to LLM-based methods. We use this dataset to evaluate CascadeNER in cross-domain and low-resource scenarios.

FewNERD. FewNERD is an English dataset designed to evaluate a model's ability to handle fine-grained entity recognition and few-shot learning, comprising 8 coarse-grained types and 66 fine-grained types. For supervised methods, FewNERD applies all 66 categories, challenging the models' classification abilities. For few-shot methods, we use the Intra-10way setting, where the train, dev, and test sets contain non-overlapping entity types. We utilize both the 8-category and 66-category settings to evaluate CascadeNER under varying levels of classification granularity.

MultiCoNER & PAN-X. MultiCoNER and PAN-X are two widely used multilingual datasets. MultiCoNER covers 6 entity types and 11 languages, while PAN-X includes 3 entity types and 282 languages. We use 6 and 8 overlapping languages from MultiCoNER and PAN-X with DynamicNER to evaluate CascadeNER's multilingual

capabilities. Notably, for the purpose of controlling variables, all methods requiring training are trained using multilingual joint training.

E.4 Experimental Results

As shown in Table 7 and 8, the results indicate that in low-resource scenarios, LLM-based methods achieve significantly better results. CascadeNER surpasses existing methods on CrossNER except Music and FewNERD, and PAN-X and MultiCoNER in some languages, achieving new SOTA performance and highlighting its exceptional generalization and capability to handle complex entity categorization. However, when handling NER tasks with ample training resources and simple classifications, LLM-based methods still lag behind existing methods, whether on the English-only CoNLL2003 or the multilingual PAN-X, indicating that supervised methods are still useful in some scenarios.

F Ablation Study

F.1 Result Fusion

In Section 4.2, we introduce our union strategy in result fusion to address the issue of extractor recall being significantly lower than precision, allowing multiple extractions for one sentence and taking the union of the results to maximize recall. For the problem of entity nesting, where different extraction rounds yield overlapping or nested entities, we adopt a length-first strategy, retaining the longer entity. Table 9 provides a example for the significantly low recall.

Dataset	Precision	Recall	F1 Score
CoNLL2003	98.4	93.6	95.9
AI	98.7	88.0	93.1
Literature	98.3	87.8	92.7
Music	98.0	92.0	94.9
Politics	97.5	90.0	93.6
Science	98.2	85.9	91.6

Table 9: Precision, recall, and F1 Score for CoNLL2003 and CrossNER. In this experiment, both base models used in CascadeNER are Qwen2.5-7B, and the results are obtained in zero-shot scenarios.

Figure 8 presents the impact of increasing the number of extraction repetitions in zero-shot scenarios on CoNLL2003. The results show that our strategy can slightly improve recall with minimal impact on precision. Given the obvious margin effect after 3 repetitions, we ultimately select 3 as the repetition count k for other experiments. It is important to emphasize that even without repetition, CascadeNER still has a significant performance advantage.

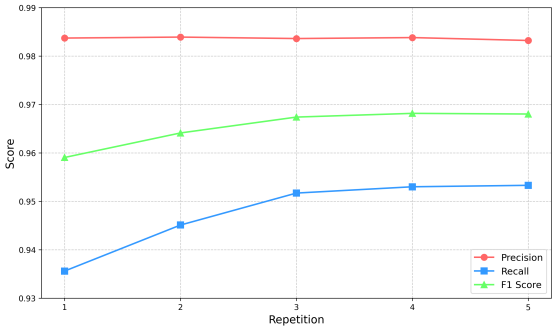

Figure 8: The curves showing visualized precision, recall, and F1 Score as a function of the number of repetitions, demonstrating how these metrics change with increasing repetition counts k. Both base models used in CascadeNER are Qwen2.5-7B.

F.2 Context in Classification

In the early stages of our research, the prompt used for classification contained only the entity itself without any context. Figure 9 provides an example comparing the two types of prompts. Although this method makes the prompt more concise, it lacks any contextual information. Our final in-context classification queries significantly improve classification accuracy, as shown in Table 10.

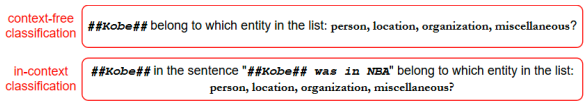

Figure 9: Example of the early context-free queries.

Dataset	ACC (context-free)	ACC (in-context)
CoNLL2003	90.1	94.2
AI	75.5	79.6
Literature	78.9	83.4
Music	84.6	88.3
Politics	87.4	90.8
Science	82.2	86.5

Table 10: Both base models used are Qwen2.5-7B. The results are obtained in zero-shot scenarios. The used datasets are CoNLL2003 and CrossNER. Accuracy is used in the evaluation of classifiers.

Model	Base-Supervised		Dynamic-Supervised		Dynamic-Fewshot	
	en	zh	en	zh	en	zh
G-1.5B	50.2 ± 1.3	44.1 ± 1.3	47.6 ± 1.5	41.2 ± 1.4	36.9 ± 1.6	35.8 ± 1.6
G-7B	55.1 ± 1.2	50.9 ± 1.2	52.3 ± 1.3	48.1 ± 1.3	42.7 ± 1.4	41.1 ± 1.4
G-GPT	62.4 ± 1.1	60.4 ± 1.2	60.6 ± 1.3	58.4 ± 1.3	49.2 ± 1.4	48.9 ± 1.3
P-1.5B	21.6 ± 1.6	20.7 ± 1.5	23.2 ± 1.7	22.7 ± 1.7	20.5 ± 2.0	19.9 ± 1.8
P-7B	41.1 ± 1.4	47.4 ± 1.4	44.3 ± 1.5	40.4 ± 1.5	39.8 ± 1.8	37.8 ± 1.8
P-GPT	49.7 ± 1.3	48.6 ± 1.3	53.0 ± 1.5	52.3 ± 1.4	49.4 ± 1.8	47.4 ± 1.9
C-1.5B	67.6 ± 1.1	64.0 ± 1.1	62.8 ± 1.2	58.9 ± 1.1	49.7 ± 1.3	48.5 ± 1.4
C-7B	73.8 ± 1.0	69.8 ± 1.0	68.2 ± 1.1	64.5 ± 1.1	55.7 ± 1.2	52.9 ± 1.2
C-GPT	77.1 ± 0.9	74.3 ± 0.9	73.1 ± 1.0	68.3 ± 1.0	61.3 ± 1.1	59.7 ± 1.1

Table 11: F1 Scores with 95% CI for English and Chinese. This table directly validates the statistical significance of our findings for representative languages. The consistently narrow CIs for CascadeNER highlight its high performance stability.

G Statistical Significance Analysis

To prove the statistical significance of our results on DynamicNER, we conduct an analysis to calculate 95% confidence intervals (CI) for our primary findings. This analysis validates whether the observed performance differences between methods, particularly the advantages of CascadeNER, are statistically meaningful.

We employ bootstrap resampling over the test set to compute the 95% CI for the F1 scores. For a test set of size N, we create a bootstrap sample by randomly drawing N instances with replacement. The F1 score is then calculated on this new sample. By repeating this process 500 times, we obtain an empirical distribution of F1 scores. The 95% confidence interval is then determined by taking the 2.5th and 97.5th percentiles of this distribution. This analysis is performed on English and Chinese as two representative languages. The results are presented in Table 11.

The results from our statistical analysis yield two key insights. First, the performance gains of CascadeNER over the baseline methods are confirmed to be statistically significant, as the confidence intervals show clear separation in most cases, particularly for the stronger GPT-4o backbone. Second, CascadeNER consistently exhibits the narrowest confidence intervals across almost all settings. This is especially evident when compared to PromptNER, whose complex prompts lead to higher performance variance. This indicates that by decomposing the NER task into two simpler sub-tasks, CascadeNER not only achieves higher accuracy but also offers a more stable and reliable performance.

H LLM-based Methods Comparsion

In this section, we compare our prompt with two existing LLM-based baselines, GPT-NER (Wang et al., 2023) and PromptNER (Ashok and Lipton, 2023). These methods are the currently main methods to achieve general NER with LLMs. A breif comparsion is shown in Figure 10.

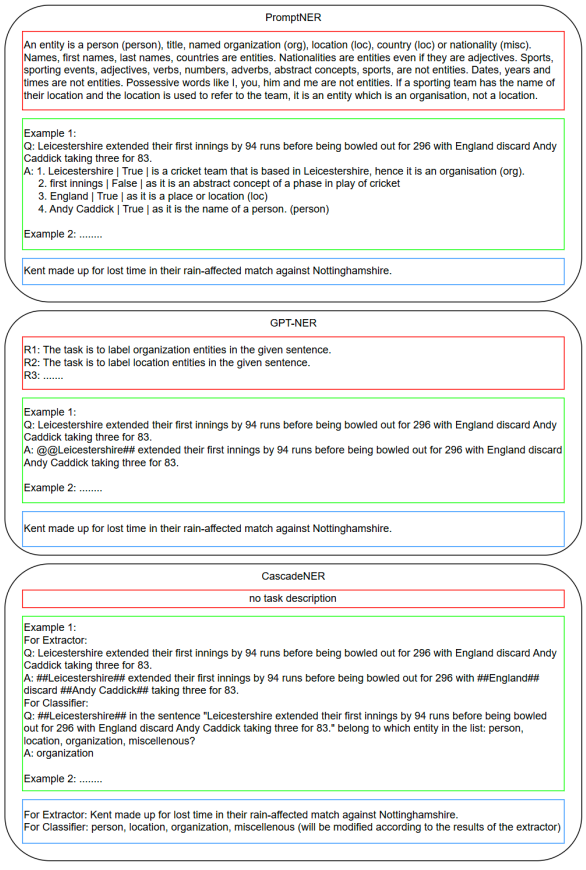

Figure 10: Examples for the three parts of the prompt for each method. The red boxes contain the task description, the green boxes contain few-shot demonstrations, and the blue boxes contain the input sentence.

PromptNER utilizes detailed descriptions of each entity's specific definition and CoT reasoning processes to fully leverage the LLM's logical reasoning abilities. However, like traditional methods, it treats NER as a sequence labeling task, failing to effectively utilize the LLM's global contextual understanding capabilities, making it prone to overlooking important context in complex sentences. Additionally, the task descriptions are overly complex, which not only makes it difficult for lightweight LLMs to correctly execute tasks, but also leads to a higher likelihood of hallucinations in tasks requiring fine-grained classification, such as the fine-grained settings of FewNERD and DynamicNER, as each category's definition requires descriptions. These issues reduce PromptNER's generalization and accuracy, limiting its application.

GPT-NER handles the NER task by determining whether a single entity belongs to a specific category, which leverages the generative capabilities of LLMs and allows for improved attention to the influence of context on entity meaning. Its drawback lies in the fact that it can only process one entity type at a time. This makes the method highly inefficient when dealing with fine-grained categorization, leading to significant resource consumption. Additionally, this method requires multiple judgments for the same entity, introducing the potential for conflicts between different rounds. Unfortunately, GPT-NER does not provide an effective solution for this issue.

CascadeNER divides the NER task into two subtasks: extraction and classification, while simplifying the input and output formats and reducing logical complexity. This ensures that even lightweight LLMs with limited capacity to handle complex tasks can still perform the tasks accurately and efficiently. In extraction, CascadeNER leverages the model's generation capabilities by producing sentences with identifiers, treating all entities uniformly, which enhances the model's generalization ability across different languages and domains. Notably, it avoids reliance on word order by consistently using "##" to mark entities, ensuring consistent annotation regardless of whether the language is right-to-left or left-to-right, improving cross-language consistency and adaptability. In classification, our method processes the entire sentence as a whole, better utilizing LLMs' strengths in contextual understanding and semantic modeling. By leveraging the LLM's ability to model long-range dependencies, the model's capacity to handle complex sentence structures is enhanced, avoiding fragmentation of information and improving overall consistency and generalization. However, our method also has limitations. The use of unified identifiers prevents CascadeNER from effectively handling nested NER. We plan to address this by developing a solution that accommodates both multilingual and nested NER tasks in future.

I Computational Resource Usage Record

In Table 12, we provide the API costs incurred when testing the complete dynamic version of DynamicNER in few-shot scenarios using three LLM-based methods with GPT-4o, serving as a reference for the computational resources required by these methods. The cost calculation follows OpenAI's official GPT-4o pricing, with input costs at 2.5 USD per 1M tokens and output costs at 10 USD per 1M tokens. The records show that CascadeNER exhibits significant advantages over existing methods in computational resource consumption.

	GPT-NER	PromptNER	CascadeNER
Cost (USD)	513.92	128.49	45.86

Table 12: Cost comparison of three LLM-based methods. The cost is calculated according to OpenAI's official GPT-4o pricing, not the actual cost.

J Human Annotation Process

J.1 Manual Annotation

For the annotators, each language in DynamicNER is annotated by two junior or higher-level students from the corresponding language departments at our universities. Each annotator receives specific training and follows DynamicNER's multi-granularity classification system to ensure consistent and accurate entity annotations across various languages and domains. The annotation process for each language are divided into two parts equally, with each annotator independently handling one part. After the initial annotation, the annotators revise their work based on the review results with the aid of Claude 3.5 Sonnet (Anthropic, 2023). For ambiguous terms or specialized domain terms, the annotators either collaborate with each other or consult experts via personal contact to ensure the accuracy and reliability of the annotation process.

K Inter-Annotator Agreement Analysis

To empirically validate the quality and semantic correctness of DynamicNER, we conduct a Inter-Annotator Agreement (IAA) analysis. We focus on the `Dynamic Version`, to ensure that the entity labels remain semantically robust and reliable even after undergoing the automated, metric-driven dynamic categorization process.

We select Cohen's Kappa (κ) as our evaluation metric, as it is a widely recognized statistical measure for evaluating the reliability of judgments between two independent annotators. The process includes two stages. Firstly, we apply our stratified sampling method (detailed in Appendix E) to randomly sample 200 sentences each for English, French, Japanese, and Chinese from the `Dynamic Version`. These languages are chosen as representative cases for the analysis. The sampled sentences are then given to two annotators who are not involved in the original dataset creation. They perform the annotation in a double-blind setting, where neither annotator was aware of the other's decisions or the original labels. The results of our IAA analysis are presented in Table 13.

Table 13: Cohen's Kappa (κ) scores from the IAA analysis conducted on the dynamic version of DynamicNER.

Language	Cohen's Kappa (κ)
English	0.86
French	0.82
Chinese	0.83
Japanese	0.81

According to established interpretations of the Kappa statistic, the obtained scores indicate substantial agreement. This strong quantitative evidence, derived from a rigorous human-centric audit, directly confirms that the entity labels in our dataset are semantically robust and reliable. It serves as a crucial validation that our automated quality-control pipeline successfully preserves high-quality annotations, thereby addressing the concern about the semantic correctness of the dynamically generated labels.

L Finetuning Hyperparameter

To ensure the reproducibility of our experiments, this section provides a detailed record of the hyperparameter configurations used for fine-tuning the lightweight LLMs in CascadeNER.

Our implementation is based on the well-regarded MS-SWIFT framework (Team, 2024), and we employ Low-Rank Adaptation (LoRa) (Hu et al., 2022) for Parameter-Efficient Fine-Tuning (PEFT) (Houlsby et al., 2019). All fine-tuning and inference experiments are conducted on a server equipped with 8 NVIDIA RTX 3090 GPUs with 24GB VRAM for each GPU.

Hyperparameter	Value
LoRA Parameters	
Target Modules	ALL
Rank (r)	8
Alpha (α)	32
Dropout	0.05
Training Parameters	
Learning Rate	1e-4
Optimizer	AdamW
Weight Decay	0.01
Max Sequence Length	4096
Training Epochs	Max 50
- Best Epoch (1.5B model)	25
- Best Epoch (7B model)	20
Effective Batch Size	32

Table 14: Hyperparameters used for fine-tuning CascadeNER models. The 'ALL' setting for Target Modules indicates that LoRA was automatically applied to all suitable linear layers within the model.

The specific parameters for LoRA and the training process are detailed in Table 14. We maintain a consistent effective batch size of 32 for both models to ensure a fair comparison. For the 1.5B model, we use 4 GPUs with a per-device batch size of 4 and 2 gradient accumulation steps. For the 7B model, we use 8 GPUs with a per-device batch size of 1 and 4 gradient accumulation steps. We train each model for a maximum of 50 epochs, saving a checkpoint every 5 epochs and using an early stopping strategy based on the validation set performance. The best-performing checkpoints are then selected for the final evaluation.

M Prompt Example for CascadeNER

In this section, we present the examples of prompts used in CascadeNER to provide an intuitive demonstration of the method and to facilitate reproducibility. The example is provided in Figure 11. We provide complete prompts and corresponding json files for training in our github repository.

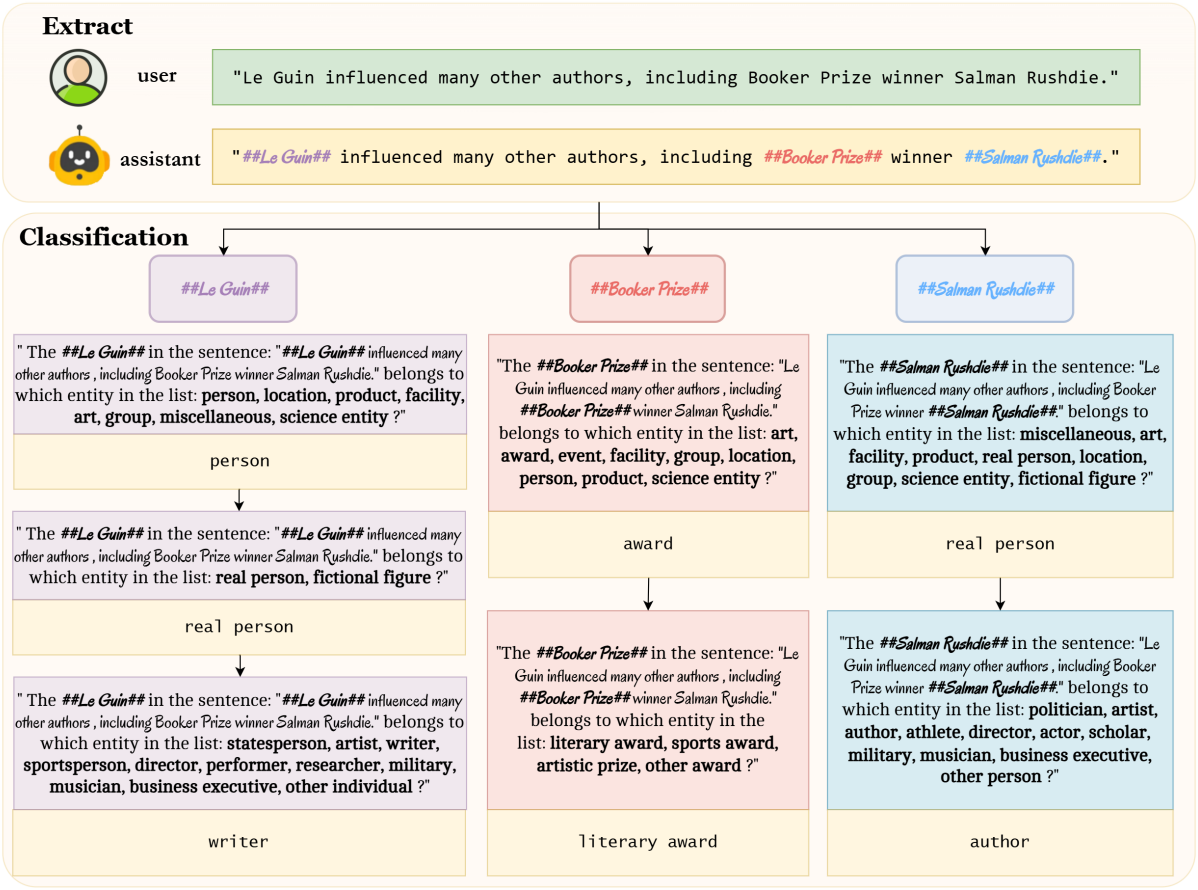

Figure 11: Examples for the complete prompts of CascadeNER.

N Data Contamination Statement

Given that LLMs are trained on data from diverse and complex sources, there is a possibility that portions of the evaluation sets may have been encountered during pre-training. However, as prior research (Chowdhery et al., 2023) indicates, contaminated data that has been seen during training does not significantly influence performance. Thus, we consider this issue negligible.

In additional experiments on CascadeNER, we notice another critical data contamination concern: potential corpus overlap between DynamicNER and other benchmark datasets utilizing Wikipedia-derived text, which can reduce evaluation fairness. To mitigate this risk, we implement a rigorous filtering protocol during DynamicNER's annotation phase. After completing the initial manual annotation of the base version, we employ Sentence-BERT to compute semantic cosine similarity between each candidate sentence and existing sentences in reference datasets. Sentences exhibiting similarity scores exceeding 0.8 are excluded from the corpus. New sentences from collected corpus meeting the similarity criteria are then re-annotated following the original annotation workflow. This iterative process continues until all sentences in the base version satisfy the similarity constraints. After this we utilize dynamic categorization to generates the dynamic version. This procedure ensures the reliability and fairness of our test results.

O Ethical Statement

When constructing DynamicNER, we strictly adhere to existing ethical guidelines (Bender and Friedman, 2018; Gebru et al., 2021; Hovy and Spruit, 2016), ensuring that our data sources and processing methods comply with legal and ethical standards while maintaining high-quality annotations. All the text in DynamicNER is sourced from Wikipedia, ensuring no violations of privacy or copyright, as Wikipedia is an open-source platform with user-contributed content from around the world. During data collection and annotation, we balance category distribution to minimize the risk of bias in the model. Furthermore, we maintain transparency by detailing the dataset development

process and data partitioning in this paper, ensuring clarity and reproducibility for future research.
In our writing, we use ChatGPT-4o (Achiam et al., 2023) and Claude 3.5 Sonnet (Anthropic, 2023) for assistance.

P Detailed Categories of DynamicNER

P.1 Person

Real Person Politician, Artist, Author, Athlete, Director, Actor, Scholar, Military, Musician, Business Executive, Other Person.

Fictional Figure Mythological Figure, Other Figure.

P.2 Location

Geographical Entity Water Body, Mountain, Island, Desert, Other Geographical Entity.

Geo-Political Entity Continent, Country, State or Province, City, District, Region, Other GPE.

Address Address, Road, Railway, Other Address.

P.3 Product

Food Beverages, Packaged Foods, Other Food.

Weapon Firearms, Biological, Chemical Weapon, Explosives, Cold Weapon, Nuclear, Other Weapon.

Technology Software, Website, Electronics, AI, Other Technology.

Vehicle Air, Car, Water, Rail, Bike, Other Vehicle.

Other Product Clothes, Household, Personal Care, Toys, Musical Instruments, Other Product.

P.4 Facility

Public Facility Hospital, Library, Park, Landmark, School, Museum, Sports Facility, Other Public Facility.

Commercial Facility Hotel, Restaurant, Market/-Mall, Theater/Cinema, Bank, Other Commercial Facility.

Transportation Facility Airport, Station, Port, Other Transportation Facility.

Production Facility Factory, Farm, Mine, Energy, Other Production Facility.

Other Facility Residential, Government Facility, Other Facility.

P.5 Art

Visual Art Painting, Sculpture, Visual Art Genre, Other Visual Art.

Music Song, Album, Music Genre, Other Music.

Literature Poem, Non-fiction, Fiction, Literature Genre, Other Literature.

Other Art Film, Play, Broadcast Program, Game, Other Art.

P.6 Group

Social Group Ethnic Group, Religious Group, Other Social Group.

Non-commercial Organization Educational and Research, Political/Military, Community, Religious Organization, Other Non-commercial Organization.

Commercial Organization Sports Team, Band, Company, Media, Other Commercial Organization.

P.7 Miscellaneous

Award Literary Award, Sports Award, Artistic Award, Other Award.

Event Political/Military Event, Sporting Event, Disaster, Business Event, Other Event.

Miscellaneous Educational Degree, Tradition, God, Law, Language, Miscellaneous.

P.8 Science Entity

Biological Protein, Species, Biological Theory, Other Biological Entity.

Chemical Element, Compound, Reaction, Chemical Theory, Other Chemical Entity.

Physical Physical Phenomenon, Astronomical Object, Physical Theory, Other Physical Entity.

Computer Science ProgramLang, Algorithm, Other Computer Science Entity.

Medical Disease, Injury, Medication, Symptom, Medical Theory, Other Medical Entity.

Other Scientific Entity Discipline, Academic Journal, Conference, Metrics, Other Scientific Entity.

Q More Categorization Quality Evaluation

In this section, we display the quantitative results of categorization metrics in Spanish, French, Russian, German, Japanese, and Korean. The results in shown in Figure 12. Experimental results demonstrate that our dynamic categorization method maintains or improves dataset quality compared to the base version in all languages.

Figure 12: Quantitative categorization metric results for 3 versions DynamicNER.