On-the-fly Definition Augmentation of LLMs for Biomedical NER

Anonymous ACL submission

Abstract

Despite their general success, LLMs still lag behind on biomedical named entity recognition (NER) tasks, which are difficult due to the presence of specialized terminology and lack of training data. In this work we set out to improve LLM performance on biomedical NER in limited data settings via: (i) A new knowledge augmentation approach which incorporates definitions of relevant concepts on-the-fly, and (ii) A comprehensive exploration of prompting strategies. Our experiments show that the proposed definition augmentation approach is useful for both open source and closed LLMs. For example, it increases GPT-4 performance (F1) by 15% on average across all (six) of our test datasets. We conduct extensive ablations and analyses to demonstrate that these performance improvements stem from adding relevant knowledge about definitions. We find that careful prompting strategies also improve LLM scores, allowing them to outperform fine-tuned language models in few-shot settings. To facilitate future research in this direction, we plan to release our code upon acceptance.

1 Introduction

LLMs have achieved remarkable success on a wide range of tasks and domains, even in zero-shot and few-shot settings (Brown et al., 2020). However, their performance on named entity recognition (NER) in biomedical text remains underwhelming. For instance, Gutiérrez et al. (2022) observe that GPT-3 in-context learning significantly underperforms compared to fine-tuning a smaller pretrained language model (PLM) on the same amount of data. Despite significant real-world utility, several aspects make this task challenging even for state-of-the-art LLMs. Biomedical texts contain a large proportion of specialized terminology that requires domain expertise to interpret. Additionally, this requirement for domain expertise makes annotation time-consuming and difficult to acquire, leading to limited availability of labeled data.

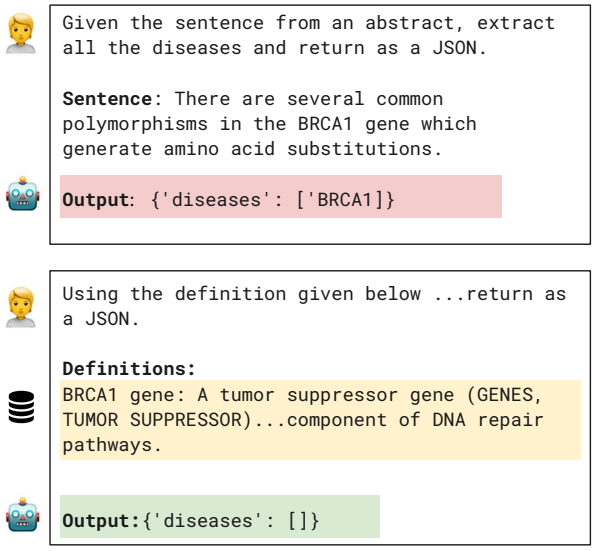


Figure 1: Illustration of our approach using a zero-shot example, with incorrect extraction (red) and correct extraction (green) when provided with the definition of the extracted entity (yellow).

Recently introduced LLMs have shown promising improvements in performance on general information extraction tasks (Ashok and Lipton, 2023; Wadhwa et al., 2023). Motivated by this, we aim to improve LLM-based biomedical NER via two approaches: (i) A new knowledge augmentation approach incorporating relevant concept definitions on-the-fly, and (ii) An exploration of prompting strategies that have demonstrated utility in other IE tasks, establishing a strong baseline to test definition augmentation.

To conduct this exploration, we first design an experimental framework for assessment of LLMs on biomedical NER (§ 2). Starting from the BigBIO (Fries et al., 2022) collection of 100+ biomedical datasets, we systematically construct an evaluation testbed consisting of six NER datasets, which cover extraction tasks of varying complexity ranging from open extraction (i.e., no entity types) to

extraction according to large, fine-grained schemas (10+ entity types). We use this testbed to benchmark the performance of a series of SOTA LLMs, both open and closed, on biomedical NER in both zero-shot and few-shot settings (§ 3).

Our benchmarking effort involves extensive exploration of a host of prompting strategies which have provided utility in recent work on using LLMs for information extraction such as using definitions/explanations (Ashok and Lipton, 2023; Wadhwa et al., 2023) and producing extractions in structured formats like code (Dunn et al., 2022; Li et al., 2023b). To the best of our knowledge, our work is the first to conduct such exploration for biomedical NER with promising results; we find that these strategies enable LLMs to surpass smaller fine-tuned language models in few-shot settings, in contrast to prior work.

Building on these strong baselines, we propose a knowledge augmentation approach to further improve LLM performance. Our approach, illustrated in Figure 1, focuses on identifying and providing definitions of relevant biomedical concepts as a *follow-up* step at inference time, allowing the model to correct its entity extractions.

We explore two strategies for follow-up prompting: (i) single-turn, which requires models to make all entity corrections within a single step, and (ii) iterative prompting, which simplifies the correction task by allowing models to make changes one entity at a time. Our results show that definition augmentation provides meaningful performance improvements on both closed and open SOTA LLMs. For example, including definitions increases GPT-4 performance by 15% on average across the datasets we use for evaluation.

We also verify that these performance improvements are due to the presence of relevant concept definitions by conducting a series of ablations adding irrelevant definition knowledge, which result in little to no performance improvement. Finally, we evaluate the utility of definitions retrieved from various human-curated sources (UMLS, WikiData) as well as ones automatically generated using LLMs, and find that human-curated definitions lead to higher performance improvements. Our results raise interesting questions about the value of definition knowledge in improving LLM performance on various tasks and domains and indicate that LLMs have made substantial advancement on IE tasks in limited data settings.

2 Experimental Framework

Models We evaluate SOTA LLMs over a set of biomedical NER datasets from the BigBio benchmark (Fries et al., 2022). We assess a variety of models including closed models available via API—i.e., Open AI's GPT 3.5 (Brown et al., 2020) and GPT 4 (OpenAI, 2023) and Anthropic's Claude 2 (Anthropic, 2023)—and an open-source model (Llama 2; Touvron et al. 2023). We enumerate these models in Table 12. We also conducted preliminary experiments with Google's PaLM (Chowdhery et al., 2022) but found its performance subpar and so did not pursue further.

Evaluation We evaluate all models with entity-level F1. Prior work has shown that strict F1 may underestimate the performance of generative models on information extraction tasks, because such models can generate outputs that differ from reference annotations but which are still correct (Wadhwa et al., 2023). To address this, we complement our automatic evaluation with manual evaluation of a subset of examples presented in Appendix C.

Dataset	Entity Types	Size
CHEM (Krallinger et al., 2017)	Chemicals, Proteins	800
CDR (Li et al., 2016)	Chemicals, Diseases	500
NCBI (Doğan et al., 2014)	Diseases	100
MEDM (Mohan and Li, 2019)	Biomedical Concepts	879
PICO (Nye et al., 2018)	Populations, Interventions, Outcomes	187
CHIA (Kury et al., 2020)	Clinical Trial Criteria	600

Table 1: Overview of all datasets included in our final biomedical NER evaluation testbed. The size column reports the size of the test split.

Dataset Selection As a testbed for biomedical NER, we select datasets from the BigBIO benchmark, a meta-resource of 100+ datasets sourced from various areas of biomedicine, covering 12 task types and 10+ languages. NER is the dominant task category in BigBIO, consisting of 76 datasets (Fries et al., 2022). We narrow these down by first excluding datasets that contain: Clinical/EHR data, social media content, and non-English texts.

Several of the remaining datasets contain annotations for the same entity types. Therefore, we further filter the corpora by retaining only 1-2 *rep-*

resentative datasets for all entity types. We discarded datasets from especially narrow/specialized domains (e.g., stem cell identification) and kept datasets which are part of existing benchmarks, e.g., BLURB (Gu et al., 2021), BLUE (Peng et al., 2019) and BoX (Parmar et al., 2022).

This filtering yields 16 datasets, out of which we manually select six for our experiments. These datasets are summarized in Table 1 and further described in Table 13 (including examples).

3 ICL for Biomedical NER

In this section we establish the baseline performance of LLMs in zero- and few-shot settings over all datasets. To contextualize these results, we also report on the performance of a smaller, fine-tuned model (Flan-T5-XL; Chung et al. 2022).

3.1 Zero-Shot Experimental Setup

We evaluate zero-shot prompting strategies along two main axes: (i) Input format (i.e., prompt template), which controls how the task description and expected target categories are provided to the model; (ii) Output format, which controls how the model structures outputs (i.e., text, code or JSON).

We explore two possible types of input format: (i) **Text**, using a standard prompt with a brief description of the task and a list of valid target entity types to be extracted; and (ii) **Schema Def**, augmenting the standard prompt with detailed descriptions of all target entity types following Ashok and Lipton (2023); Shao et al. (2023).

For output format, we explore two types of *structured* formats: (i) **JSON** (Dunn et al., 2022; Li et al., 2023a), and (ii) **Code** snippets (Li et al., 2023b; Wang et al., 2023a). Recent work has shown that such formats improve zero-shot IE performance of LLMs, while producing valid extractions which are easier to post-process and evaluate.

Our zero-shot experiments evaluate the performance of all four combinations of input and output formats on all models to determine the best prompting strategy (except GPT-4, omitted in these experiments given the high costs of querying the API). Example prompts for each combination are presented in Appendix 5.

3.2 Few-Shot Experimental Setup

For our few-shot experiments, we adopt the combination of input/output formats that performed the best for each dataset in the zero-shot setting. We validated this decision by evaluating all combinations of input/output formats on one of the datasets (i.e., CDR) and observing that the best performing format in zero-shot also applies to the few-shot setting (for $k = \{1, 3, 5\}$). These results are shown in Table 7 of the Appendix B.1.

In addition to input/output formats, few-shot prompting can also vary along two axes: (i) selection of few-shot exemplars; and (ii) ordering of chosen exemplars. For the former, we compared selection of few-shot exemplars at random to the similarity-based approach due to (Gutiérrez et al., 2022). For the latter, we compared passing exemplars in a random but fixed order against shuffling exemplars per test instance. In preliminary experiments, we did not observe meaningful differences in performance based on these strategies, therefore we carried the rest of the experiments with randomly selected exemplars shuffled per test instance. See Appendix B.2 for additional details on these few-shot prompting strategies.

We test the performance of all models for $k = \{1, 3, 5\}$. For each setting, we conduct three runs with different seeds and report the average performance (additional results for larger values of k are provided in Figure 3).

3.3 Fine-tuning Experimental Setup

To put our results in context, we also measure the performance of a smaller language model fine-tuned on the each of the datasets. Specifically, we fine-tune Flan-T5-XL on linearized targets. We train the model on the same set of 5 instances used in the few-shot experiments using LoRA, a parameter efficient fine-tuning method (Hu et al., 2021). We provide implementation details in E.

3.4 Results

In preliminary experiments, we observed that Claude 2 was unable to generate valid code outputs so we only report results for JSON outputs. In regards to input formats, we see that prompts augmented with schema definitions perform worse across all models and datasets. As for output formats, we find that JSON was preferred on most datasets with the exception of PICO and CHIA. However, this observation holds consistently across all models. See Table 2 for the results of GPT-3.5, Claude 2 and LLama 2 on all datasets.

Given these findings, we executed few-shot experiments using plain prompts and JSON outputs (aside from PICO and CHIA, for which we used

Model	Input	Output	CHEM	CDR	MEDM	NCBI	PICO	CHIA
GPT3.5	Text	JSON	49.60	65.64	**43.42**	**54.05**	10.71	7.43
		Code	42.31	50.72	42.91	44.23	14.88	**31.28**
	+ Schema Def	JSON	47.70	64.74	**43.72**	46.79	9.53	4.72
		Code	41.49	51.16	42.46	47.13	13.52	29.43
Claude 2	Text	JSON	56.36	**67.96**	36.39	44.17	7.70	19.96
	+Schema Def	JSON	45.19	60.51	34.30	37.93	4.81	19.11
LLaMA2	Text	JSON	**59.75**	66.77	28.93	34.23	7.49	4.03
		Code	57.53	55.18	23.69	24.64	**15.39**	21.59
	+Schema Def	JSON	52.47	55.47	23.05	28.22	3.95	3.32
		Code	56.04	54.91	28.82	24.05	15.12	7.49

Table 2: Zero Shot scores with *text input, JSON output*, *text input and code output*, *definition input and JSON output* and *definition input and code output*, with an exception of Claude 2 which we experimented on JSON (did not output executable code).

Model	#Shots	CHEM	CDR	MEDM	NCBI	PICO	CHIA
GPT3.5	0	49.60	65.64	43.42	54.05	14.88	31.28
	1	56.06 (± 1.03)	64.05 (± 2.92)	49.15 (± 1.69)	44.27 (± 2.59)	15.83 (± 1.9)	33.72 (±0.99)
	3	59.54 (± 2.24)	67.44 (± 0.52)	48.47 (± 1.63)	54.20 (± 1.53)	17.11 (± 1.65)	34.8 (±0.65)
	5	58.66 (± 0.79)	68.19 (± 1.07)	48.10 (± 1.28)	56.02 (± 1.48)	17.12 (±3.83)	36.47 (±0.6)
Claude 2	0	56.36	67.96	36.39	44.17	7.70	19.96
	1	55.19 (± 2.21)	66.43 (± 3.08)	44.82 (± 3.04)	37.89 (± 13.42)	6.3 (± 1.2)	18.94 (± 1.43)
	3	59.68 (± 1.61)	68.13 (± 6.01)	48.20 (± 1.91)	43.89 (± 1.63)	6.21 (± 2.6)	19.87 (± 3.41)
	5	63.04 (± 0.21)	69.74 (± 1.47)	48.12 (± 1.45)	42.99 (± 1.59)	6.12 (± (8.21)	19.88 (± 1.63)
LLaMa 2	0	59.75	66.77	28.93	34.23	15.39	21.59
	1	57.11 (± 1.73)	54.77 (± 12.23)	45.04 (± 1.07)	37.88 (± 14.05)	12.95 (±1.49)	24.1 (±2.75)
	3	55.23 (± 4.94)	64.76 (± 0.99)	45.25 (± 1.51)	45.08 (± 6.17)	17.08 (±1.32)	32.78 (±1.79)
	5	59.86 (± 0.93)	64.89 (± 1.63)	47.37 (± 1.33)	46.96 (± 3.75)	18.26 (±0.91)	35.44 (±1.85)
Flan-T5	5	30.32 (±6.62)	29.33 (±1.8)	38.84 (±4.23)	30.68 (±12.53)	14.74 (±6.78)	4.84 (±1.32)

Table 3: Few shot scores with $k = \{1, 3\ and\ 5\}$. We ran experiments with 3 seeds and averaged the results. Results show F1 scores and standard deviation. We have chosen the format that works best for each dataset. CHEM, CDR, MEDM, NCBI on *text input, JSON output* and PICO and CHIA with *text input and code output*, with an exception of Claude 2 which we experimented on JSON.

code outputs). As we can see in Table 3, model performance tends to increase with the number of shots (except for NCBI and MEDM datasets, where we observe minor fluctuations in performance). Finally, we see that few-shot learning with instruction tuned LLMs performs much better than a smaller LM fine-tuned on the same 5 instances.

4 Augmenting Prompts with Definitions

ICL approaches rely on the parametric knowledge acquired by the models during pre-training. However, this internal knowledge can be incorrect, insufficient or outdated. Prior work has tried to address knowledge gaps in LLMs by augmenting prompts with relevant factual knowledge *on-the-fly*, improving performance on language understanding tasks like question answering (Baek et al., 2023; Wang et al., 2023b).

This motivates us to explore whether augmenting prompts with relevant knowledge dynamically improves ICL performance for biomedical NER. In our work, we focus on a specific category of knowledge — *definitions of biomedical concepts* present in the input text. Intuitively, generic LLMs may not be proficient with biomedical concepts; providing targeted information at test time may permit fast adaptation to this domain.

We propose to operationalize this approach as follows. First, we curate a knowledge base of biomedical concept definitions and leverage an

Model	Setting	CHEM	CDR	MedM	NCBI	PICO	CHIA
GPT3.5	**ZS**	48.61	67.65	43.77	54.05	10.25	7.50
	+Def	48.34 (-0.27)	68.21 (+0.56)	45.00 (+1.23)	51.94 (-2.11)	10.20 (-0.05)	7.95 (+0.45)
	IP	47.27 (-1.34)	66.12 (-1.53)	42.71 (-1.06)	51.18 (-2.87)	10.27 (+0.02)	7.59 (+0.09)
	+Def	56.39 (+7.78)	72.86 (+5.21)	50.05 (+6.28)	58.24 (+4.19)	9.88 (-0.37)	17.64 (+10.14)
Claude 2	**ZS**	54.28	70.07	36.98	44.17	7.26	20.12
	+Def	57.62 (+3.34)	68.91 (-1.16)	36.12 (-0.86)	43.65 (-0.52)	7.67 (+0.41)	19.17 (-0.95)
	IP	52.93 (-1.35)	69.34 (-0.73)	36.71 (-0.27)	43.43 (-0.74)	7.66 (+0.40)	19.82 (-0.30)
	+Def	59.96 (+5.68)	73.04 (+2.97)	41.82 (+4.84)	51.60 (+7.43)	8.98 (+1.72)	22.12 (+2.00)
LLaMA2	**ZS**	60.30	64.07	25.98	47.38	7.88	4.24
	+Def	67.49 (+7.19)	68.54 (+4.47)	35.56 (+9.58)	51.44 (+4.06)	8.54 (+0.66)	9.50 (+5.26)
	IP	58.31 (-1.99)	65.63 (-1.56)	24.54 (-1.44)	45.58 (-1.80)	7.49 (-0.39)	4.50 (+0.26)
	+Def	67.54 (+7.24)	69.05 (+4.98)	34.90 (+8.92)	50.57 (+3.19)	9.59 (+1.71)	9.42 (+5.18)
GPT4	**ZS**	62.12	70.92	47.13	54.67	7.29	16.39
	+Def	67.05 (+4.93)	76.19 (+5.27)	51.91 (+4.78)	60.91 (+6.24)	9.24 (+1.95)	20.88 (+4.49)
	IP	59.67 (-2.45)	69.41 (-1.51)	47.01 (-0.12)	52.31 (-2.36)	7.47 (+0.18)	17.94 (+1.55)
	+Def	65.39 (+3.27)	75.62 (+4.70)	52.13 (+5.00)	58.72 (+4.05)	9.47 (+2.18)	20.09 (+3.70)

Table 4: Zero shot (ZS) scores with Definition Augmentation (+Def), Iterative Prompting (IP) and Iterative Prompting augmented with Definitions (+Def) on four models. Results show F1 scores and the delta wrt zero-shot in the parenthesis.

Model	Setting	CHEM	CDR	MedM	NCBI	PICO	CHIA
GPT3.5	**FS**	57.92 (± 0.78)	68.89 (± 1.03)	49.08 (± 01.33)	56.02 (± 1.48)	11.07 (± 1.77)	21.72 (± 1.23)
	+Def	59.23 (± 1.54)	68.7 (± 2.47)	48.41 (± 0.77)	57.6 (± 2.75)	11.19 (± 0.52)	22.15 (± 1.03)
Claude 2	**FS**	61.6 (± 0.36)	71.95 (± 2.62)	48.3 (± 1.44)	44.92 (± 1.62)	6.2 (± 2.83)	19.72 (± 2.94)
	+Def	61.17 (± 0.26)	72.81 (± 1.58)	49.32 (± 1.36)	48.98 (± 1.51)	9.97 (± 2.13)	22.21 (± 1.03)
LLaMA2	**FS**	60.15 (± 0.92)	66.77 (± 1.32)	38.92 (± 11.83)	47.97 (± 3.65)	8.0 (±1.98)	9.32 (± 0.45)
	+Def	59.86 (± 0.93)	64.89 (± 1.63)	47.37 (± 1.33)	46.96 (± 3.75)	18.26 (± 0.91)	35.44 (± 1.85)
GPT4	**FS**	64.92 (± 1.28)	74.23 (± 3.48)	54.59 (± 1.89)	62.28 (± 1.97)	8.74 (± 1.68)	23.21 (± 1.60)
	+Def	69.72 (± 0.68)	79.63 (± 2.96)	59.17 (± 1.5)	66.21 (± 0.96)	7.63 (± 0.58)	24.51 (± 0.77)

Table 5: Few shot scores with Definition Augmentation (+Def) with $k = 5$. We ran experiments with 3 seeds and averaged the results. Results show F1 scores and standard deviation in the parenthesis.

off-the-shelf entity linker to map occurrences of concepts to entries in the knowledge base (§4.1). Second, we perform inference with a sequence of prompts: first, we prompt models to extract entities as discussed in §3; then, we craft follow-up prompts augmented with concept definitions and asking the model to revise the initial extractions, which can include removing/adding entities or reassigning entity types. We provide definitions for all the entities identified by the model in the first turn, and all other biomedical concepts that can be linked to the knowledge base (as identified by the entity linker). We evaluate this approach in zero-shot (§4.2) and few-shot (§4.3) settings.

4.1 Concept Definitions

We obtain concept definitions from Unified Medical Language System (UMLS), a collection of key terminology and coding standards from several biomedical vocabularies, standards and knowledge bases (Bodenreider, 2004). Some concepts in UMLS belong to fairly broad categories (e.g., event, activity, group) and their definitions might not provide much utility to LLMs. We avoid including definitions for such concepts by curating a set of fine-grained categories which contain specific and useful concepts. The final set of categories used for all definition augmentation experiments is listed in Table 15. At inference time, we use the entity linker available in the SciSpaCy package (Neumann et al., 2019) to map all mentions of biomedical concepts in the input text to entries in UMLS, and retrieve the associated definitions.

4.2 Zero-Shot Definition Augmentation

In the zero-shot setting, we first prompt the model to extract entities as described in §3.1. Then we consider two strategies for follow-up prompting.

Single-turn (ZS+Def): A single definition augmented follow-up prompt asks the model to make corrections to all extracted entities.

Iterative Prompting (IP+Def): iterative prompts augmented with the definition of a single concept and asking the model to make corrections to a single extracted entity (if needed) at a time. This breaks down the correction process into atomic steps, but significantly increases the number inference steps (which incurs additional costs when using proprietary models). This approach is related to prior work suggesting that LLMs are able to correct and revise their own outputs and this self-verification can improve performance in clinical information extraction tasks (Gero et al., 2023). The novelty on offer here is providing contextual knowledge to aid the process of self-verification. In our experiments, we ablate the impact of self-verification from that of the concept definitions.

4.3 Few-Shot Definition Augmentation

In the few-shot setting, again we first prompt the model to extract entities as described in §3.2, and then ask it to correct the extractions in a follow-up prompt with concept definitions. The follow-up prompt includes: (i) all few-shot exemplars provided in the first prompt along with the associated concept definitions; and (ii) definitions for all the concepts identified in the current input (both for extracted entities and other biomedical concepts).

Here, we only test the single-turn strategy because including few-shot examples rapidly increases context size, making iterative prompting very expensive.

4.4 Definition Augmentation Results

All the experiments are carried out with JSON outputs to maintain a uniform experimental setting across all datasets. The few-shot experiments are all carried with $k = 5$ shots randomly selected and shuffled per test instance. We run each experiment with three different random seeds and report the average performance. In addition to the models considered in the previous section, here we also evaluate GPT-4 — this is motivated by prior suggesting that GPT-4 is more competent that GPT-3.5 at editing previous outputs, which is a key step of our proposed approach (Gero et al., 2023). However, given the high costs of querying the API, we subsampled our test sets to 100 instances in the experiments with this model.

Tables 4 and 5 present the performance of GPT3.5, Claude 2 and Llama 2 and GPT-4 with definition augmentation on all datasets in the zero- and few-shot settings, respectively. In zero-shot settings, we see consistent and significant improvements in the performance of Llama 2 and GPT-4 with both prompting strategies.

We see an average increase of 32.6% and 33.9% for Llama 2 and 15% and 13.7% for GPT-4 on single turn and iterative prompting, respectively. However, Claude 2 and GPT-3.5 can only benefit from the iterative prompting approach with average gains of 12% and 29.5%, respectively. We also assessed the performance of iterative prompting but *without* the definitions - this is similar to the (Gero et al., 2023) self-verification method. However, our results show that the models are not able to correct their predictions in the absence of the definitions.

In the few-shot setting, we also see improvements in most cases. Claude 2 and GPT-4 improve in 5 out of 6 datasets; Llama 2 and GPT-3.5 show gains in 3 and 4 datasets, respectively. Overall, we found that GPT-4 with iterative prompting achieves the best performance.

Our results show that concept definition augmented prompts improve the performance of biomedical NER. A key step of this approach is linking biomedical concepts to definitions in UMLS. One natural question is how much of the observed gains are simply due to the use of an entity linking model which was explicitly trained to recognize entities. To answer this question, we measured the performance of the entity linker by itself on the same test sets and found that it performs poorly, with an average F1 of 1.05 across all the datasets.

5 Assessing the Utility of Definition Knowledge

We further assess the utility of concept definitions by conducting ablation experiments probing the following dimensions: (1) Relevance of the concept definitions; (2) Source of the knowledge base.

We conduct all experiments in the single-turn

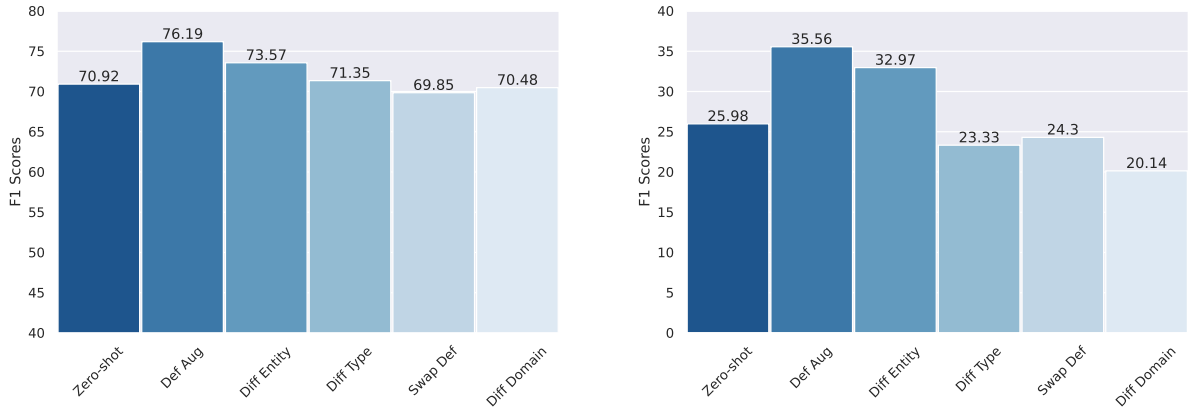


Figure 2: Definition relevance ablations with GPT-4 on CDR dataset (left) and LLaMa2 on MEDM dataset (right). We see similar trends across all models and datasets - a consistent decrease in performance with less relevant definitions.

zero-shot setting (§4.2), with one closed model (GPT-4) and one open-source model (Llama 2), over the two datasets with the largest gains in performance from concept definitions (CDR and NCBI for GPT4; MEDM and CHIA for Llama 2).

5.1 Probing Definition Relevance

Motivated by prior work showing that LLMs often produce correct predictions even with misleading or irrelevant prompts (Webson and Pavlick, 2022), we ablate over the *relevance* of definitions provided for a given entity. This allows us to assess whether performance gains are due to accurate definitions or simply from additional context, irrespective of relevance. To this end, we measure the performance of increasingly *less* relevant knowledge by swapping out various components of provided definitions. These ablations are realized as follows.

Diff Entity include definitions of concepts mentioned in a different instance (within the same dataset). As this samples instances in the same dataset, it will include concepts from the same entity types being extracted (e.g., for NCBI, the swapped concepts will include some diseases).

Diff Type include definitions from concepts mentioned in a different instance within the same dataset, but exclude concepts from the entity types being extracted (e.g., for NCBI, add all swapped concepts that are not diseases).

Swap Def replace definitions for all concepts mentioned in the current instance with random incorrect definitions (e.g., for NCBI, if the disease extracted is Arrhythmia, we provide a an incorrect definition for Arrhythmia).

Diff Domain include definitions for concepts mentioned in an instance from a *different domain*. For instance, for datasets containing Pubmed abstracts (MEDM), we add concepts mentioned in a dataset of clinical trial criteria (CHIA) and vice versa.

Figure 2 shows the performance of GPT-4 and Llama 2, with different definition relevance ablations, respectively on CDR (left) and MEDM (right). See **??** and **??** for plots with NCBI and CHIA datasets. We see similar trends across all models and datasets: A consistent decrease in performance with less relevant definitions. This provides evidence that the model is indeed capitalizing on the definitions and suggests that the quality of the definitions plays a critical role on our proposed method. Interestingly, we observe that augmenting prompts with definitions of other entities (of the same type) also yields consistent gains across models and datasets. A possible explanation is that since the entities are of the same type, they may be similar enough that model can still learn from their definitions. Finally, we do observe some gains from definitions of entities of a different type, but these are smaller and less consistent.

5.2 Probing Definition Sources

After establishing that the success of our approach is largely due to adding relevant definition knowledge, we assess the impact of the *source* of definitional knowledge. We evaluate the same models and datasets as in the previous experiments but using concept definitions: (i) collected from Wikidata; and (ii) automatically generated by GPT-4.

Table 6 shows the results for all models and data sources. We observe that definitions from Wiki-

Setting	CDR	NCBI	MEDM	CHIA
ZS	70.92	54.67	25.98	4.24
+UMLS	76.19	60.91	35.56	9.50
+Wiki	72.9	57.5	32.6	9.53
+GPT4	69.24	54.83	25.29	7.32

Table 6: Ablations with GPT-4 [CDR, NCBI] and LLaMa 2 [MEDM, CHIA], providing definitions from different sources. Original source being **UMLS** and ablations with Wikipedia and GPT 4 generated definitions.

data also improve over the zero-shot baseline, albeit to a lesser degree than UMLS. On the other hand, the definitions generated by GPT-4 seem to have little to no impact on the model's performance. These results again highlight the importance of the knowledge source: we see larger improvements with concept definitions from a more domain specific source. However, seeing that models can also benefit from concept definitions from more general sources such as Wikidata, suggests that our proposed approach may also be suitable for applications in other, less specialized, domains.

6 Related Work

Information Extraction with LLMs Recent work has shown that LLMs are capable of extracting information from documents in zero- and few-shot settings. For instance, (Agrawal et al., 2022) found that GPT-3 competes with or outperforms smaller models on a small set of clinical tasks extraction tasks. However, in the scientific and biomedical domain, LLMs were lagging in the performance as compared to their pretrained and fine-tuned counterparts (Gutiérrez et al., 2022). GPT-3's ICL (Brown et al., 2020) compares favorably to supervised models in several tasks (e.g., NLI, text classification, machine translation (Liu et al., 2022)). Several methods have been introduced to improve its performance, optimizing prompt retrieval (Shin et al., 2021), ordering (Lu et al., 2022), and design (Perez et al., 2021).

Iterative Prompting with LLMs Recent works including (Gero et al., 2023) explores self verification as a strategy to improve the performance on IE tasks. This builds on prior works (Wu et al., 2022) and (Wang et al., 2022) that iteratively prompt LLMs to improve their performance. In recent work, (Gero et al., 2023), the authors performed clinical information retrieval along with self verification and grounding the extraction with LLMs for clinical information extraction.

Knowledge Augmentation with LLMs Prior to LLMs, REALM (Guu et al., 2020) and RAG (Lewis et al., 2021) proposed to integrate the knowledge, retrieve documents such as documents from unstructured corpora (e.g., Wikipedia) and facts from Knowledge Graphs (KGs), into LMs. With adding this information to these methods the accuracy improves. Recently, concurrent to our work, (Nori et al., 2023) explores iterative prompting with knowledge augmentation in clinical domain. Their prompting strategy combines kNN-based few-shot example selection, GPT-4–generated chain-of-thought prompting, and answer-choice shuffled ensembling reduces the error rate by 27% medical question answering (MedQA) dataset.

7 Conclusions

In this work, we extensively evaluated the performance of ICL approaches for biomedical NER with modern LLMs. We compared different combinations of input and output formats and characterized the main types of errors made by these models. Then, we proposed and evaluated a method for rapid adaptation of general LLMs to biomedical NER by providing models with concept definitions from an external knowledge base dynamically.

We perform inference with a sequence of prompts which allows models to revise their predictions given definitions of key concepts in the input. The first prompt asks the model to extract entities from the input; and the subsequent prompts are augmented with definitions for all biomedical concepts including the entities identified in the first prompt, and ask the model to revise its predictions.

Our evaluation, conducted over 6 datasets, showed consistent and often substantial improvements over baselines, especially in zero-shot settings. Ablation experiments confirm that the observed gains stem from the models' ability to capitalize on the concept definitions. In particular, we observe that without these definitions the models are unable to meaningfully improve their predictions.

While we only considered datasets from a specialized domain (biomedicine), our ablations show that our approach can also be used with more general knowledge bases, such as Wikidata. This provides some evidence for the potential utility of this approach in other domains. We leave a thorough exploration of this for future work.

8 Limitations

Since our work evaluates LLMs trained on undisclosed data sources, it is possible that the models have encountered parts of our evaluation sets during pre-training or instruction tuning. The underlying text corpora for all datasets in our NER evaluation testbed are sourced from easily accessible text collections (e.g., PubMed, AACT) and so it is quite likely that these have been seen by models during pre-training. However, this is not a major issue in the case of NER, because simply training on these sentences with a language modeling objective does not provide any indication of which words are named entities. Consequently, our primary concern is potential exposure of *label information* from these datasets during some form of entity-aware training or instruction tuning phase. To assess this, we provide models with the raw text and some entity labels and test whether they are able to correctly produce the remaining entities in the original format. We observe that all models failed at this, indicating that though we cannot make strong claims about data contamination, it is unlikely that models have accurately memorized these test sets.

Another limitation of our work is that we only evaluate on biomedical NER and do not test how well our approach would work for other tasks or domains. Additionally, we rely on the availability of expert-curated knowledge (UMLS) for biomedicine — however, such resources may not be readily available for for other tasks or domains. Even within biomedical NER, we test our approach on a limited number of datasets due to the experimental costs of testing proprietary LLMs, and it is possible that our approach may not work for other datasets.

Finally, current metrics for IE tasks are not well-suited to generative models. We mitigate this by performing additional human evaluation, but this approach is not scalable.

References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are few-shot clinical information extractors.

Anthropic. 2023. Anthropic. introducing claude 2, 2023. https://www.anthropic.com/index/claude-2. Accessed: 2023-07-11.

Dhananjay Ashok and Zachary Chase Lipton. 2023. Promptner: Prompting for named entity recognition. *ArXiv*, abs/2305.15444.

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136*.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. 2014. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10.

Alexander Dunn, John Dagdelen, Nicholas Walker, Sanghoon Lee, Andrew S Rosen, Gerbrand Ceder,

Kristin Persson, and Anubhav Jain. 2022. Structured information extraction from complex scientific text with fine-tuned large language models. *arXiv preprint arXiv:2212.05238*.

Jason Fries, Leon Weber, Natasha Seelam, Gabriel Altay, Debajyoti Datta, Samuele Garda, Sunny Kang, Rosaline Su, Wojciech Kusa, Samuel Cahyawijaya, et al. 2022. Bigbio: a framework for data-centric biomedical natural language processing. *Advances in Neural Information Processing Systems*, 35:25792–25806.

Zelalem Gero, Chandan Singh, Hao Cheng, Tristan Naumann, Michel Galley, Jianfeng Gao, and Hoifung Poon. 2023. Self-verification improves few-shot clinical information extraction. *ArXiv*, abs/2306.00024.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

Bernal Jiménez Gutiérrez, Nikolas McNeal, Clay Washington, You Chen, Lang Li, Huan Sun, and Yu Su. 2022. Thinking about gpt-3 in-context learning for biomedical ie? think again.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training.

SU Hongjin, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. 2022. Selective annotation makes language models better few-shot learners. In *The Eleventh International Conference on Learning Representations*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Martin Krallinger, Obdulia Rabal, Saber A Akhondi, Martın Pérez Pérez, Jesús Santamaría, Gael Pérez Rodríguez, Georgios Tsatsaronis, Ander Intxaurrondo, José Antonio López, Umesh Nandal, et al. 2017. Overview of the biocreative vi chemical-protein interaction track. In *Proceedings of the sixth BioCreative challenge evaluation workshop*, volume 1, pages 141–146.

Fabr'ıcio Kury, Alex Butler, Chi Yuan, Li-heng Fu, Yingcheng Sun, Hao Liu, Ida Sim, Simona Carini, and Chunhua Weng. 2020. Chia, a large annotated corpus of clinical trial eligibility criteria. *Scientific data*, 7(1):1–11.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Bo Li, Gexiang Fang, Yang Yang, Quansen Wang, Wei Ye, Wen Zhao, and Shikun Zhang. 2023a. Evaluating chatgpt's information extraction capabilities: An assessment of performance, explainability, calibration, and faithfulness. *arXiv preprint arXiv:2304.11633*.

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 J. Biol. Databases Curation*, 2016.

Peng Li, Tianxiang Sun, Qiong Tang, Hang Yan, Yuanbin Wu, Xuanjing Huang, and Xipeng Qiu. 2023b. Codeie: Large code generation models are better few-shot information extractors.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098.

Sunil Mohan and Donghui Li. 2019. Medmentions: A large biomedical corpus annotated with umls concepts. *arXiv preprint arXiv:1902.09476*.

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy. Association for Computational Linguistics.

Harsha Nori, Yin Tat Lee, Sheng Zhang, Dean Carignan, Richard Edgar, Nicolo Fusi, Nicholas King, Jonathan Larson, Yuanzhi Li, Weishung Liu, Renqian Luo, Scott Mayer McKinney, Robert Osazuwa Ness, Hoifung Poon, Tao Qin, Naoto Usuyama, Chris White, and Eric Horvitz. 2023. Can generalist foundation models outcompete special-purpose tuning? case study in medicine.

Benjamin Nye, Junyi Jessy Li, Roma Patel, Yinfei Yang, Iain Marshall, Ani Nenkova, and Byron Wallace.

2018. A corpus with multi-level annotations of patients, interventions and outcomes to support language processing for medical literature. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 197–207, Melbourne, Australia. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report.

Mihir Parmar, Swaroop Mishra, Mirali Purohit, Man Luo, Murad Mohammad, and Chitta Baral. 2022. In-BoXBART: Get instructions into biomedical multi-task learning. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 112–128, Seattle, United States. Association for Computational Linguistics.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, Florence, Italy. Association for Computational Linguistics.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models.

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

Richard Shin, Christopher Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained language models yield few-shot semantic parsers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7699–7715, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Amanpreet Singh, Mike D'Arcy, Arman Cohan, Doug Downey, and Sergey Feldman. 2022. Scirepeval: A multi-format benchmark for scientific document representations. *arXiv preprint arXiv:2211.13308*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Somin Wadhwa, Silvio Amir, and Byron C. Wallace. 2023. Revisiting relation extraction in the era of large language models.

Boshi Wang, Xiang Deng, and Huan Sun. 2022. Iteratively prompt pre-trained language models for chain of thought. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2714–2730, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xingyao Wang, Sha Li, and Heng Ji. 2023a. Code4struct: Code generation for few-shot event structure prediction.

Yubo Wang, Xueguang Ma, and Wenhu Chen. 2023b. Augmenting black-box llms with medical textbooks for clinical question answering. *arXiv preprint arXiv:2309.02233*.

Albert Webson and Ellie Pavlick. 2022. Do prompt-based models really understand the meaning of their prompts? In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2300–2344, Seattle, United States. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Sherry Wu, Hua Shen, Daniel S Weld, Jeffrey Heer, and Marco Tulio Ribeiro. 2023. Scattershot: Interactive in-context example curation for text transformation. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*, pages 353–367.

Tongshuang Wu, Michael Terry, and Carrie J. Cai. 2022. Ai chains: Transparent and controllable human-ai interaction by chaining large language model prompts.

A Input Format

Selection of few-shot examples: Prior work has shown that in-context learning can benefit from sophisticated strategies for selecting exemplars, e.g. based on diversity (Hongjin et al., 2022) or informativeness (Wu et al., 2023) of the samples. We defer a thorough exploration of these strategies to future work, and here focus on two relatively simple approaches: (i) **Random**, where k examples are randomly sampled; and (ii) **Retrieval**, which follows Gutiérrez et al. (2022). The training set is subsampled to 100 examples; then for every test instance, k *most similar* examples are retrieved from this pool. Similarity between examples is computed using SPECTER2 embeddings (Singh et al., 2022).

Ordering of few-shot examples: Prior work has also shown that models can be very sensitive to the order in which examples are provided for in-context learning (e.g., Lu et al. (2022)), thus we compared two ordering criteria: (i) **Fixed order**, chosen at random; and (ii) **Shuffled order** of examples per test instance. Note that for the retrieval-based shot selection, examples are provided in decreasing order of similarity (Gutiérrez et al., 2022).

B Ablations

B.1 Best output format in Few Shot

Ablation experiment testing multiple format combinations on CDR with k=1, 3 and 5 shots. We use text as the input format as this was the best performing over def prompts across all models and all datasets.

Setting	K	CDR
JSON	1	64.35
	3	65.98
	5	66.26
Code	1	56.17
	3	60.26
	5	60.56

Table 7: Few-shot JSON input and code output ablations. Results show F1 scores. We evaluate combinations of input/output formats on CDR dataset and observe that the best performing format in zero-shot also applies to the few-shot setting.

B.2 Ordering shots in Few Shot

Ablations testing example selection and ordering strategies on CDR with k=1, 3 and 5 shots.

- **Random:** Fixed order of k examples are randomly sampled.
- **Retrieval:** For every test instance, k *most similar* examples are retrieved from this pool. Similarity between examples is computed using SPECTER V2 embeddings and examples are provided in decreasing order of similarity.
- **Random + Shuffle:** Shuffling order of examples per test instance where k examples are randomly sampled.

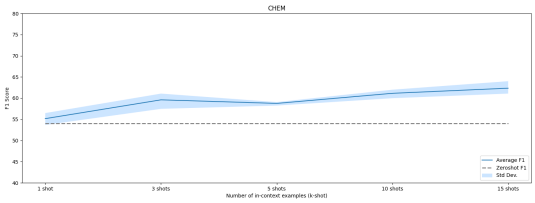
Setting	K	CDR
Random	1	68.25
	3	70.93
	5	72.02
Random + Shuffle	1	68.06
	3	70.29
	5	71.93
Retrieved	1	63.94
	3	71.46
	5	72.22

Table 8: Few-shot shot selection ablations. Results show F1 scores. We do not observe meaningful differences in performance based on these strategies, therefore we carried few-shot experiments with randomly selected exemplars shuffled per test instance.

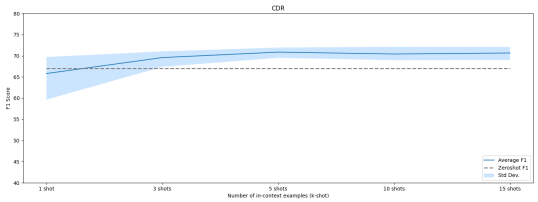
C Qualitative Error Analysis

To better understand the performance of LLMs on biomedical NER and characterize errors these models still make, we conduct a qualitative error analysis of 50 examples from the best performing zero-shot and few-shot models per dataset. This analysis surfaced four major categories of errors:

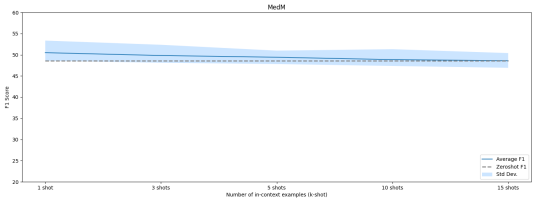
- **Type mismatch:** An entity is extracted correctly but assigned the wrong type.
- **Boundary issues:** The extracted entity is missing terms or contains extra terms when compared to the gold entity.
- **Extra entities:** Model extracts entities which are not present in gold annotations. We observe that these extractions are not always errors either, which motivates the need for human evaluation.
- **Missing entities:** Model does not extract entities present in gold annotation.



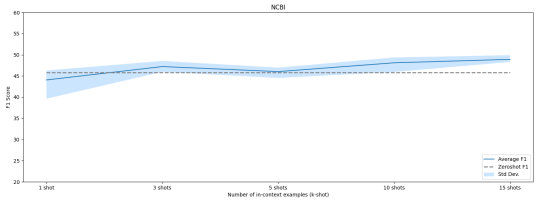
(a) Few shot performance on CHEM



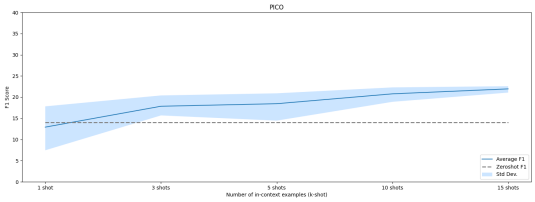
(b) Few shot performance on CDR



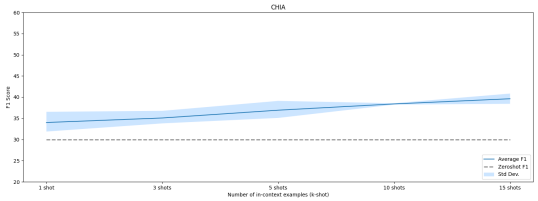
(c) Few shot performance on MedM



(d) Few shot performance on NCBI



(e) Few shot performance on PICO



(f) Few shot performance on CHIA

Figure 3: F1 score plotted against the number of shots in few-shot setting. Performance of all models tends to increase with the number of shots (except for NCBI and MEDM datasets where we observe minor fluctuations in performance).

Table 10 in the appendix provides an overview of the error distribution for every dataset. Several error categories mentioned above could potentially be corrected by providing models access to additional definition knowledge about those entities. This further motivates our exploration of definition-augmented information extraction using LLMs.

Manual Evaluation Prior work has shown that strict F1 can underestimate the performance of generative models on information extraction tasks (Wadhwa et al., 2023). To quantify the impact of this issue on our results, we conduct a small scale human evaluation on two of our datasets (i.e., PICO and CHIA) by randomly sampling 100 sentences with incorrect predictions and re-assessing all the false positive and false negatives. Our analysis showed 51% of PICO and 30% of CHIA predictions deemed incorrect were actually correct.

D Definition Augmentation Error Analysis

We wanted to understand which categories of errors (as per the taxonomy in §C) does definition augmentation help with. For each dataset, we randomly sampled 50 instances with one or more incorrect extractions which were corrected with definition augmentation (in the zero-shot setting). We then looked at the distribution of error types, and found that *extra entities* and *missing entities* were the most common error types fixed using definition information (Table 11).

Model	CDR	CHEM	MedM	NCBI	PICO	CHIA
Missing Entities	75	22.6	47.1	5.5	10.6	39.2
Extra Entities	14.5	21.3	14.2	75	54.54	11.7
Boundary Issues	10.4	22.6	38.5	19.4	12.12	49
Entity Mismatch	0	33.3	-	-	22.7	0

Table 9: Percentage (%) distribution of different types of errors mentioned in C for all datasets in zero-shot setting. Note that NCBI and MEDM datasets have only one entity type, hence there are no type mismatch errors.

Model	CDR	CHEM	MedM	NCBI	PICO	CHIA
Missing Entities	51.2	19.7	24.3	17	32.7	46
Extra Entities	12.1	25.35	18.9	70.2	21.8	9.5
Boundary Issues	34.1	28.1	56.7	12.7	12.7	44.4
Entity Mismatch	2.4	26.7	-	-	32.7	0

Table 10: Percentage (%) distribution of different types of errors mentioned in C for all datasets in few-shot setting. Note that NCBI and MEDM datasets have only one entity type, hence there are no type mismatch errors.

Setting	CDR	NCBI	MEDM	CHIA
Type Mismatch	7.5	-	-	28.9
Boundary Issue	9.4	5.8	0	24
Extra Entities	71.6	82.3	16.4	42
Missing Entities	11.3	11.7	83.5	4.8

Table 11: Percentage (%) distribution of different types of errors mentioned in C for 4 datasets. Note that NCBI and MEDM datasets have only one entity type, hence there are no type mismatch errors.

Model	Engine	Cutoff
GPT 3.5	`gpt-3.5-turbo-0613`	Sep 2021
GPT 4	`gpt4-0613`	Sep 2021
Claude 2	`claude-2`	Dec 2022
LLaMa 2	`llama-2-70b-chat`	Jul 2023

Table 12: Overview of all models.

Dataset	Descriptions	Examples
CHEM	The BioCreative VI Chemical-Protein Interaction corpus (Krallinger et al., 2017) contains biomedical abstracts with annotations for chemical and protein entities.	**Sentence** : AMPK activity was measusalmon as the amount of radiolabelled phosphate transfersalmon to the SAMS peptide. **Entities** : 'Chemicals': ['phosphate'], 'Proteins': ['AMPK']
CDR	The BioCreative V Chemical-Disease Relation corpus (Li et al., 2016) contains biomedical abstracts with annotations for *diseases* and *chemical entities*.	**Sentence** : Pre-treatment of bupivacaine-induced cardiovascular depression using different lipid formulations of propofol. **Entities** : Chemicals : ['bupivacaine', 'propofol'], "Diseases": ['cardiovascular depression']
NCBI	The Natural Center for Biotechnology Information Disease corpus (Doğan et al., 2014) contains biomedical abstracts annotated with *disease mentions*	**Sentence**: Twins with AS were identified from the Royal National Hospital for Rheumatic Diseases database. **Entities**: ['AS', 'Rheumatic Diseases']
MEDM	(Mohan and Li, 2019)corpus consists of biomedical abstracts with annotations for *biomedical concepts* that can be found in knowledge bases.	**Sentence**: A premature electrical impulse from one of four grid corners was utilized to initiate activation. **Entities** : ['premature', 'electrical impulse', 'initiate', 'activation']
PICO	The EBM-NLP corpus (Nye et al., 2018) contains clinical trial abstracts annotated with *(P)articipants*, *(I)nterventions*, and *(O)utcomes*.	**Sentence**: Evaluation of lidocaine in human inferior alveolar nerve block. **Entities** : 'population': ['human inferior alveolar nerve block'], 'intervention': ['lidocaine'], 'outcome': []
CHIA	This dataset contains text snippets from clinical trial eligibility criteria annotated with entities that can be used to form executable logic statements/queries representing the criteria.(Kury et al., 2020)	**Sentence**: Use of medications that alter the absorption or metabolism of levothyroxine. **Entities** : 'Drug' : ['medications', 'levothyroxine'], 'Negation' : ['alter'], 'Observation' : ['absorption of levothyroxine', 'metabolism of levothyroxine'], 'Scope' : ['absorption or metabolism of levothyroxine']

Table 13: Overview of all datasets included in our final biomedical NER evaluation testbed.

TUI id	Name of the entity
T017	Anatomical Structure
T018	Embryonic Structure
T019	Congenital Abnormality
T020	Acquisalmon Abnormality
T021	Fully Formed Anatomical Structure
T024	Tissue
T025	Cell
T026	Cell Component
T028	Gene or Genome
T032	Organism Attribute
T034	Laboratory or Test Result
T037	Injury or Poisoning
T038	Biologic Function
T039	Physiologic Function
T040	Organism Function
T041	Mental Process
T045	Genetic Function
T046	Pathologic Function
T047	Disease or Syndrome
T048	Mental or Behavioral Dysfunction
T059	Laboratory Procedure
T060	Diagnostic Procedure
T061	Therapeutic or Preventive Procedure
T064	Governmental or Regulatory Activity
T082	Spatial Concept

Table 14: The final set of categories used for all definition augmentation experiments (Part 1)

TUI id	Name of the entity
T082	Spatial Concept
T063	Molecular Biology Research Technique
T083	Geographic Area
T085	Molecular Sequence
T086	Nucleotide Sequence
T087	Amino Acid Sequence
T088	Carbohydrate Sequence
T089	Regulation or Law
T095	Self-help or Relief Organization
T097	Professional or Occupational Group
T101	Patient or Disabled Group
T121	Pharmacologic Substance
T122	Biomedical or Dental Material
T123	Biologically Active Substance
T125	Hormone
T126	Enzyme
T127	Vitamin
T129	Immunologic Factor
T131	Hazardous or Poisonous Substance
T169	Functional Concept
T170	Intellectual Product
T191	Neoplastic Process
T192	Receptor
T203	Drug Delivery Device
T204	Eukaryote

Table 15: The final set of categories used for all definition augmentation experiments (Part 2)

E Implementation Details

We used OpenAI API [1], Anthropic API [2] and Together API [3] to run inference. We use the following settings for all closed source models. Temperature is 0 and max number of tokens for extractions being 256. For generating definitions with GPT-4, we increase the max number of tokens to 4096. We use the spaCy (en_core_web_sm) library (Honnibal and Montani, 2017) for tagging biomedical entities.

We fine-tune `Flan-T5-XL` from HuggingFace (Wolf et al., 2020) library on NVIDIA RTX A6000 GPU. We fine-tune with a learning rate of 1e-3 for 10 epochs. We adapt Low-Rank Adaptation of LLM (LoRA) (Hu et al., 2021) with the following parameters : lora_alpha: 32, lora_dropout: 0.05 and SEQ_2_SEQ_LM as the task type.

Output formatting: For datasets with a single entity type (i.e., MEDM and NCBI), we format the outputs as `entity_name <sep> entity_name`; for datasets with multiple types (i.e., CHEM, CDR, PICO and CHIA) we use the format: `[entity_name:entity_type, ..., entity_name:entity_type]`.

[1] https://platform.openai.com/
[2] https://console.anthropic.com/
[3] https://api.together.xyz/

Model	Setting	CHEM	CDR	MedM	NCBI	PICO	CHIA
GPT-3.5	**ZS**	48.61	67.65	43.77	54.05	10.25	7.50
	SC	47.18	68.01	45.6	52.29	8.16	8.53
Claude 2	**ZS**	54.28	70.07	36.98	44.17	7.26	20.12
	SC	55.43	68.75	35.55	37.28	6.9	20.17
Llama 2	**ZS**	60.30	64.07	25.98	47.38	7.88	4.24
	SC	57.63	64.07	26.08	44.81	6.7	5.87
GPT-4	**ZS**	62.12	70.92	47.13	54.67	7.29	16.39
	SC	63.85	71.02	46.86	56.75	7.41	16.96

Table 16: F1 scores of zero-shot (ZS) followed by self-consistency (SC) for all models and datasets. We don't see gain in the performance when prompted without augmenting with the definitions.

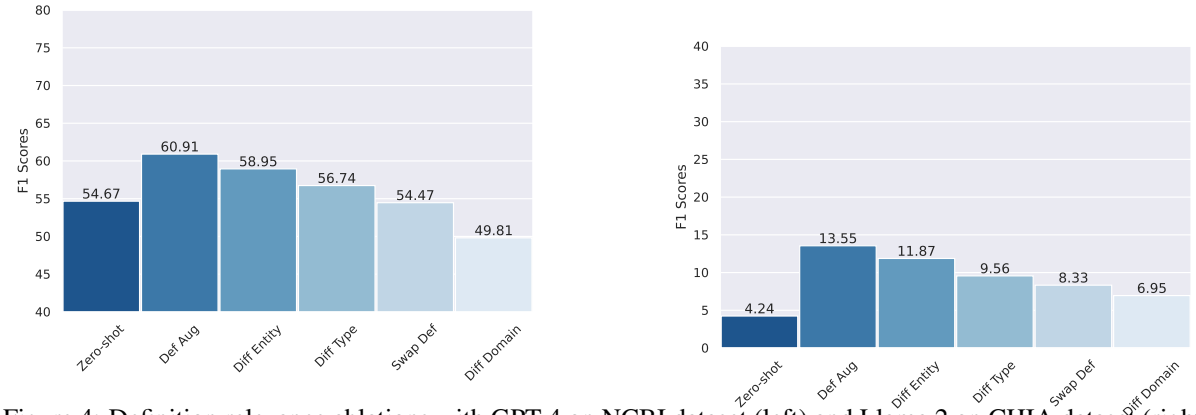


Figure 4: Definition relevance ablations with GPT-4 on NCBI dataset (left) and Llama 2 on CHIA dataset (right). We see similar trends across all models and datasets - a consistent decrease in performance with less relevant definitions.

Given the following sentence from an abstract, extract all the chemicals and diseases and return as a json.

Sentence: In the present case, complete atrioventricular (AV) block with syncopal attacks developed secondary to lithium therapy, necessitating permanent pacemaker implantation.

Output: {"Chemical": ["lithium"], "Disease": ["atrioventricular (AV) block", "syncopal attacks"]}

Figure 5: Zero-shot Prompt with *text input* and *JSON output*

Given the following sentence from an abstract and the definitions of chemicals and diseases, extract all the chemicals and diseases.

Chemicals: A chemical substance is a form of matter having constant chemical composition and characteristic properties. Chemical substances cannot be separated into their constituent elements by physical separation methods, i.e., without breaking chemical bonds. Chemical substances can be simple substances (substances consisting of a single chemical element) chemical compounds, or alloys.

Diseases: A disease is a particular abnormal condition that negatively affects the structure or function of all or part of an organism, and that is not immediately due to any external injury. Diseases are often known to be medical conditions that are associated with specific signs and symptoms. A disease may be caused by external factors such as pathogens or by internal dysfunctions.

Sentence: In the present case, complete atrioventricular (AV) block with syncopal attacks developed secondary to lithium therapy, necessitating permanent pacemaker implantation.

Output: {"Chemical": ["lithium"], "Disease": ["atrioventricular (AV) block", "syncopal attacks"]}

Figure 6: Zero-shot Prompt with *schema def input* and *JSON output*

```
def named_entity_recognition(input_text):
      """
      Given a sentence from an abstract,extract all the chemicals and diseases.
      A chemical entity is a dictionary of the format {"text": "extracted
entities", "type": "chemicals"}
      A disease entity is a dictionary of the format {"text": "extracted entities", "type":
"diseases"}
      Find all the entities in input_text and append only the entities and not other
information to entity_list one by one. """

      input_text = "Pre-treatment of bupivacaine-induced cardiovascular depression using
different lipid formulations of propofol."
      entity_list = []
      # extracted entities
      entity_list.append({\''
```

Output:
```
"text': 'bupivacaine', 'type': 'chemicals'})
        entity_list.append({'text': 'propofol', 'type': 'chemicals'})"
```

Figure 7: Zero-shot Prompt with *text input* and *code output*

```
def named_entity_recognition(input_text):
      """
      Chemicals: A chemical substance is a form of matter having constant chemical
composition and ...single chemical element) chemical compounds, or alloys.
          Diseases: A disease is a particular abnormal condition that negatively affects the
structure or function of all ...A disease may be caused by external factors such as pathogens
or by internal dysfunctions.

      Given a sentence from an abstract, and the definitions of chemicals and diseases,
extract all the chemicals and diseases.
      A chemical entity is a dictionary of the format {"text": "extracted entities", "type":
"chemicals"}
      A disease entity is a dictionary of the format {"text": "extracted entities", "type":
"diseases"}
      Find all the entities in input_text and append only the entities and not other
information to entity_list one by one. """

      input_text = "Pre-treatment of bupivacaine-induced cardiovascular depression using
different lipid formulations of propofol."
      entity_list = []
      # extracted entities
      entity_list.append({\''
```

Output:
```
"text': 'bupivacaine', 'type': 'chemicals'})
            entity_list.append({'text': 'propofol', 'type': 'chemicals'})"
```

Figure 8: Zero-shot Prompt with *schema def input* and *code output*

Given the sentence from an abstract, extract all the chemicals and diseases and return as a json.

Sentence: BE-Induced seizures occurred more frequently and had significantly longer latencies than those induced by equimolar amounts of cocaine.'},

Output :{"chemicals": ["BE", "cocaine"],"diseases": ["seizures"]}

Sentence: Famotidine-associated delirium.

Output: {"chemicals": ["Famotidine"],"diseases": ["delirium"]}

Figure 9: Few-shot Prompt with *text* and *JSON output*

```
def named_entity_recognition(input_text):
      """ Given the above definitions of entities, Extract a list of all population,
intervention, comparator and outcome spans from input_text.
      A population entity is a dictionary of the format {"text": entity, "type":
"population"}
      ...
      Find all entities in input_text and append them to entity_list one by one. If there are
no entities, return an empty list.
      """

      input_text = One subject was identified to be a poor metabolizer.
      entity_list = []
      # extracted entities
      Output:entity_list.append({text": "poor metabolizer ", "type": "outcome"}

      input_text = "OBJECTIVE To compare the effect of budesonide Turbuhaler 400 microg/day
with budesonide aqua 256 microg/day in the treatment of seasonal allergic rhinitis ( SAR ) ."
      entity_list = []
      # extracted entities
```

Output: 'entity_list = [{"text": "budesonide Turbuhaler 400 microg/day",
"type":"intervention"}
entity_list.append{"text": "budesonide aqua 256 microg/day", "type": "intervention"}
entity_list.append{"text": "seasonal allergic rhinitis", "type": "population"}\n]

Figure 10: Few-shot Prompt with *text* and *code output*

Given the sentence from an abstract, extract all the chemicals and diseases and return as a JSON.

Sentence: There are several common polymorphisms in the BRCA1 gene which generate amino acid substitutions.

Output: {'diseases': ['BRCA1']}

To assist you with extraction, here are the definitions of the extracted entities:
BRCA1 gene: A tumor suppressor gene (GENES, TUMOR SUPPRESSOR) located on human CHROMOSOME 17 at locus 17q21. Mutations of this gene are associated with the formation of HEREDITARY BREAST AND OVARIAN CANCER SYNDROME. It encodes a large nuclear protein that is a component of DNA repair pathways.

Using these definitions only as a reference, add or remove incorrect entities from the output json only if you think the entities in the output json are wrong else don't change the output. Please only output the final json.

Sentence: There are several common polymorphisms in the BRCA1 gene which generate amino acid substitutions.

Output: Output: {'diseases': []}

Figure 11: Zero-shot Definition Augmentation with Single Turn

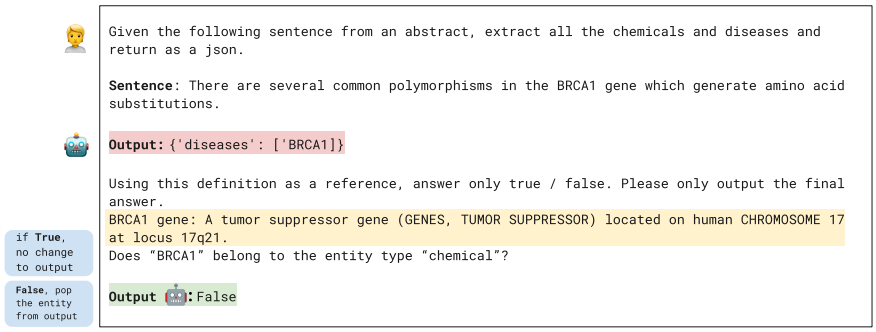


Figure 12: Zero-shot Definition Augmentation with Iterative Prompting with extracted entities

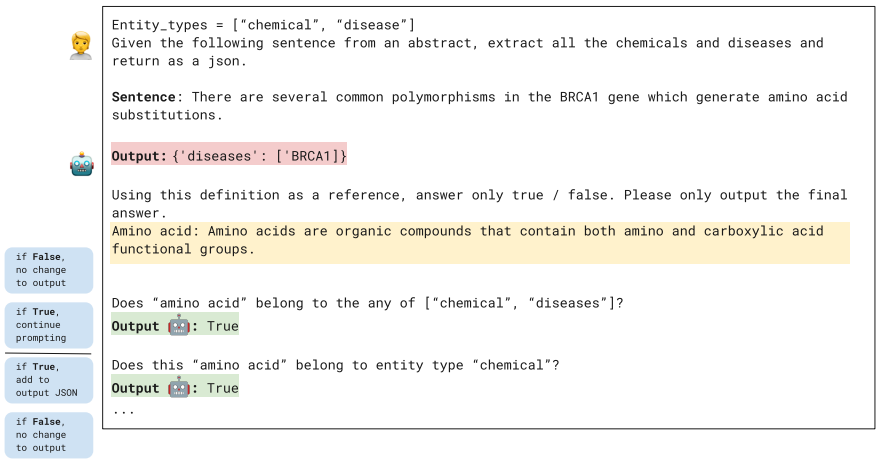


Figure 13: Zero-shot Definition Augmentation with Iterative Prompting with biomedical phrases

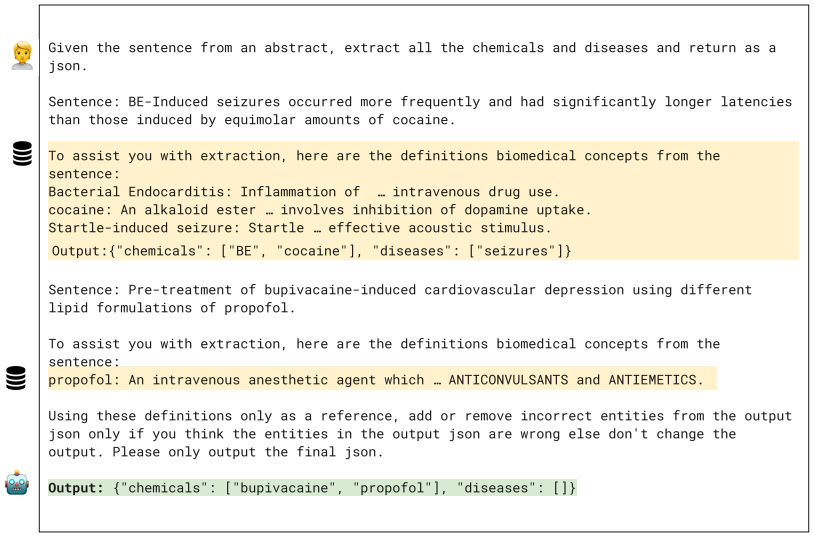


Figure 14: Few-shot Definition Augmentation with Single Turn