Enhancing Software-Related Information Extraction via Single-Choice Question Answering with Large Language Models

Wolfgang Otto[1][0000−0002−9530−3631], Sharmila Upadhyaya[1][0009−0003−7142−3887], and Stefan Dietze[1,2][0009−0001−4364−9243]

[1] GESIS – Leibniz Institute for the Social Sciences, Cologne, Germany
{firstname.lastname}@gesis.org
[2] Heinrich-Heine-University Düsseldorf, Germany
stefan.dietze@hhu.de

Abstract. This paper describes our participation in the Shared Task on Software Mentions Disambiguation (SOMD), with a focus on improving relation extraction in scholarly texts through generative Large Language Models (LLMs) using single-choice question-answering. The methodology prioritises the use of in-context learning capabilities of LLMs to extract software-related entities and their descriptive attributes, such as distributive information. Our approach uses Retrieval-Augmented Generation (RAG) techniques and LLMs for Named Entity Recognition (NER) and Attributive NER to identify relationships between extracted software entities, providing a structured solution for analysing software citations in academic literature. The paper provides a detailed description of our approach, demonstrating how using LLMs in a single-choice QA paradigm can greatly enhance IE methodologies. Our participation in the SOMD shared task highlights the importance of precise software citation practices and showcases our system's ability to overcome the challenges of disambiguating and extracting relationships between software mentions. This sets the groundwork for future research and development in this field.

Keywords: Generative Large Language Models · Information Extraction · Named Entity Recognition · Relation Extraction · Software Citation · Retrieval-Augmented Generation · Single-Choice Question Answering · Software Mentions Disambiguation Task

1 Introduction

The evolution of information extraction (IE) in scholarly communications necessitates the development of innovative methodologies to accurately identify and categorise software mentions. This is a critical component for ensuring research transparency and reproducibility. The Shared Task on Software Mentions Disambiguation (SOMD) highlights the importance of refined citation practices amidst inconsistent referencing of software artifacts. This research employs generative

Large Language Models (LLMs) to address the challenges in software mention extraction and relation identification, marking a significant step towards the sophisticated extraction of software-related information. Our approach integrates Retrieval-Augmented Generation (RAG) techniques with LLMs to dissect and comprehend the intricate web of software citations and their attributive details (e.g. the developer or the version) within scientific texts. By examining the potential of LLMs in performing IE tasks by transforming them into single-choice question-answering, we present a comprehensive analysis that addresses the nuances of software mention extraction and could be applied on a broader scope of scholarly artifacts NER and relation extraction. This paper explores the complexities of applying LLMs to NER tasks, providing insights into the challenges and proposing a new methodology for relation extraction that could pave the way for future innovations in the field.

2   Related Work

This section explores works in Named Entity Recognition (NER) and Information Extraction (IE), focusing on software mention extraction, scholarly artifact NER, and the use of generative LLMs in these areas.

In software mention extraction, the Softcite dataset [2] and the SoMeSci knowledge graph [7] are significant. Softcite offers a gold-standard dataset for extracting software mentions from biomedical and economic research, while SoMeSci Knowledge Graph includes software mentions in scientific articles with relation labels such as version, developer and citations, highlighting the need for accurate software mention extraction.

In scholarly artifact NER, Saji and Matsubara [6] introduced a method using academic knowledge graphs to extract research resource metadata from scholarly papers, enhancing metadata quality and repository size. Otto et al. [5] developed the GSAP-NER corpus for extracting machine learning-related entities from scientific publications, filling the gap in general-purpose NER models. These words demostrate the importance of domain-specific NER tasks and leverageing knowledge graphs for enhancing research resource repositories.

Regarding LLMs for NER, Wang [10] explores the use of text-generation models for sequence labeling and underlying challenges, particularly in low-resource and few-shot setups. Furthermore, Xie [11] investigates LLMs' self-improving capabilities for zero-shot NER.

For LLMs in relation extraction, Wadhwa [8] uses few-shot prompting and fine-tuning with large language models achieving state-of-the-art performances in relation extraction using LLMs, while Wan [9] introduce GPT-RE to enhancing relation extraction accuracy through task-specific entity representations.

Lastly, Xu [12] conducts a comprehensive survey providing an overview of generative information extraction using LLMs, categorizing works by IE subtasks and learning paradigms, and highlighting the transformative potential of LLMs in IE.

Overall, these studies underscore the evolving methods and significant contributions in extracting information from scholarly texts using specialized datasets, domain-specific approaches, and advanced generative models.

3   SOMD Shared Task

The Shared Task on Software Mentions Disambiguation (SOMD) aims to enhance transparency and reproducibility in scientific research by improving software citation practices. Participants are required to develop models that can identify and disambiguate software mentions in scholarly texts using the expanded Software Mentions in Science (SoMeSci) knowledge graph, with a focus on AI and Computer Science.

Subtask 1, Software NER, involves identifying four types of software-related entities: Application, Plugin, Operating System, and Programming Environment. Accurately classifying software mentions is necessary to understand their role in academic discourse.

Subtask 2, Attribute NER, aims to extract ten different attributive information associated with software entities, such as alternative names, abbreviations, authorship (developers), software release maintenance (versions, extensions, release dates, licenses), and elements of the referencing system (in-text citations, URLs, and software coreferences for cross-sentence linkage), to a specific software mention.

Subtask 3, Relation Extraction, involves establishing relationships between software entities and their attributes using specific relation types and mapping each attribute to these relations. This task aids in comprehending the interrelations of software entities within scholarly texts.

The tasks **evaluation metric**, is the Weighted Average Macro F1 score. This metric adjusts the influence of each label on the final result based on the number of test instances assigned to it, ensuring that labels with fewer instances have a proportionally smaller impact on the overall performance evaluation.

4   Using LLMs for Software Related IE-Tasks

The exploration of generative LLMs such as GPT-4 for Information Extraction (IE) tasks related to software underscores the transformative potential these models hold as general-purpose task solvers. The allure of leveraging LLMs in this capacity is significant, given their ability to process and generate human-like text across a wide range of topics and formats. However, the challenge arises in the specificity and nuanced requirements of domain-specific tasks, such as NER within specialized fields. It has been observed that, despite their vast knowledge and versatility, current models like GPT-4 often fall short when tasked with domain-specific NER, primarily due to their generalized training and lack of domain-specific tuning.

To mitigate these shortcomings and enhance the performance of LLMs in specialized IE tasks, various in-domain learning strategies are employed. These

strategies are designed to equip the LLM with a deeper understanding of the task at hand, essentially guiding the model towards more accurate identification and classification of relevant text spans. Among these strategies, optimizing the task description plays a crucial role. A well-crafted, precise task description can significantly improve the model's focus and comprehension of the task's objectives, leading to more relevant and accurate outcomes.

Furthermore, the provision of speaking, prototypical examples serves as another effective strategy. By presenting the model with clear, illustrative examples that encapsulate the essence of the task, we can anchor its understanding and enhance its ability to generalize from these examples to new, unseen instances. This approach leverages the model's inherent learning capabilities, allowing it to draw parallels and apply learned concepts to the task at hand.

Additionally, augmenting the model's capabilities with Retrieval Augmented Generation (RAG) introduces a powerful dimension to the information extraction process. RAG combines the generative prowess of LLMs with the specificity and relevance of retrieved documents, enabling the model to access a broader context and detailed examples that are pertinent to the task. This strategy is particularly advantageous in domain-specific applications, where the relevance and accuracy of the information extracted are paramount.

In our approach, we capitalize on the last strategy, RAG, to maximize the utility of training sets for each specific task. The retrieval component of this strategy entails identifying instances within the training set that are similar to the test instance and can provide valuable insights for the identification and classification of relevant information. This method not only enhances the model's performance by providing it with task-relevant data but also ensures that the information extracted is of high relevance and quality, tailored to the specific demands of the domain-specific NER task. Through these tailored strategies, we aim to bridge the gap between the broad capabilities of LLMs and the precise requirements of domain-specific information extraction tasks, paving the way for more effective and efficient utilization of generative language models in specialized domains.

4.1   Challenges in Applying LLMs to NER Tasks

The integration of generative LLMs into NER tasks introduces a set of unique challenges that can significantly impact the performance and reliability of extraction outcomes. One of the most prevalent issues in generative NER approaches is the phenomenon of hallucination[10], where the model generates entities not present in the test instance. This can result from the model misinterpreting the provided examples as part of the text from which entities should be extracted, leading to inaccuracies and inconsistencies in the results. n Furthermore, matching problems during the location of mention positions present considerable challenges, particularly in the context of span-based evaluation systems. These systems evaluate the accuracy of entity extraction based on the precise span of text identified as an entity. Discrepancies in the extracted span—whether through corrected spellings or variations in representation—can complicate the

matching process. For example, the extraction of "jquery" as "jQuery" illustrates a common scenario where the prevalent spelling of a software library may differ from its mention in the text, yet both are correct. This variability necessitates sophisticated matching strategies to ensure accuracy in evaluation.

The situation is further complicated by texts containing multiple mentions of a single entity. The challenge lies in determining whether each mention can be accurately classified as an entity, alongside dealing with the ambiguity of different entities that are written in the same way. Overlapping or nested entity mentions that do not align with the ground truth data introduce additional layers of complexity, requiring nuanced approaches to entity recognition and classification. Our proposed baseline solution, filtering out non-matching entities and employing rule-based decisions for handling multiple matches have been adopted. However, the disadvantage of this method is that it is based on simplistic heuristics.

Future work may explore advanced solutions like using LLMs for precise entity matching, offering potential improvements for NER challenges. While this paper does not delve into these complex methods, it highlights the importance of ongoing research to further refine and improve LLMs for more accurate and efficient entity extraction.

4.2   Sample Retrieval for RAG on various IE-Tasks

Retrieval-Augmented Generation (RAG) significantly bolsters the capabilities of LLMs in IE tasks by effectively utilizing both unstructured and structured contexts [3, 4]. This dual approach is essential in IE for achieving precise extractions, yet the selection of optimal samples for the generative process poses a substantial challenge. Addressing this involves two primary strategies: utilizing sentence embeddings to find contextually similar textual content for the LLM, and identifying analogous entities to uncover beneficial training sentences. These methods enable the LLM to discern structural and semantic patterns for more accurate text extractions. A crucial obstacle is accurately identifying target entities within test instances, for which we employ a pre-trained Language Model (PLM) tailored to our NER task. This LM is instrumental in both spotting potential entity candidates and facilitating entity similarity searches, leveraging last hidden state embeddings from training examples to locate matching entities within the dataset.

Our methodology extends to evaluating various retrieval techniques and their impact on the LLM's efficiency, particularly within a Few-Shot learning framework. We explore different methods, including random illustrative samples, text similarity-based RAG, and entity-based sentence retrieval, to provide the LLM with contextually relevant examples, thereby optimizing the software entity extraction process from scientific texts. This exploration aims to identify the most effective strategies for utilizing LLMs in domain-specific software entity extraction tasks. By overcoming challenges related to resource demand, execution time, and precision in entity identification, our approach aims to enhance the accuracy and efficiency of NER processes, contributing valuable insights and methodologies to the field of computational linguistics and information extraction.

A special retrieval method is used for the RE Task. Because for this task, the entities for each test instance are given, we easily could list all possible relations (compare Table 3). We could use them to find similar relations in the train set. If we find more then one, we decide for the one, with highest sentence similarity.

For the RE Task, a special retrieval method is employed. As entities for each test instance are provided, all possible relations can be listed based on Table 3. These relations can then be used to find relations of the same type, and with the same domain and range in the train set. If multiple similar relations are found, the one with the highest sentence similarity is selected.

4.3   Extraction of Software Entities

The extraction of software entities from scientific texts represents a specialized challenge within the realm of NER, targeting the identification of software entities across four distinct categories ranging from applications to operating systems. Furthermore, this task extends beyond mere identification, seeking to understand the intent behind each mention of software entities—whether it pertains to creation, usage, deposition, or mere citation.

We address this challenging task using LLMs, despite their high demands on resources and time, especially when processing extensive publications with sparse relevant text. Our approach includes a pipeline strategy that prioritizes selecting relevant text passages for LLM analysis, improving efficiency by filtering out unrelated content. This method's success depends on the selection accuracy, directly impacting recall. However, the trade-off for reduced computational costs justifies the potential minor decrease in recall. Our performance optimization employs a hybrid method, combining a fine-tuned NER model for sentence selection with LLMs for information extraction. This approach faces limits, notably when sentences crucial for analysis are missed in the selection phase, capping the LLM extraction phase's accuracy as indicated by an initial sentence classification task recall of 0.882 (0.884 F1). This establishes a theoretical limit on extraction accuracy due to potential false negatives, illustrating a balance between efficiency and the precision constraints of LLMs in detailed text analysis.

4.4   Extraction of Software Attributes

Following the identification of software entities within scientific texts, a further nuanced aspect of NER and IE tasks emerges in the extraction of associated software attributes. These attributes encompass a wide array of specific details, the version, developer, citations, URLs, release dates, abbreviations, licenses, extensions, software co-references, and alternative names. We used a similar approach as for subtask 1, and utilised train sample retrieval to augment the task description in a few shot setup. For each sample, including those derived from few-shot learning, the process entails presenting the sentence containing the software entity(ies) and then predicting a JSON list of identified entities along with their respective attribute types.

4.5   Relation Extraction as Single-Choice Question Answering Task

In the field of Natural Language Processing (NLP), the extraction of relations between entities within a text corpus poses significant challenges. This study proposes a novel approach by conceptualizing the task of relations extraction as a single-choice question-answering (QA) activity. This method entails generating a comprehensive list of all possible entities within a sentence, drawing from the existing entities and their relationships as delineated in the training dataset. Each potential pair of entities is then evaluated to ascertain if a specific relation, such as "version_of", appropriately links them. For instance, considering the relationship "version_of", a sentence may be formulated as "8 is the version of SPSS", representing a possible relation between the version number and the software entity.

   For every sentence in the dataset, this process yields a set of single-choice questions, each positing a potential relationship between entities. These questions are then prompted to a LLM for answering. The LLM's task is to select the most plausible relation from among the given options, thereby facilitating the extraction of accurate entity relations from the text. However, this approach is not without its challenges. A primary source of error stems from instances where multiple relations could plausibly link a pair of entities, leading to ambiguity and complicating the single-choice question-answering framework. Despite the challenges, we demonstrate that treating relation extraction as a single-choice QA task provides a structured and innovative approach to extracting valuable insights from complex textual data.

5   Experiments

5.1   Models

Fine-tuned Model In our experimental setup, the initial phase focused on fine-tuning a language model specifically for Subtask 1, which involved NER. Following the methodology similar to Schindler [7], we employed the SciBERT model [1], given its pre-training on scientific corpus, making it apt for NER fine-tuning within scholarly texts. The fine-tuning process entailed rigorous parameter optimization, including adjustments to the batch size, learning rate, and the relative share of negative samples—sentences that do not contain any annotations. This optimization utilized a 90/10 train/evaluation data split from the available training dataset. Subsequent to parameter tuning, we conducted a final training run with a modified data split of 95/5 train/evaluation to maximize the training data's utility.

Generative Large Language Models For a comparative analysis, our study integrated LLMs, specifically examining the performance differences between GPT-3.5-fast and GPT-4-fast models accessed via the OpenAI API. To ensure deterministic outputs for comparison, we set the temperature parameter to zero, eliminating randomness in the model's response generation.

5.2   Prompting

Software NER The prompting strategy for Software NER involved providing a concise task description, which included specifying the task as NER and intention classification, alongside an enumeration of target labels. We also highlighted the domain specificity of the texts (i.e., scientific publications) and requested the output in JSON format, delineating separate labels for entity type and intention. Sample sentences from the training set, along with their corresponding JSON output, were included as illustrative examples. This setup varied in the number and order of displayed examples based on the retrieval method, exploring the impact of these factors on model performance through separate experiments.

Attributive NER For Attributive NER, the prompt construction mirrored the approach taken in Subtask 1 but incorporated more detailed rules, emphasizing the necessity for attributes to relate directly to software entities. Known software entities were additionally provided as input, even for test instances, adhering to the oracle setup defined in the shared task guidelines. This method aimed to refine the model's ability to extract and classify attributive information accurately.

Relation Extraction The approach to Relation Extraction reimagined the task as a series of single-choice question-answering challenges. Each potential relation, given the software and attributive entities within a sentence, was listed, with claims formulated for each possible relationship (e.g., "IBM is the developer of Windows"). The model was tasked with identifying the veracity of each claim through a single-choice question format, where all claims were enumerated and solutions provided in a batch format for each example sentence. This setup culminated in presenting the test instance alongside its single-choice questions, expecting the model to deliver decisions on the relational claims.

5.3   Train Sample Retrieval for Few-Shot Generation

Our experimental framework explored various methods for test sample retrieval to ascertain the most effective approach in enhancing model performance. These methods included the use of random illustrative samples to represent every possible signature, retrieval based on entity similarity, and retrieval based on sentence similarity. Each method was evaluated against a baseline to determine its impact on the accuracy and efficiency of the information extraction tasks at hand.

5.4   Relation Extraction Baseline

In the development of our baseline for relation extraction, we established a robust heuristic framework derived from an analysis of potential relations indicated within the text. Our strategy was guided by two principal rules aimed at simplifying the decision-making process for identifying accurate relationships between entities. Firstly, we limited our consideration to relations that necessitate the presence of at least one related entity, deliberately excluding optional inter-software entity relations such as "specification_of" and "PlugIn_of." Given the infrequent

Table 1. Results of Different Models and Retrieval Methods on Subtask 1

Paradigm	Retrieval	F1	Model	parameter
Finetuned	-	0.599	SciBERT	-
Prompt	Random	0.483	GPT 3-5	Random k=7
Prompt	Random	0.525	GPT 3-5	Random all entity types shown
Prompt	Sim. sentences	0.647	GPT 3.5	topn=10
Prompt	Sim. entities	0.624	GPT 3.5	topn=7 x n entities
Prompt	Random	0.574	GPT 4	Random all entity types shown
Prompt	Sim. sentences	0.677	GPT 4	topn=10
Prompt	Sim. entities	**0.679**	GPT 4	topn=7 x n entities

occurrence of these cases within the training dataset, we anticipated only a minimal impact on the overall performance metric, specifically the weighted mean macro F1 score for the subtask. Secondly, for all remaining relation types, our approach favored selecting the closest possible entity positioned to the left of the focal software entity as the most likely relation partner. This heuristic was not only straightforward but proved to be highly effective, aligning our baseline performance with that of the top contenders in the shared task. This methodology highlights the potential of leveraging simple, rule-based strategies to achieve competitive results in complex relation extraction challenges.

6   Results

Our analysis in Subtask 1 (Software NER) shows a varied performance landscape across different models and retrieval methods (Table 1). The finetuned baseline, which uses SciBERT, achieved a solid foundation with a 59.9% F1 score. However, LLMs that use random samples without fine-tuning showed a decrease in performance, with the highest F1 score reaching only 57.4%.

A closer examination of retrieval-based models indicates that LLMs perform better. The highest F1 score of 67.9% was achieved by sentence similarity retrieval models, while entity retrieval showed the best performance at 67.7% F1. The transition from GPT 3.5 to GPT 4 models resulted in a significant improvement of approximately 3–5%, although it required around three times more computation time. Notably, our best models were able to perform within a mere 3% below the theoretical maximum by utilizing SciBERT for sentence selection, leveraging oracle positive sentences.

In Subtask 2 (Attributive NER), our methodology showed a significant improvement of +10% in F1 performance compared to our competitors, demonstrating the effectiveness of our approach in a low data regime (Table 2). For Subtask 3 (Relation Extraction), our LLM Single-Question Answering model further improved the F1 score by 5.1%, building on the already competent performance of the heuristic baseline and highlighting the advantage of our method. Furthermore, it has been demonstrated that using 7-10 samples is the most effective strategy, as it optimises the balance between input complexity and model performance.

Table 2. SOMD Performance Rankings (Weighted Average Macro)

Task	#	User	F1	Precision	Recall	Task	#	User	F1	Precision	Recall
1	1	**phinx**	**.740**	**.761**	**.750**	2	1	*ours*	*.838*	*.835*	*.847*
1	2	david-s477	.692	.739	.711	2	2	phinx	.743	.745	.748
1	3	ThuyNT03	.678	.729	.649	3	1	*ours*	*.916*	*.911*	*.924*
1	4	*ours*	*.652*	*.679*	*.664*	3	2	phinx	.897	.900	.897
1	5	vampire	.648	.682	.637	3	-	*baseline*	*.864*	*.857*	*.875*
						3	-	*necessary*	*.562*	***.933***	*.415*

This analysis highlights the potential of utilising advanced LLM techniques and carefully selected retrieval methods to significantly improve the accuracy and efficiency of NER tasks in specialised domains.

7 Conclusion

Our research on enhancing Relation Extraction (RE) with LLMs through Single-Choice QA has introduced a novel intersection of methodologies aimed at improving the precision of information extraction in the context of scientific texts. By integrating Retrieval-Augmented Generation (RAG) with LLMs and adopting a methodical approach to fine-tuning and leveraging large language models such as SciBERT and use the model to support GPT variants, we have demonstrated the capability of LLMs to navigate the complexities inherent in the extraction of software entities and their attributes.

The exploration of different retrieval strategies—ranging using entity and sentence similarity underscores our commitment to refining the inputs for generative models, ensuring that they are fed the most relevant and contextually appropriate data. This meticulous preparation has allowed us to significantly boost the performance of LLMs in recognising nuanced distinctions among software-related entities and accurately extracting relation types within scholarly articles. Our experiments have not only highlighted the efficacy of LLMs in addressing domain-specific tasks with a limited set of examples but also revealed the inherent challenges, such as the difficulties in matching entity mentions accurately. Despite these hurdles, our single-choice QA approach for RE lead to a strong heuristic baseline for relation extraction and show how altering the task simplifies the problem.

The outcomes of our research indicate a promising direction for future work in leveraging LLMs for NER and RE tasks. The development of our system for participation in the SOMD shared task has illustrated the potential of a single-choice QA approach to relation extraction, offering a structured and scalable method for extracting meaningful insights from textual data. Our findings contribute to the growing body of knowledge on the application of generative models in the field of computational linguistics, paving the way for more sophisticated and efficient methodologies in information extraction from scientific literature.

A   Dataset Overview

Each subtask in the SOMD Shared Task is evaluated independently. Subtask 2's attributive NER uses the identified software entities from Subtask 1 as inputs, and the annotations from the first two subtasks are combined for Subtask 3's Relation Extraction. The dataset is hierarchically structured, with each subtask's test set forming a subset of the previous one. This lead to data leakage in Subtask 1 and to a lesser extent in Subtask 2. This design ensures that tasks are interconnected, but it requires careful interpretation of results, particularly with respect to error propagation. Although this design creates data leakage, it ensures the interconnectedness and cumulative nature of the challenges. The initial presentation of the Corpus is as the SoMeSci knowledge graph [7]. It is a comprehensive dataset consisting of 1,367 documents and containing 399,942 triples that represent 47,524 sentences. The dataset includes 2,728 software entities, totaling 7,237 labeled entities. The dataset includes positive and negative examples, i.e. sentences without software mentions, which improves the accuracy of the model. The presence of duplicate sentences, headings, and varying sentence lengths increases the complexity of the text and makes extraction tasks from scientific texts more challenging. For possible domains and ranges of the relations int the train set compare Table 3

B   Similarity Search Examples

B.1   Search By Entity Similarity

Table 4. Retrieval example based on entity similarity for RAG. The first row is the query entity. "sim" reflects the cosine similarity of the enities.

entity	label	sim	split	sentence
PhosphOrtholog	Application Creation	1.00	test	To this end , we have developed an automated web - based tool , PhosphOrtholog , which allows batch processing and mapping of large species - specific PTM datasets to compare overlap at a site - specific level .
SNPdetector	Application Creation	0.93	train	We developed a software tool , SNPdetector , for automated identification of SNPs and mutations in fluorescence - based resequencing reads .
ESPRIT - Forest	Application Creation	0.92	train	In this paper we developed a new algorithm called ESPRIT - Forest for parallel hierarchical clustering of sequences .

Table 3. Object Properties, Domain, and Range Classes with Inner Software Relations.

Range Class ⇒ Object Property	Application	Operating System	PlugIn	Programming Environment
Abbreviation ⇒ Abbreviation_of	✓	✓	✓	✓
Developer ⇒ Developer_of	✓	✓	✓	✓
Release ⇒ Release_of	✓	✓	✓	✓
Version ⇒ Version_of	✓	✓	✓	✓
Citation ⇒ Citation_of	✓	✗	✓	✓
PlugIn ⇒ PlugIn_of	✓	✗	✓	✓
URL ⇒ URL_of	✓	✗	✓	✓
Extension ⇒ Extension_of	✓	✓	✗	✗
License ⇒ License_of	✓	✗	✓	✗
AlternativeName ⇒ AlternativeName_of	✓	✗	✗	✗
Application ⇒ PlugIn_of ⇒ Specification_of	✓	✗	✗	✗
ProgrammingEnvironment ⇒ Specification_of	✗	✗	✗	✓

B.2 Search By Sentence Similarity

Table 5. Retrieval example based on sentence similarity for RAG. The first row is the query entity. "sim" reflects the cosine similarity of the sentences. No information about the annotated entities are given for retrieval. The example sentence is the same as in Table 4

split	sim	text
test	1.00	To this end , we have developed an automated web - based tool , PhosphOrtholog , which allows batch processing and mapping of large species - specific PTM datasets to compare overlap at a site - specific level .
train	0.99	Here we present a tool , Podbat (Positioning database and analysis tool) , that incorporates data from various sources and allows detailed dissection of the entire range of chromatin modifications simultaneously .
train	0.99	We designed and developed a new method , MSACompro , to synergistically incorporate predicted secondary structure , relative solvent accessibility , and residue - residue contact information into the currently most accurate posterior probability - based MSA methods to improve the accuracy of mult...
train	0.99	In this paper , we present a lossless compression tool , MAFCO , specifically designed to compress MAF (Multiple Alignment Format) files .

C   Prompting Examples

Fig. 1. Software NER Few-Shot prompt (n=2). The shown sample is the same as in the similarity examples in Table 4 and 5.

```
 1 # Task: Exctract entities in a closed NER setting and classify their
        intention.
 2 ## Entity Types (closed setting):
 3 The entity types in this closed setting are: ['Application', '
        ProgrammingEnvironment', 'SoftwarePackageOrPlugin', '
        OperatingSystem', 'SoftwareCoreference']
 4 Do not include version information for the entities.
 5 if no entity of the given entity types in the context return an
        empty list
 6
 7 ## Intention Classification:
 8 For each entity mention classify the intention of the usage in the
        context.
 9 Use only and exact one intention per entity.
10 The intention classes are: ['Usage', 'Deposition', 'Creation', '
        Mention']
11
12 ## The Contexts:
13 The contexts to extract the entities are about software and from
        scientific publications.
14
15 ## Output Format:
16 Return the result in a json list of objects.
17
18 # Examples:
19 ## Context:
20 """We developed a software tool , SNPdetector , for automated
        identification of SNPs and mutations in fluorescence - based
        resequencing reads ."""
21 ## Detected Named Entities with Intention:
22 [ {
23     "text": "SNPdetector",
24     "label": "Application",
25     "intention": "Creation" } ]
26 ## Context:
27 """In this paper we developed a new algorithm called ESPRIT - Forest
         for parallel hierarchical clustering of sequences ."""
28 ## Detected Named Entities with Intention:
29 [ {
30     "text": "ESPRIT␣-␣Forest",
31     "label": "Application",
32     "intention": "Creation" } ]
33 ## Context:
34 """To this end , we have developed an automated web - based tool ,
        PhosphOrtholog , which allows batch processing and mapping of
        large species - specific PTM datasets to compare overlap at a
        site - specific level ."""
35 ## Detected Named Entities with Intention:
```

Fig. 2. Attibutive NER Few-Shot prompt (n=2).

```
 1  # Task:
 2  ## Task Description
 3  Exctract attributive text spans about software related entities in a
         closed attibutive NER setting.
 4
 5  ## The Contexts:
 6  The contexts to extract the attributive text spans are about
        software and from scientific publications.
 7
 8  ## The Software Related Entities:
 9  The software related entities are given as a list of json object
        with mention and entity type.
10  Do not annotate any related software entities.
11
12  ## Attribute Types (closed setting):
13  The attribute types to identify in this closed setting are:
14  ['Version', 'Developer', 'Citation', 'URL', 'Release', '
         Abbreviation', 'License', 'Extension', 'AlternativeName']
15  Keep the "Version" note separate, without overlap with other
        entities or attribute spans.
16  No span overlaps are allowed inbetween attributive spans of to the
        given software related entites.
17  Note that the attribute types must be associated with a software-
        related entity of one of the following types: ['Application', '
        ProgrammingEnvironment', 'PlugIn', 'OperatingSystem', '
        SoftwareCoreference'].
18  To help you, you will receive all software-related entities as json
         for each context.
19
20  ## Output Format:
21  Return the result in a json list of objects.
22
23  # Examples:
24  ## Context:
25  """LAMINA uses the Java Advanced Imaging ( JAI ) package http://java
        .sun.com/javase/technologies/desktop/media/jai/ to support
        common image file formats , which is bundled with the
        installation and hence no additional installation should be
        required ."""
26  ### Detected Software Related Named Entities:
27  [{
28      "text": "LAMINA",
29      "label": "Application"
30  }, {
31      "text": "Java_Advanced_Imaging",
32      "label": "Application"}]
33
34  ### Detected Attributive Text Spans:
35  [{
36      "text": "JAI",
37      "label": "Abbreviation"
38  }, {
39      "text": "http://java.sun.com/javase/technologies/desktop/media/
           jai/",
40      "label": "URL"}]
```

Fig. 3. Single-Choice QA prompt for Relation Annotation (n=2).

```
 1 # Answer the following single-choice questions on sentences from
      scientific publications on the subject of software.
 2 * Please note that there is exactly one answer to each question.
 3 ## Sentence:
 4 """The web service [44] available at http://www.fragit.org enables
      users to upload their structure , fragment it and download the
      resulting input file to GAMESS ."""
 5 ### Question(s):
 6 Which of the statements about the Citation '[44]' is true?
 7 idx_0: '[44]' is the citation of 'web_service'.
 8 idx_1: '[44]' is the citation of 'GAMESS'.
 9
10 Which of the statements about the URL 'http://www.fragit.org' is
      true?
11 idx_2: 'http://www.fragit.org' is the url of 'web_service'.
12 idx_3: 'http://www.fragit.org' is the url of 'GAMESS'.
13
14 ### Answers to the questions about the sentence(s):
15 idx_0: True
16 idx_1: False
17 idx_2: True
18 idx_3: False
19
20 ## Sentence:
21 """ASPASIA , released under the Artistic License ( 2.0 ) , can be
      downloaded from http://www.york.ac.uk/ycil/software ."""
22 ### Question(s):
23 Which of the statements about the Version '2.0' is true?
24 idx_4: '2.0' is the version of 'ASPASIA'.
25 idx_5: '2.0' is the version of 'Artistic_License'.
26
27 Which of the statements about the URL 'http://www.york.ac.uk/ycil/
      software' is true?
28 idx_6: 'http://www.york.ac.uk/ycil/software' is the url of 'ASPASIA'
      .
29 idx_7: 'http://www.york.ac.uk/ycil/software' is the url of 'Artistic
      _License'.
30
31 ### Answers to the questions about the sentence(s):
32 idx_4: False
33 idx_5: True
34 idx_6: True
35 idx_7: False
36
37 ## Sentence:
38 """The web - tool can be accessed from http://www.phosphortholog.com
      / using any typical web browser ( except Internet Explorer ) .
      """
39
40 ### Question(s):
41 Which of the statements about the URL 'http://www.phosphortholog.com
      /' is true?
42 idx_8: 'http://www.phosphortholog.com/' is the url of 'web_-_tool'.
43 idx_9: 'http://www.phosphortholog.com/' is the url of 'Internet_
      Explorer'.
44
45 ### Answers to the questions about the sentence(s):
```

References

1. Beltagy, I., Lo, K., Cohan, A.: Scibert: A pretrained language model for scientific text. In: Conference on Empirical Methods in Natural Language Processing (2019). https://doi.org/10.18653/v1/d19-1371

2. Du, C.F., Cohoon, J., Lopez, P., Howison, J.: Softcite dataset: A dataset of software mentions in biomedical and economic research publications. Journal of the Association for Information Science and Technology **72**, 870 – 884 (2021). https://doi.org/10.1002/asi.24454

3. Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., Wang, H.: Retrieval-augmented generation for large language models: A survey (2024). https://doi.org/10.48550/arXiv.2312.10997

4. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.t., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-augmented generation for knowledge-intensive nlp tasks. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 9459–9474. Curran Associates, Inc. (2020)

5. Otto, W., Zloch, M., Gan, L., Karmakar, S., Dietze, S.: GSAP-NER: A novel task, corpus, and baseline for scholarly entity extraction focused on machine learning models and datasets. In: Bouamor, H., Pino, J., Bali, K. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2023. pp. 8166–8176. Association for Computational Linguistics, Singapore (Dec 2023). https://doi.org/10.18653/v1/2023.findings-emnlp.548, https://aclanthology.org/2023.findings-emnlp.548

6. Saji, A., Matsubara, S.: Extracting information about research resources from scholarly papers. In: International Conference on Asian Digital Libraries. pp. 440–480 (2022). https://doi.org/10.1007/978-3-031-21756-2_35

7. Schindler, D., Bensmann, F., Dietze, S., Krüger, F.: Somesci- a 5 star open data gold standard knowledge graph of software mentions in scientific articles. Proceedings of the 30th ACM International Conference on Information & Knowledge Management pp. 4574–4583 (2021). https://doi.org/10.1145/3459637.3482017

8. Wadhwa, S., Amir, S., Wallace, B.: Revisiting relation extraction in the era of large language models. In: Rogers, A., Boyd-Graber, J., Okazaki, N. (eds.) Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 15566–15589. Association for Computational Linguistics, Toronto, Canada (Jul 2023). https://doi.org/10.18653/v1/2023.acl-long.868, https://aclanthology.org/2023.acl-long.868

9. Wan, Z., Cheng, F., Mao, Z., Liu, Q., Song, H., Li, J., Kurohashi, S.: GPT-RE: In-context learning for relation extraction using large language models. In: Bouamor, H., Pino, J., Bali, K. (eds.) Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. pp. 3534–3547. Association for Computational Linguistics, Singapore (Dec 2023). https://doi.org/10.18653/v1/2023.emnlp-main.214, https://aclanthology.org/2023.emnlp-main.214

10. Wang, S., Sun, X., Li, X., Ouyang, R., Wu, F., Zhang, T., Li, J., Wang, G.: Gpt-ner: Named entity recognition via large language models (2023). https://doi.org/10.48550/arXiv.2304.10428

11. Xie, T., Li, Q., Zhang, Y., Liu, Z., Wang, H.: Self-improving for zero-shot named entity recognition with large language models (2024). https://doi.org/10.48550/arxiv.2311.08921

12. Xu, D., Chen, W., Peng, W., Zhang, C., Xu, T., Zhao, X., Wu, X., Zheng, Y., Chen, E.: Large language models for generative information extraction: A survey. ArXiv (2023). https://doi.org/10.48550/arxiv.2312.17617