LiP-NER: Literal Patterns Benefit LLM-Based NER

Ruiqi Li and **Li Chen***
College of Computer Science, Sichuan University
ruiqi_li@stu.scu.edu.cn, cl@scu.edu.cn

Abstract

Large Language Models (LLMs) can enhance the performance of Named Entity Recognition (NER) tasks by leveraging external knowledge through in-context learning. When it comes to entity-type-related external knowledge, existing methods mainly provide LLMs with semantic information such as the definition and annotation guidelines of an entity type, leaving the effect of orthographic or morphological information on LLM-based NER unexplored. Besides, it is non-trivial to obtain literal patterns written in natural language to serve LLMs. In this work, we propose LiP-NER, an LLM-based NER framework that utilizes **Li**teral **P**atterns (LiP), the entity-type-related knowledge that directly describes the orthographic and morphological features of entities. We also propose an LLM-based method to automatically acquire literal patterns, which requires only several sample entities rather than any annotation example, thus further reducing human labor. Our extensive experiments suggest that literal patterns can enhance the performance of LLMs in NER tasks. In further analysis, we found that entity types with relatively standardized naming conventions but limited world knowledge in LLMs, as well as entity types with broad and ambiguous names or definitions yet low internal variation among entities, benefit most from our approach. We found that the most effective written literal patterns are (1) detailed in classification, (2) focused on majority cases rather than minorities, and (3) explicit about obvious literal features.

1 Introduction

Named Entity Recognition (NER) seeks to recognize and classify named entities in unstructured text, and is an essential component in numerous natural language processing (NLP) applications

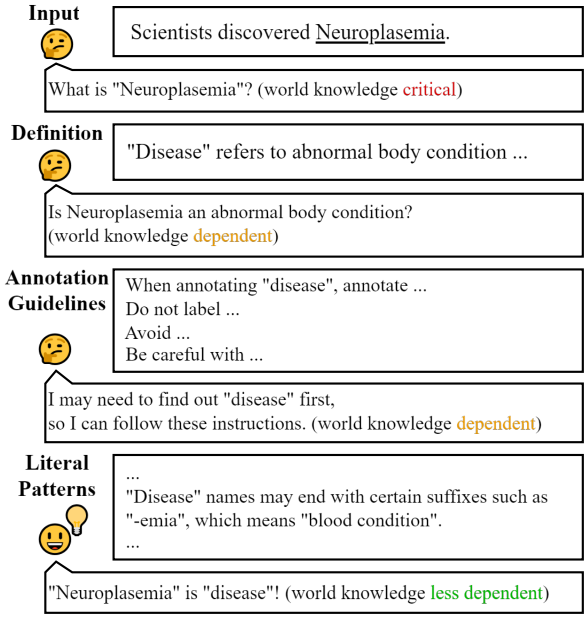

Figure 1: An illustration of the concept of LiP-NER. Literal Patterns (LiP) provide direct description about the appearance of the entities in a certain type, reducing the dependence on world knowledge of LLMs.

such as question-answering (Molla et al., 2006), information retrieval (Weston et al., 2019) and so on. Initially, NER systems were built with traditional approaches like rule-based (Borkowski and Watson, 1967) and feature-engineering-based (Zhou and Su, 2002). With the release of transformer-based (Vaswani et al., 2017) pre-trained language models, a new paradigm of NER has been established with BERT (Devlin et al., 2019) and models alike (Wu et al., 2021), which eliminates the burden of training a model from scratch.

Recently, generative large language models (LLMs) such as ChatGPT (OpenAI, 2023) have shown outstanding performance among various fields of NLP (Min et al., 2023; Zhao et al., 2023). Prompt engineering, including careful prompt design and extra information provision, has emerged as an economical way to make further improve-

*Corresponding author.

ment of LLMs over downstream tasks at test-time (Peng et al., 2023).

When it comes to NER, the initial capabilities of LLMs are not as promising (Jimenez Gutierrez et al., 2022). One reason is that LLMs rely on their world knowledge, which is learned during pre-training stage, to process tasks. Thus, in domains that have less textual resources about the entities and the types available for pre-training, the vanilla performance of LLMs will be less impressive. Injecting external knowledge related to the type of entities could help, as the models know more details about the type they are annotating (Seyler et al., 2018). Recent works mainly utilize the definition and the annotation guidelines of an entity type (Sainz et al., 2024; Zamai et al., 2024). As is depicted in Figure 1, a definition is a semantic description of an entity type, whereas annotation guidelines mainly contain edge case clarification, and are offered in a way that is reminiscent of human annotators. Both types of information offer more semantic details about the concept of an entity type, but still rely on the world knowledge of the connection between the entity and these semantic information.

Historically, literal feature information has played an essential role in NER task (McDonald, 1993), for its direct description on orthographic and morphological patterns of an entity type, and does not depend on semantic knowledge. However, to utilize such information in LLM-based NER systems, it shall be described in natural language, which is not trivial as it involves expert labor. Besides, documents of literal features are scarce on Internet, making it difficult to utilize such information via retrieval-augmented generation (RAG) strategies(Gao et al., 2023).

In this paper, we introduce LiP-NER, a method of LLM-based NER utilizing **Li**teral **P**atterns (LiP) written in natural language. Literal patterns are external knowledge that directly describe the literal features of an entity type, which can be expected that have less requirement on world knowledge than semantic external knowledge. We also propose an LLM-based method to automatically acquire literal patterns of an entity type. Instead of the requirement of several annotation examples (Zamai et al., 2024), our method needs only a list of sample entities. It gets rid of human annotation, thus further reducing labor requirements. Our experiments demonstrate the effectiveness of LiP-NER across different LLMs. Furthermore, our analysis provides preliminary insights into the entity types that benefit from our method and the key characteristics of suitable literal patterns for LLM-based NER tasks.

In summary, our contributions are threefold:

1. We proposed LiP-NER, an LLM-based NER framework that utilizes literal patterns as entity-type-related external knowledge, with less dependency on world knowledge within LLMs.

2. We also proposed an LLM-based method to automate the acquisition of the literal patterns of an entity type. It requires only a list of sample entities rather than any annotation example, thus further reducing labor requirement without a sacrifice in performance.

3. Through extensive experiments, we demonstrated the effectiveness of LiP-NER in LLM-based NER. Our analysis provides preliminary insights into the entity types that benefit from our method and the key characteristics of suitable literal patterns for LLM-based NER.

2 Related Work

2.1 Named Entity Recognition

Initially, NER systems were built with rule-based (Borkowski and Watson, 1967) approaches. Starting from the era of feature-engineering-based (Zhou and Su, 2002) approaches, NER is framed as a sequence labeling task, which aims to assign an entity label in BIO format to each token in a given sentence (Tjong Kim Sang and De Meulder, 2003). Recent well-established approaches include BiLSTM-CRF methods (Lample et al., 2016) and fine-tuning BERT-based models (Devlin et al., 2019). These supervised models have shown excellent performance, but they are difficult to generalize to other domains (Gururangan et al., 2020). In addition, in specific domains, the scarcity of labeled data has been a long-lasting challenge, making it difficult to train models on these domains (Hedderich et al., 2021).

2.2 LLM-Based NER

In recent years, generative LLMs have demonstrated impressive generalization capabilities across various challenging tasks (Hegselmann et al., 2023; Robinson and Wingate, 2023; Hendy et al., 2023), inspiring a series of studies that attempt

to reframe NER tasks into a generative format. For instance, Wang et al. (2023) proposed GPT-NER, which effectively transforms the NER task from sequence-labeling to text-generation with some special tokens involved. Li et al. (2023) proposed CodeIE, which utilizes code generator LLMs and formulates the NER task into a code generation task. However, efforts of applying generative LLMs to NER have been less promising, lagging far behind supervised methods (Jimenez Gutierrez et al., 2022; Hu et al., 2024).

2.3 External Knowledge for LLM-Based NER

Seyler et al. (2018) have demonstrated that the provision of external knowledge benefits in NER. Recent methods take full advantage of external knowledge via prompt-based augmentation of LLMs.

When it comes to entity-type-related knowledge, an intuitive idea is the definition of a type. Prompt-NER (Ashok and Lipton, 2023) utilizes definitions and annotated examples as external knowledge, with a prompt that instruct LLM to perform self-correction via justifying the entries in its potential entity list. Zhou et al. (2024) proposed Universal-NER and tried to replace the type name with a short description of the type but with no gain. Mimic human annotators, GoLLIE (Sainz et al., 2024) and SLIMER (Zamai et al., 2024) applied annotation guidelines in code- and natural-language-LLM-based NER, respectively. Hu et al. (2024) applied annotation guidelines with additional instructions based on error analysis in LLM-based clinical NER tasks and observed constant improvement over vanilla performance.

Both definition and annotation guidelines provide more semantic details about an entity type, but still rely on world knowledge of the connection between the entity and the knowledge, which is learned by LLMs during the pretraining stage.

3 LiP-NER

3.1 Literal Patterns

The motivation of this work is to provide LLMs with type-related knowledge that is less semantic and directly describes the superficial traits of potential entity names, so that the LLMs can process NER tasks with less dependence on the world knowledge within the models.

In rule-based and feature-engineering–based NER systems, researchers often exploit characteristics inherent to the entity names, such as morphological characteristics, including affixes and keywords, and orthographic characteristics, including initial capitalization or all-caps, alphanumeric sequence structures, the use of punctuations (e.g., hyphens and delimiters) and so on. These features are either hand-crafted by experts or automatically extracted from large-scale gazetteers, and the resulting patterns are employed in NER systems as decision rules, regular expressions, or dimensions of feature vectors.

For LLMs, external knowledge is injected by writing it directly into prompts in natural language. In this paper, we define **Li**teral **P**atterns (LiP) as a list of literal features written in natural language. This list typically includes the orthographic and morphological properties of a given entity type: common affixes, keywords, capitalization conventions, alphanumeric patterns, punctuation usage, and so on. In our method, these features are discovered from a relatively small list of sample entities by LLMs. Hence, we refer to them as "patterns".

Prompt Template for the Generation of Literal Patterns
[EXAMPLES] {List of Sample Entities} [TASK] Make a brief summary on the nomenclature (orthography, or naming routine) of type '{Type Name}', based on the examples given above. reply in an unsequenced list.
Example
INPUT: [EXAMPLES] TNF alpha mRNA; EBV mRNA transcripts; exon 23; RXR-beta mRNA; IL-8 mRNA; G2715T; interleukin-1 beta mRNA; spi-1 mRNA; BRLF1; CTCF mRNA [TASK] Make a brief summary on the nomenclature (orthography, or naming routine) of type 'RNA', based on the examples given above. reply in an unsequenced list.
OUTPUT: RNA is typically preceded by the gene or protein name (e.g., TNF alpha, EBV, IL-8, interleukin-1 beta, CTCF). The word "mRNA" is commonly used to specify the messenger RNA form of the gene or protein, indicating the transcript of the gene (e.g., TNF alpha mRNA, IL-8 mRNA). Specific exon or mutation numbers may be included to specify certain variants or sequences (e.g., exon 23, G2715T). Gene names or acronyms are often presented in capital letters, with a hyphen sometimes used for isoform identifiers (e.g., RXR-beta). The names of genes or proteins are often written in their full form or as abbreviations (e.g., BRLF1, SPI-1). Gene names are typically not italicized, but the mRNA designation is appended as "mRNA" to clarify the specific molecule.

Figure 2: The prompt template used to query LLMs for the generation of literal patterns, which includes a list of sample entities and a generation instruction. The term "nomenclature" was used in experiments but is deprecated in this paper, due to its inaccuracy-while nomenclature refers to a system of naming, the resource generated in this way is more like a list of patterns.

3.2 Acquire Literal Patterns via LLMs

Although literal patterns are useful resources, it is not trivial to obtain them. To write literal patterns in natural language, expert labor is required. Especially for the entity types with more diversity in entity names, it's nearly impossible to exhaust the nuances.

To overcome this limitation, we exploited ChatGPT (OpenAI, 2023) to generate literal patterns. Being different from the method of generating annotation guidelines (Zamai et al., 2024), which utilizes manually labeled annotation examples, generating literal patterns requires only a small list of sample entities. In particular, we designed a zero-shot prompt template shown in Figure 2 to query LLMs. In this template, we provide a small list of sample entities to prompt the LLM to generate literal patterns in a list.

3.3 Case Study

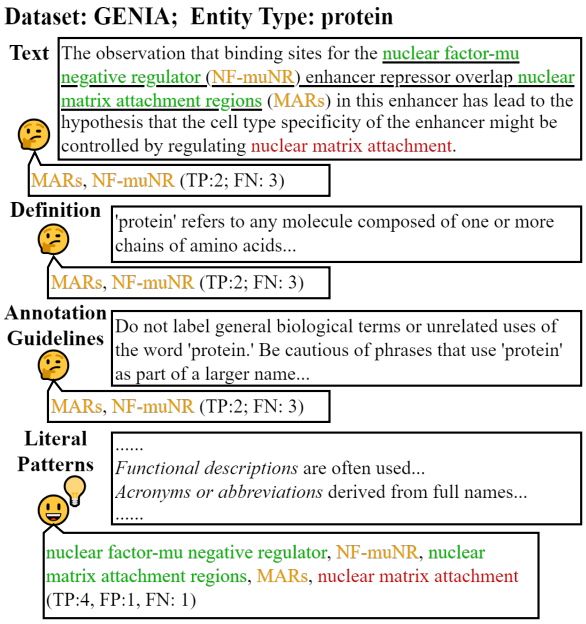

Figure 3: Case study example. The golden and green entities are correct labels, while the red one is wrong. The underline in the text labels a nested long entity, which is missed in all configurations.

Figure 3 shows an case study example. This is an example from GENIA dataset, labeling *protein* entities, tested on LLAMA-3-8B-INSTRUCT with 4 configurations: vanilla, with definition, with annotation guidelines, and with literal patterns. The full texts of external knowledge used in this example are listed in Appendix B.

The vanilla model labels 2 correct entities, both are abbreviations. The model may have some world knowledge about these two mentions, or the model learned that proteins often appear in text as abbreviations or code names, so it labels all abbreviations in this text, which are two correct labels.

Providing a definition of protein, the performance stays still. Although the definition enriches the meaning of protein, offers more semantic information to the context, it fails to provide more clue for the LLM to label. Providing annotation guidelines, the performance does not change. Annotation guidelines offer several regulations and notices, which may help refining the borders of labels or filtering out potential false labels, but in this case, there is no false label to be refined or filtered out.

Providing literal patterns, two additional entities are correctly labeled, while one incorrect label is introduced. With literal patterns, the model learns what entities of a certain type may look like, and follows the provided patterns to label. In this case, the model learned that protein entities may appear as functional descriptions and abbreviations, so it labeled 3 more mentions that involve functional descriptions, which were 2 correct labels and 1 wrong label.

4 Experiments

In the experiments, we comprehensively investigated the effect of literal patterns on low resource LLM-based NER tasks. All experiments were conducted on original models without any fine-tuning. Our research questions include:

- **RQ1**: Can LiP-NER help LLMs to process NER?
- **RQ2**: What kinds of entity types are more likely to benefit from LiP-NER?
- **RQ3**: What is a helpful list of literal patterns?

4.1 Datasets & Metrics

We conducted experiments on six publicly accessible datasets, including:

MIT dataset series (Liu et al., 2013) is a widely-used benchmark for zero-shot NER, which consists of three datasets: restaurant, movie, and movie-trivia. **MIT-restaurant** contains queries about restaurants with 8 entity types. **MIT-movie** are those about movies and **MIT-movie-trivia** contains more complex queries, each of them has 12 entity types.

CoNLL-2003 (Tjong Kim Sang and De Meulder, 2003) is a famous dataset in news domain, which has 4 entity types including *person, organization, location* and *miscellaneous*.

GENIA (Kim et al., 2003) is a dataset in biomedical domain. We follow Collier et al. (2004) to simplify GENIA into 5 entity types including *DNA, RNA, cell_line, cell_type* and *protein*.

BC5CDR (Li et al., 2016) is another dataset in biomedical domain, including 2 entity types: *chemical* and *disease*.

We followed the official splits of training, development and test sets of these datasets. We merged training and development sets for the extraction of annotation examples or sample entities for the generation of the definitions, guidelines and literal patterns, and test these knowledge on the test sets.

During evaluation, we processed deduplication on both the model predictions and the ground truth. We filtered out the pure hallucination predictions (i.e. predicted entities that were not in the target text) before evaluation, as these predictions would not introduce false annotation in the text. We performed strict matching in evaluation, where a predicted entity was considered correct only if both its boundaries and type exactly matched those of the corresponding ground-truth entity.

We report micro-precision (P), recall (R) and F1 scores in our results, where all entity types are treated equally.

4.2 Models

We conducted our experiments on two open-source LLMs, META-LLAMA-3-8B-INSTRUCT (Grattafiori et al., 2024) and QWEN2.5-7B-INSTRUCT (Yang et al., 2024). These instruct-tuned models could follow natural language instructs and provide outputs in JSON format, which helped post-processing. We ran these models locally without fine-tuning. Greedy decoding (i.e., *do_sample = false*) was applied and the seeds were fixed for reproducible generation. Our inference template is listed in Appendix A.

4.3 Baselines

We compare our method with aforementioned commonly used entity-type-related external knowledge, including definition and annotation guidelines.

To generate definition and guidelines, following SLIMER (Zamai et al., 2024), for each entity type of each dataset, we extracted 3 annotation examples from the train&dev set and utilized the 1-shot prompt template reported in the original paper to prompt OpenAI's GPT-4O-MINI. To Briefly introduce the template, it contains a fixed demonstration, including 3 annotation examples and a pair of manually written definition and guidelines of a type, an instruction saying *Now do the same for the Named Entity: type_name. Examples:*, and the 3 annotation examples extracted from the train&dev set.

We examined LLMs' capabilities under the circumstances of without any external knowledge (vanilla), with the definition (marked as *w/ Definition*) and annotation guidelines (*w/ Guidelines*) respectively, and with the combination of these two kinds of information (*w/ Def&Guide*).

4.4 LiP-NER

We utilized the proposed zero-shot prompt template to acquire literal patterns. For each entity type, we extracted 10 sample entities from the train&dev set to prompt OpenAI's GPT-4O-MINI to generate literal patterns. We added generated literal patterns into aforementioned four baseline circumstances and compared the results (marked as + *LiP*) with the baselines.

5 Results

5.1 Effectiveness of LiP-NER (RQ1)

From the results in Table 1, we have the following observations:

(1) Comparison with vanilla abilities Comparing the vanilla capability of each model (row 1 of each model) with the augmentation of literal patterns (row 2), on both models, injecting literal patterns yields better F1-scores. On LLAMA-3-8B-INSTRUCT, precision rates consistently increase, and recall rates improve on every dataset except a small decrease on CoNLL-2003, as a trade-off for precision rates. On QWEN-2.5-7B-INSTRUCT, all precision scores rise, and recall improves on all datasets except MIT-movie-trivia and GENIA, as a trade-off for precision rates.

(2) Comparison with other knowledge Comparing literal patterns (row 2 of each model) with definition (row 3) and annotation guidelines (row 5) under the circumstances where only one kind of knowledge is injected, literal patterns reach more

Prompt	Dataset (Metrics: Micro-P, R, F1 percentages)					
	restaurant	MIT movie	movie-trivia	CoNLL-2003	GENIA	BC5CDR
META-LLAMA-3-8B-INSTRUCT						
Vanilla	26.1 55.4 35.5	24.6 68.9 36.2	18.5 56.0 27.8	23.6 84.3 36.9	25.6 56.0 35.1	60.0 66.8 63.2
+ LiP	28.0 59.3 38.0	26.2 72.4 38.4	23.9 56.8 33.7	36.8 82.8 51.0	28.1 57.7 37.8	73.5 68.1 70.7
(Δ F1)	↑ 2.5	↑ 2.2	↑ 5.9	↑ 14.1	↑ 2.7	↑ 7.5
w/ Definition	25.7 59.9 36.0	26.2 71.9 38.4	19.5 58.1 29.2	26.3 85.2 40.2	32.6 54.4 40.8	64.2 71.5 67.6
+ LiP	29.6 60.1 39.6	26.6 72.1 38.9	22.7 59.1 32.8	33.6 85.3 48.3	32.1 58.1 41.3	70.2 70.5 70.4
(Δ F1)	↑ 3.6	↑ 0.5	↑ 3.6	↑ 8.1	↑ 0.5	↑ 2.8
w/ Guidelines	29.5 51.7 37.5	30.2 67.1 41.7	22.7 59.4 32.9	31.5 87.6 46.3	31.5 51.1 39.0	67.9 65.4 66.6
+ LiP	31.1 53.1 39.2	30.5 70.6 42.6	25.3 59.9 35.6	34.0 85.9 48.7	29.1 55.8 38.3	72.7 62.6 67.3
(Δ F1)	↑ 1.7	↑ 0.9	↑ 2.7	↑ 2.4	↓ 0.7	↑ 0.7
w/ Def&guide	29.6 55.6 38.7	28.1 68.5 39.9	20.5 58.9 30.4	30.0 87.2 44.6	38.3 52.0 44.1	69.1 66.5 67.8
+ LiP	30.5 58.3 40.0	28.7 70.5 40.7	21.7 60.0 31.9	30.8 87.1 45.5	34.2 58.2 43.1	69.2 66.0 67.6
(Δ F1)	↑ 1.3	↑ 0.8	↑ 1.5	↑ 0.9	↓ 1.0	↓ 0.2
QWEN2.5-7B-INSTRUCT						
Vanilla	33.0 37.2 35.0	36.9 58.6 45.3	24.2 53.4 33.3	41.7 66.4 51.2	46.2 30.7 36.9	77.6 52.1 62.4
+ LiP	38.6 44.0 41.1	44.1 62.9 51.8	29.0 52.3 37.3	42.0 72.1 53.1	52.8 29.3 37.7	77.8 52.9 63.0
(Δ F1)	↑ 6.1	↑ 6.5	↑ 4.0	↑ 1.9	↑ 0.8	↑ 0.6
w/ Definition	33.4 46.4 38.8	43.0 63.9 51.4	23.0 53.6 32.2	47.9 66.9 55.9	45.9 24.7 32.1	81.7 53.5 64.7
+ LiP	37.7 46.3 41.5	48.1 60.9 53.7	34.1 54.7 42.0	45.3 71.9 55.6	53.2 23.6 32.7	81.6 46.7 59.4
(Δ F1)	↑ 2.7	↑ 2.3	↑ 9.8	↓ 0.3	↑ 0.6	↓ 5.3
w/ Guidelines	36.2 43.1 39.4	37.8 62.5 47.1	23.0 50.7 31.7	43.8 71.4 54.3	47.5 29.2 36.2	81.1 48.8 61.0
+ LiP	41.0 39.5 40.2	43.5 59.2 50.1	30.1 48.6 37.2	46.4 69.6 55.6	51.0 27.4 35.7	77.6 44.8 56.8
(Δ F1)	↑ 0.8	↑ 3.0	↑ 5.5	↑ 1.3	↓ 0.5	↓ 4.2
w/ Def&Guide	38.8 43.0 40.8	40.8 62.8 49.4	24.8 51.3 33.5	47.5 67.8 55.9	48.0 25.0 32.9	83.4 48.3 61.2
+ LiP	41.0 43.1 42.0	44.2 59.9 50.9	33.2 49.2 39.6	47.3 71.0 56.8	51.8 25.3 34.0	80.5 46.1 58.7
(Δ F1)	↑ 1.2	↑ 1.5	↑ 6.1	↑ 0.9	↑ 1.1	↓ 2.5

Table 1: Main experiment results.

top F1-scores than other knowledge, with a requirement of only a small list of sample entities to generate, rather than annotated examples. On LLAMA-3, literal patterns reach 4 out of 6 top F1-scores, where definition and annotation guidelines reach 1 respectively. On QWEN-2.5, literal patterns reach 4 out of 6 top F1-scores, where definition reaches 2 and none for annotation guidelines.

(3) Literal patterns as add-on Considering literal patterns as an add-on over other knowledge (row 4 to 3, 6 to 5, 8 to 7), for LLAMA-3, injecting literal patterns often yields simultaneous improvements in precision and recall over the baselines; although trade-offs occasionally occur, higher F1-scores are frequently attained. In 18 comparisons on LLAMA-3, 10 demonstrate concurrent gains in precision and recall, 8 exhibit trade-offs (of which 5 yield F1-score improvements and 3 declines).

For QWEN-2.5, trade-offs are more prevalent: among 18 comparisons, 3 achieve simultaneous precision and recall enhancements, 12 involve trade-offs (with 10 F1-score increases and 2 decreases), and 3 result in reductions in both precision and recall.

(4) Comparison between LLMs Generally, LLAMA-3 achieves higher recall, while QWEN-2.5 yields higher precision, which indicates that LLAMA-3 tends to include more potential entities in its prediction, leading to an increment in both true and false labels. Moreover, literal patterns that are effective on one model may fail to improve the performance on another (see BC5CDR). This indicates that model-specific characteristics are also essential in the efficiency of external knowledge injection, highlighting the necessity of model-specific prompt engineering when applying LiP-NER.

5.2 Type-wise Analysis (RQ2)

By looking into the results, we have some observations about the characteristics of the entity types that benefit from literal patterns and those does not. Table 2 shows the results of the entity types mentioned in this section.

The first kind of entity types that may benefit

Prompt	Dataset & Entity Type (Metrics: Micro-P, R, F1 percentages)														
	MIT-restaurant						movie-trivia			GENIA					
	Dish			Price			Relationship			DNA			RNA			cell_line		
META-LLAMA-3-8B-INSTRUCT																		
Vanilla	24.8	**85.7**	38.5	28.0	45.6	34.7	1.3	20.5	2.4	23.9	46.8	31.6	4.5	66.4	8.4	15.2	49.4	23.3
+ LiP	27.8	84.0	41.7	33.9	**49.1**	40.1	**9.9**	**50.9**	**16.5**	20.7	**52.0**	29.6	4.5	76.0	8.5	17.1	43.3	24.5
w/ Definition	26.0	85.4	39.9	21.3	43.3	28.5	1.6	32.8	3.1	32.2	42.4	36.6	9.1	50.0	15.4	18.5	**49.9**	27.0
+ LiP	**28.9**	83.6	**43.0**	32.2	48.5	38.7	4.9	48.0	9.0	24.8	49.5	33.0	8.4	74.0	15.1	18.6	45.1	26.4
w/ Guidelines	25.5	83.6	39.1	27.4	39.2	32.2	1.6	26.9	2.9	26.9	35.9	30.8	5.5	52.9	10.0	19.4	38.5	25.8
+ LiP	26.7	82.9	40.4	36.9	40.4	38.6	4.0	**50.9**	7.3	24.2	50.9	32.8	5.7	76.9	10.7	17.2	40.3	24.1
w/ Def&guide	25.8	84.7	39.5	27.9	33.3	30.4	1.2	20.5	2.2	**36.1**	36.5	36.3	**11.4**	53.9	**18.8**	23.5	45.1	30.9
+ LiP	27.8	83.6	41.7	**37.7**	43.9	**40.5**	3.1	44.4	5.8	29.8	51.3	**37.7**	9.6	**77.9**	17.1	**24.5**	42.8	**31.1**
QWEN2.5-7B-INSTRUCT																		
Text-first	57.0	67.9	62.0	39.6	40.9	40.2	0.6	5.9	1.0	36.3	13.0	19.1	31.4	42.3	36.1	29.8	23.9	26.6
+ LiP	62.3	62.7	62.5	**49.7**	**54.4**	**52.0**	8.2	33.9	13.2	**57.0**	**17.5**	**26.8**	**62.1**	**51.9**	**56.5**	27.0	21.2	23.8
w/ Definition	59.8	69.0	**64.1**	40.3	36.3	38.2	2.7	**43.3**	5.1	38.0	4.2	7.6	42.2	26.0	32.1	**32.1**	21.6	25.9
+ LiP	**63.0**	59.9	61.4	47.9	53.8	50.7	9.6	36.3	15.2	52.2	10.8	17.9	57.1	30.8	40.0	29.8	18.5	22.8
w/ Guidelines	47.9	**74.6**	58.3	12.5	4.1	6.2	2.0	28.7	3.7	34.3	5.8	9.9	39.3	31.7	35.1	30.1	**26.2**	**28.0**
+ LiP	59.3	59.9	59.6	44.8	42.7	43.7	6.7	37.4	11.4	49.3	8.7	14.7	56.1	35.6	43.5	29.7	21.4	24.9
w/ Def&Guide	57.9	69.0	63.0	20.4	6.4	9.8	2.1	32.2	4.0	37.6	4.1	7.3	51.4	36.5	42.7	30.9	23.2	26.5
+ LiP	61.4	63.1	62.2	38.1	40.4	39.2	**12.6**	39.2	**19.0**	50.2	8.5	14.5	57.1	30.8	40.0	26.9	19.8	22.8

Table 2: The results of the entity types mentioned in Section 5.2.

from literal patterns is the entity types with relatively standardized naming conventions but limited world knowledge in LLMs. For these entity types, LLMs may fail to gather sufficient world knowledge about entities and their types during the pre-training stage, leading to an underperformance of both their vanilla ability and the capacity to leverage semantic knowledge that relies on such knowledge. These entity types are often from specialized domains, where naming conventions are commonly standardized, allowing LLMs to summarize them coherently through few sample entities. This kind of entity types highlight the motivation of this work: provide literal features to alleviate the requirement of world knowledge within the LLMs.

For instance, for the GENIA dataset on QWEN-2.5, literal patterns have a significant impact on both precision and recall of the *DNA* and *RNA* types, leading to a leap on F1-scores (DNA: 19.1 to 26.8; RNA: 36.1 to 56.5). On LLAMA-3, the same literal patterns lead to a drastic boost in recall at the cost of precision. This is consistent with the feature of LLAMA-3: it tends to include more potential entities, and literal patterns further amplify this tendency. This indicates that the capability of utilizing literal patterns is model-specific.

Another kind of entity types that may benefit from literal patterns is the entity types with broad and ambiguous name or definition, while the actual entities within these types exhibit limited variation. For such types, the type names and definitions may fail to accurately describe the target type and could even mislead LLMs. However, the limited variation in the entity names allows effective literal patterns to be formulated, which may mitigate the deficiencies in type names and definitions in representing entity distributions, thereby improving performance. This kind of entity types highlights the importance of precisely describing target entity types when applying LLMs to NER tasks.

For instance, MIT-restaurant's *Price* type includes adjectives (e.g. cheap, high) and price ranges (e.g. below 10 dollars) beyond numeral prices, which are not likely to be covered by the type name and are not detailed in the generated definition and annotation guidelines. Hence, literal patterns which address these nuances could improve both precision and recall scores on both models.

Another example is MIT-movie-trivia's *Relationship* type. This type focuses on the relationships between a movie and the series it belongs to, and between a role and the movie, etc., where the entities are often multi-word phrases like "third film in a series". This specialized annotation scope requires detailed information to enable proper alignment.

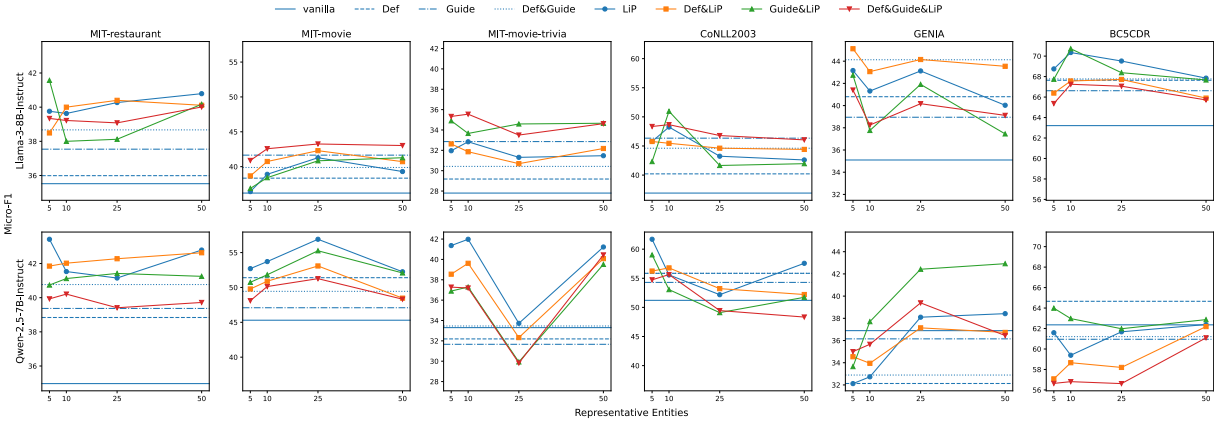

Figure 4: Few-shot experiments on MIT-restaurant dataset. We tested the literal patterns generated with different amount of sample entities from 5 to 50. The results show that the performance of LiP-NER does not necessarily grow with the increment in the amount of sample entities.

On the contrary, for the types that is diverse in names, applying literal patterns may lead to a focus on a subset of the type. An example is MIT-restaurant's *Dish* type, which includes the main ingredients and the forms of dishes, the methods to prepare, etc., and literal patterns with high coverage are hard to form. Thus, the results demonstrate an increment in precision and a decrease in recall.

Another example is GENIA's *cell_line* type. This type is almost identical to another *cell_type* type, the biggest literal difference is the "line" word at the end, which doesn't always appear. The literal patterns may mislead the models to include *cell_type* entities into predictions, or focus on the "line" word, leading to a decrease in both precision and recall.

5.3 Quality Analysis of Literal Patterns (RQ3)

To investigate the effect of the amount of sample entities, we generated literal patterns using various amounts of sample entities (from 5 to 50) across six datasets, with results presented in Figure 4. We observe that increasing the number of sample entities does not necessarily yield performance gains, and the trends of performance differ on different models. These findings suggest that the performance of LiP-NER is more driven by the quality of the literal patterns and the characteristics of the models than by the sheer quantity of sample entities.

In MIT-movie's *RATINGS_AVERAGE* type, MIT-restaurant's *Hours* type, CoNLL-03's *MISC* type, GENIA's *cell_line* type, and BC5CDR's *Disease* type, we found the literal patterns that consistently perform well across different models and whether other knowledge are provided or not, as well as those that perform poorly in any condition. By comparing the well-performing literal patterns with those that underperform, we offer preliminary insights about the quality of literal patterns. We list these literal patterns in appendix C.

For types with certain spelling patterns, it is necessary to explicitly indicate their main spelling features (such as keywords and affixes) in a dedicated entry. Including several example entities that contain these keywords or roots in an implicit way does not substitute for directly specifying these key spelling features.

For entity types that have numerous branches featuring different patterns, listing patterns of different branches in detail could lead to a broader potential coverage. The descriptions of the branches should reflect genuine regularities, rather than stiff explanations based on a single example.

For miscellaneous types like *MISC* in CoNLL-03, which consist of a mix of different subtypes, the literal patterns should cover the subtype that constitutes the majority rather than the minorities. This way, the annotation pattern aligns more closely with the target type, thereby improving performance.

6 Conclusion

In this paper, we presented LiP-NER, an LLM-based NER framework that leveraged literal patterns written in natural language to inject orthographic and morphological knowledge of target entity types into LLMs. In addition, we introduced a method to acquire literal patterns via LLMs, which required only a small list of sample entities rather than any annotation example. Through extensive

experiments, we demonstrated the effectiveness of our framework over baselines. We analyzed performance across various entity types and observed that types with relatively standardized naming conventions but limited world knowledge in LLMs, as well as those with broad or ambiguous names or definitions yet low internal variation among entities, benefited most from our approach. We conducted few-shot experiments and found that it was the quality of literal patterns and the intrinsic characteristics of the models that affect the performance. We conducted a quality analysis of literal patterns and concluded that the most effective literal patterns were (1) detailed in classification, (2) focused on majority cases rather than minorities, and (3) explicit about obvious literal features. Considering the feasibility of LiP-NER as a model-agnostic approach and its demonstrated generalization capabilities, we expect our work to enhance the performance in LLM-based NER.

Limitations

Our prompt templates require a separate inference for each entity type. While this allows the LLM to focus on recognizing one entity type at a time, it ties the computational cost for processing each input to the number of entity types. In addition, literal patterns are relatively lengthy form of external knowledge, which incurs a high inference cost. How to compress the literal patterns without sacrificing its effectiveness, or how to represent it in a more efficient form, is left for future work. Besides, providing several kinds of external knowledge in one-round conversation causes interplay between them in a black-box way. Offering these knowledge in a CoT way may have different result, which is left for future work. Finally, for most types, literal patterns can cover a large portion but not all entities. Even for domains and entity types with naming conventions approved by expert committees—for example, the human gene naming conventions ratified by the HUGO Gene Nomenclature Committee (HGNC)—it is impossible to retrospectively cover every gene name. Therefore, one should not expect to find a perfect set of literal patterns that encompasses all potential entities.

Ethnics Statement

There are no ethics-related issues in this paper. The data and resources utilized in this work are open-source and widely used in many existing studies.

References

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

Casimir Borkowski and Thomas J. Watson. 1967. An experimental system for automatic recognition of personal titles and personal names in newspaper texts. In *COLING 1967 Volume 1: Conference Internationale Sur Le Traitement Automatique Des Langues*.

Nigel Collier, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Jin-Dong Kim. 2004. Introduction to the bio-entity recognition task at JNLPBA. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)*, pages 73–78, Geneva, Switzerland. COLING.

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

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, et al. 2023. Retrieval-augmented generation for large language models: A survey. *CoRR*.

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

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Michael A. Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2021. A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. 2023. Tabllm: Few-shot classification of tabular data with large language models. In *International Conference on Artificial Intelligence and Statistics*, pages 5549–5581. PMLR.

Amr Hendy, Mohamed Abdelrehim, Amr Sharaf, Vikas Raunak, Mohamed Gabr, Hitokazu Matsushita, Young Jin Kim, Mohamed Afify, and Hany Hassan Awadalla. 2023. How good are gpt models at machine translation? a comprehensive evaluation. *arXiv preprint arXiv:2302.09210*.

Yan Hu, Qingyu Chen, Jingcheng Du, Xueqing Peng, Vipina Kuttichi Keloth, Xu Zuo, Yujia Zhou, Zehan Li, Xiaoqian Jiang, Zhiyong Lu, et al. 2024. Improving large language models for clinical named entity recognition via prompt engineering. *Journal of the American Medical Informatics Association*, 31(9):1812–1820.

Bernal Jimenez Gutierrez, Nikolas McNeal, Clayton Washington, You Chen, Lang Li, Huan Sun, and Yu Su. 2022. Thinking about GPT-3 in-context learning for biomedical IE? think again. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4497–4512, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

J-D Kim, T Ohta, Y Tateisi, and J Tsujii. 2003. Genia corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl_1):i180–i182.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016.

Peng Li, Tianxiang Sun, Qiong Tang, Hang Yan, Yuanbin Wu, Xuanjing Huang, and Xipeng Qiu. 2023. CodeIE: Large code generation models are better few-shot information extractors. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15339–15353, Toronto, Canada. Association for Computational Linguistics.

Jingjing Liu, Panupong Pasupat, Scott Cyphers, and Jim Glass. 2013. Asgard: A portable architecture for multilingual dialogue systems. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8386–8390. IEEE.

David McDonald. 1993. Internal and external evidence in the identification and semantic categorization of proper names. In *Acquisition of Lexical Knowledge from Text*.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2):1–40.

Diego Molla, Menno Van Zaanen, and Daniel Smith. 2006. Named entity recognition for question answering. In *Proceedings of the Australasian Language Technology Workshop 2006*, pages 51–58.

OpenAI. 2023. Chatgpt. https://openai.com/blog/chatgpt.

Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813*.

Joshua Robinson and David Wingate. 2023. Leveraging large language models for multiple choice question answering. In *The Eleventh International Conference on Learning Representations*.

Oscar Sainz, Iker García-Ferrero, Rodrigo Agerri, Oier Lopez de Lacalle, German Rigau, and Eneko Agirre. 2024. GoLLIE: Annotation guidelines improve zero-shot information-extraction. In *The Twelfth International Conference on Learning Representations*.

Dominic Seyler, Tatiana Dembelova, Luciano Del Corro, Johannes Hoffart, and Gerhard Weikum. 2018. A study of the importance of external knowledge in the named entity recognition task. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 241–246, Melbourne, Australia. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

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*.

L. Weston, V. Tshitoyan, J. Dagdelen, O. Kononova, A. Trewartha, K. A. Persson, G. Ceder, and A. Jain. 2019. Named entity recognition and normalization applied to large-scale information extraction from the materials science literature. *Journal of chemical information and modeling*, 59(9):3692–3702.

Yue Wu, Jie Huang, Caie Xu, Huilin Zheng, Lei Zhang, and Jian Wan. 2021. Research on named entity recognition of electronic medical records based on roberta and radical-level feature. *Wireless Communications and Mobile Computing*, 2021(1):2489754.

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

Andrew Zamai, Andrea Zugarini, Leonardo Rigutini, Marco Ernandes, and Marco Maggini. 2024. Show less, instruct more: Enriching prompts with definitions and guidelines for zero-shot ner. *arXiv preprint arXiv:2407.01272*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

GuoDong Zhou and Jian Su. 2002. Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 473–480, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

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

A Prompt Template for Inference

See Figure 5.

B External Knowledge of Case Study

Definition. *'protein' refers to any molecule composed of one or more chains of amino acids, which serve various biological functions including structural support, catalysis, signaling, and immune response.*

Annotation Guidelines. *Do not label general biological terms or unrelated uses of the word 'protein.' Be cautious of phrases that use 'protein' as part of a larger name (e.g., 'protein kinase A' refers to a specific protein, not a general reference to a protein). Avoid labeling entities such as 'protein' in non-scientific contexts or when referring to food, like in 'protein-rich diet,' unless specifically referring to the biological molecule.*

Literal Patterns. *Protein names may include abbreviations (e.g., SAPK, ERP, NGF-R) that represent functional categories, molecular families, or receptor types. Hyphenated forms (e.g., gp39-CD8 fusion protein, Gal4-Eed fusion protein) indicate fusion proteins or chimeric molecules, where two distinct proteins are combined. Functional descriptions are often used to specify the activity or role of the protein (e.g., active death effector proteases). Acronyms or abbreviations derived from full names (e.g., mitogen-activated kinase, CCACC/Sp1) may be used to simplify naming. Some protein names reflect specific sequences or motifs (e.g., CCACC/Sp1, which may indicate a DNA-binding motif for Sp1). Use of "anti-" prefix (e.g., anti-Ig) suggests the protein is an antibody or related to immune recognition. Names often include detailed structural or domain information (e.g., Gal4-Eed fusion protein), highlighting the origin or interaction of specific domains.*

Prompt Template for Inference

```
# TASK:
Recognize named entities of type [{Type Name}]
from the text given below.
# TEXT:                                   Text Input
{Target Text}
# DEFINITION OF THE TYPE:                 + Definition
[{Type Name}]: {Definition}
# GUIDELINES OF THE TYPE:                 + Annotation Guidelines
[{Type Name}]:
{Annotation Guidelines}
# NOMENCLATURE OF THE TYPE:               + Literal Patterns
[{Type Name}]:
{Literal Patterns}
# OUTPUT FORMAT:                          Output Regulation for LLaMA-3
Return a JSON list containing only entity names
of type [{Type Name}].
First, retrieve entities of the required type from the user
text above.
Then, put only the original strings of the entities into
a JSON array. Do not make any object in the array.
Surround your JSON output with <JSON></JSON> tags.
Do not greet or explain.
# OUTPUT FORMAT:                          Output Regulation for Qwen2.5
Return a JSON list containing only entity names
of type [{Type Name}]. Do not greet or explain.
```

Figure 5: The prompt template for inference of LiP-NER. The term "nomenclature" was used in our experiments but is deprecated in this paper, due to its inaccuracy.

C Literal Patterns for Comparison

- **(a) MIT-restaurant: Hours**

  Good: *Use of specific time-related phrases such as "open," "close," and "dinner," often combined with times of day (e.g., "open until midnight," "dinner until 10 pm"). Occasional men-*

tion of days of the week or specific dates (e.g., "open on sunday," "friday at 6 pm"). Reference to time intervals and specific periods like "all night," "before noon," or "in the evening." Indication of time precision (e.g., "2 am," "around 6 pm," "until 11 pm"). Terms like "24/7," "open late," "late hours," and "open at this hour" are common. Informal phrases that refer to being open for an extended time or continuously (e.g., "still open," "stay open," "open all night"). Mention of meal times or specific events (e.g., "for lunch," "breakfast before 5 am," "dine in after 10"). Use of "right now" to indicate current availability or operational status. Casual time expressions like "soonest available," "in an hour," or "this late at night." Usage of "open after" or "close after" in specific time references (e.g., "open after 12," "close after 4 pm"). References to business operation, often using "open" or "open hours" (e.g., "business hours," "operation," "clock"). Daypart terms like "afternoon," "evening," and "midnight" to describe times of day. Some references to specific time intervals (e.g., "in 45 minutes," "two weeks").

Bad: *The term "Hours" encompasses specific time indications, either precise (e.g., "5 pm") or approximate (e.g., "late"). Time references can include both exact and relative phrasing (e.g., "open after 10 pm"). Phrasing may indicate frequency or availability (e.g., "open every day"). Contextual indicators like "today" can specify the relevance of the time mentioned (e.g., "5 pm today").*

- **(b) MIT-movie: RATINGS_AVERAGE**

Good: *Use of adjectives to describe the quality of films (e.g., "good," "very good," "mediocre"). Specific numeric ratings are commonly included (e.g., "five stars," "two stars," "eight stars and above"). Phrases indicating popularity or critical acclaim (e.g., "critically acclaimed," "liked by many," "blockbuster film"). Terms related to viewer opinions (e.g., "viewers rating," "audience," "reviews"). Reference to awards and recognition (e.g., "oscar," "best picture," "highest rated"). Descriptors that indicate comparison or ranking (e.g., "top 10," "lowest rated," "higest rated"). Use of superlative or comparative forms to emphasize quality (e.g., "best work," "higher viewers rating"). Informal or conversational language indicating recommendations (e.g., "must see," "should consider seeing"). Inclusion of categorical terms related to the context (e.g., "newly released comedy," "sequelsprequels").*

Bad: *The naming routine for type 'RATINGS_AVERAGE' includes specific requests for film ratings and reviews. It often mentions awards or accolades associated with the films, such as "Oscar winning" or specific award categories like "Best Picture." The requests typically specify a year or other criteria for the ratings, such as "four stars or higher." Language used in queries can include references to audiences, viewer ratings, and quality indicators (e.g., "best viewer rating").*

- **(c) CoNLL2003: MISC**

Good: *The examples include a variety of terms referring to specific countries, regions, or groups (e.g., "Zimbabwean," "Syrians," "Dutch"). There are several references to sporting events or competitions (e.g., "Davis Cup," "Ryder Cup," "Belgian Grand Prix"). Terms may reference political affiliations or ideologies (e.g., "Democrat," "Communist-led"). Some examples point to organizations or institutions (e.g., "CPI," "Australian Rules-AFL"). Names can refer to specific ethnic, cultural, or national identifiers (e.g., "Zionists," "Arab," "Turkish Kurd"). Some terms are related to specific product names or models (e.g., "VW Passat," "GT2 Konrad Porsche 911"). There are references to time periods, holidays, or specific events (e.g., "Labour Day," "Second Empire"). The use of capital letters is prominent for place names, events, and titles (e.g., "Windows NT," "MOROCCAN"). There are occasional abbreviations or acronyms (e.g., "SBF-120," "C$"). Some examples represent specific locations (e.g., "Vancouver-based," "Palestinian-ruled"). Terms may be linked to specific nationalities or identities (e.g., "New Zealander," "Belgian").*

Bad: *Many entries are related to organizations, tournaments, or events, often with geographic or descriptive modifiers (e.g., "PGA Tour", "21st African Cup of Nations"). Some entries refer to specific currencies, regions, or historical terms (e.g., "US$", "East Java", "Gulf War"). Abbreviations or acronyms are common, sometimes indicating military, organizational, or political groups (e.g., "NATO-led", "IMF-hosted"). Common use of hyphenated terms, often combining locations or political entities (e.g., "Burundi-*

Central Africa", "Serb-held"). Some entries refer to awards, recognitions, or titles (e.g., "Bharat Ratna", "Most Valuable Player"). Titles and names of products or specific items also appear (e.g., "AK-47", "F-14"). Entries may involve sports and entertainment, referencing leagues, players, or events (e.g., "Davis Cup", "All-Star"). Geographic references may specify regions or areas linked with political or historical significance (e.g., "Nablus-based", "Gaza-based"). Occasionally, cultural or historical references are used without modification (e.g., "Nazism", "Civil War").

- **(d) GENIA: cell_line**

  Good: *The nomenclature often includes the type of cell or organism followed by the descriptor "cell line" or a specific cell line identifier. Common terms include "cells" or "cell line" after the name (e.g., "Daudi cells", "H9 T-cell line"). Specific terms often refer to the function, origin, or stimulation type of the cells (e.g., "IL-5-stimulated cells", "PHA-activated cells"). Abbreviations for specific cell lines or organisms are frequently used (e.g., "CV-1 cells", "CHO cells"). Cell lines are sometimes referred to by their species of origin (e.g., "murine B-cell lymphoma cell line"). The use of prefixes or markers, such as "CD68+" or "Nef-expressing", provides further classification or description. Some entries include the specific context or condition under which the cells are used (e.g., "IL-2-dependent cell lines", "monoblast-like U937 cells"). The cell line name may also include additional specific features, such as mutations, expression markers, or environmental conditions (e.g., "BFU-E-derived cells", "promonocytic THP-1 cells").*

  Bad: *Cell line names often reflect the species, cell type, or functional characteristics. Specific terminology like "T-cell line," "B-cell line," or "myeloid precursor" indicates the origin or differentiation pathway of the cells. Abbreviations and acronyms (e.g., "CTLL-2," "U937") are commonly used for well-established cell lines. Modifiers such as "estrogen-dependent," "peptide-specific," or "serum-activated" provide additional functional or behavioral details about the cell lines. Numeric designations in names (e.g., "CTLL-2") are typically unique identifiers for specific subtypes or variations of cell lines. Cell type description (e.g., "monocytoid," "myeloid," "lymphoblastoid") is frequently used to classify the cells based on their morphology or lineage. Species indicators may be included (e.g., "murine," "human") to specify the origin of the cell line. No uniform standard for combining terms: cell lines may sometimes include hybrid terms like "myeloid precursor" or "hemopoietic cells."*

- **(e) BC5CDR: Disease**

  Good: *- Many disease names consist of medical terms combined with suffixes indicating a condition (e.g., "hypoxaemia," "myocarditis"). - A variety of diseases are named based on their affected organs or body systems (e.g., "cardiac disease," "renal damage"). - Conditions with a genetic or clinical origin often feature terms like "dysfunction," "disorder," or "syndrome" (e.g., "attention-deficit/hyperactivity disorder," "nephrotic syndrome"). - Some diseases are named after the type of abnormality they involve, such as "dysphoric reaction" or "tremor" (e.g., "dyskinesia"). - Certain terms describe the cause or mechanism of the disease (e.g., "poisoning," "viremia"). - Malignant and benign tumor types often include descriptors of tissue or cell type (e.g., "squamous cell carcinoma," "mesenchymal tumors"). - Diseases may be named after specific symptoms or affected features (e.g., "amnesia," "impaired renal function"). - Specific acronyms or shortened terms may be used for more complex or widely recognized conditions (e.g., "TDFS," "RPN"). - A few names use the combination of a region or function with a clinical suffix indicating the condition (e.g., "cerebral infarction," "putaminal hemorrhage"). - Some diseases include the word "disorder" or "syndrome" to denote an abnormal condition or disease state (e.g., "gastrointestinal disorder," "major depression").*

  Bad: *- The naming of diseases often involves the use of specific medical terms that describe the condition or its effects. - Many names reflect a combination of anatomical locations (e.g., "liver mass," "renal failure") and physiological processes or symptoms (e.g., "sepsis," "apnea"). - Conditions may also be named after specific characteristics or pathological features (e.g., "intermittent claudication," "Ehrlich ascites tumor"). - Some names may include a combination of organ systems or multiple conditions (e.g., "renal and*

hepatic dysfunction," "acute renal failure and hepatic failure"). - The nomenclature can also involve abbreviations or shorthand for more complex conditions (e.g., "TD," "TAA"). - Certain terms may refer to a specific disease entity or syndrome (e.g., "Angiosarcoma," "L1210 leukemia," "Ebstein's anomaly"). - Descriptions may involve a process or complication caused by a disease, such as "adverse effect," "disruptive behaviors," or "Q-T prolongation." - Several conditions are defined by their clinical manifestations or outcomes, such as "deaths" or "respiratory distress."