# Task as Context Prompting for Accurate Medical Symptom Coding Using Large Language Models

Chengyang He
Stevens Institute of Technology
Hoboken, New Jersey, USA
che14@stevens.edu

Wenlong Zhang
Stevens Institute of Technology
Hoboken, New Jersey, USA
wzhang71@stevens.edu

Violet (Xinying) Chen
Stevens Institute of Technology
Hoboken, New Jersey, USA
vchen3@stevens.edu

Yue Ning
Stevens Institute of Technology
Hoboken, New Jersey, USA
yning5@stevens.edu

Ping Wang
Stevens Institute of Technology
Hoboken, New Jersey, USA
pwang44@stevens.edu

## ABSTRACT

Accurate medical symptom coding from unstructured clinical text, such as vaccine safety reports, is a critical task with applications in pharmacovigilance and safety monitoring. Symptom coding, as tailored in this study, involves identifying and linking nuanced symptom mentions to standardized vocabularies like MedDRA, differentiating it from broader medical coding tasks. Traditional approaches to this task, which treat symptom extraction and linking as independent workflows, often fail to handle the variability and complexity of clinical narratives, especially for rare cases. Recent advancements in Large Language Models (LLMs) offer new opportunities but face challenges in achieving consistent performance. To address these issues, we propose Task as Context (TACO) Prompting, a novel framework that unifies extraction and linking tasks by embedding task-specific context into LLM prompts. Our study also introduces SYMPCODER, a human-annotated dataset derived from Vaccine Adverse Event Reporting System (VAERS) reports, and a two-stage evaluation framework to comprehensively assess both symptom linking and mention fidelity. Our comprehensive evaluation of multiple LLMs, including Llama2-chat, Jackalope-7b, GPT-3.5 Turbo, GPT-4 Turbo, and GPT-4o, demonstrates TACO's effectiveness in improving flexibility and accuracy for tailored tasks like symptom coding, paving the way for more specific coding tasks and advancing clinical text processing methodologies.

## KEYWORDS

Medical coding, large language models, task as context, chain-of-thought prompting

## 1 INTRODUCTION

Accurate symptom coding from unstructured clinical text, particularly in the context of vaccine safety and pharmacovigilance, remains a critical yet complex task. Systems such as the Vaccine Adverse Event Reporting System (VAERS) [8] play a vital role in monitoring potential adverse events following immunization on a global scale. However, the highly variable and informal nature of clinical narratives within these reports presents significant challenges for automatically extracting and linking symptoms to standardized medical vocabularies like Medical Dictionary for Regulatory Activities (MedDRA) [7].

In this work, we formulate symptom coding as a task of identifying and linking nuanced symptom mentions to standardized vocabularies, which is different from broader medical coding tasks such as International Classification of Diseases (ICD) coding. While ICD coding primarily addresses diagnoses and procedures, symptom coding emphasizes the extraction of subjective experiences (e.g., "dizziness" or "rash") and their precise mapping to a predefined set of codes. This tailored task is crucial for downstream applications such as pharmacovigilance, safety monitoring, and trend analysis, offering a more flexible and specific approach compared to traditional medical coding practices.

Traditional methods for medical symptom coding typically separate symptom extraction and linking into independent workflows. Earlier approaches relied on rule-based systems or contextual models, which often struggled to capture the full context of clinical narratives and were prone to errors, especially for rare or ambiguous symptoms [15, 19, 27]. The disjointed nature of these processes introduced inefficiencies and inconsistencies, limiting their effectiveness in complex cases. These limitations are particularly pronounced in tasks like symptom coding, where ensuring reliable mappings of symptom mentions to standardized vocabularies is critical.

Recent advancements in Large Language Model (LLMs) have shown promise in addressing these challenges, offering enhanced contextual understanding and nuanced language processing capabilities [28]. By integrating extraction and linking tasks into a single process, LLMs have the potential to overcome the limitations of traditional approaches. However, ensuring consistent accuracy across
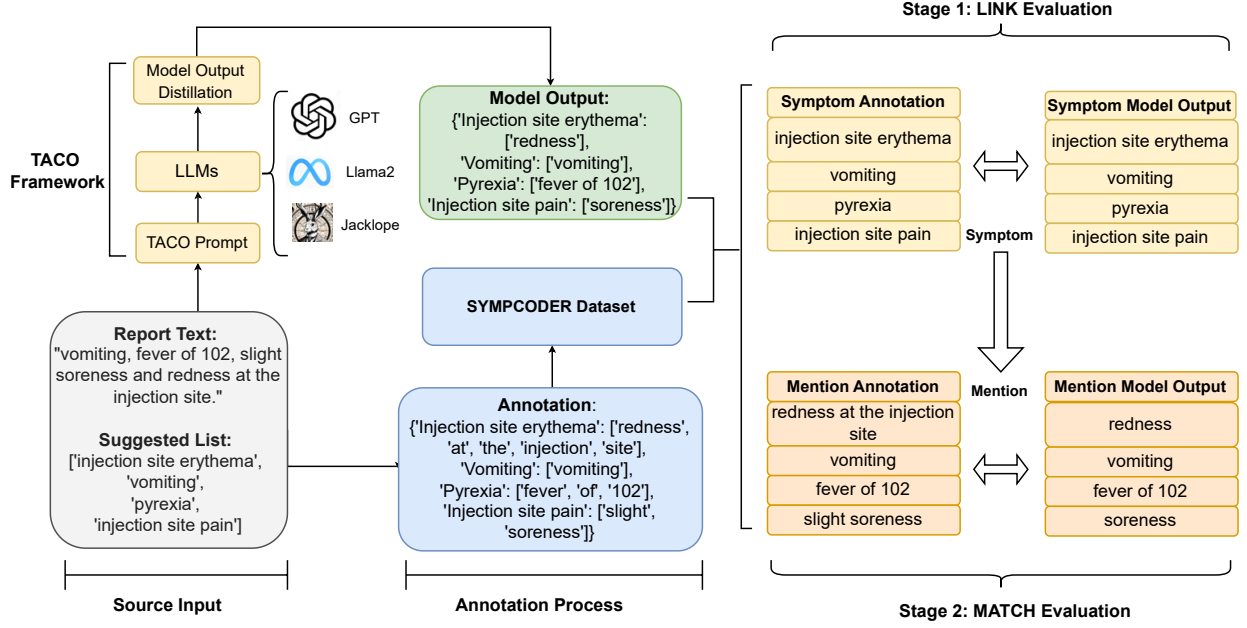
**Figure 1: SYMPCODER data creation and overall workflow of TACO prompting and evaluation. Key components in the framework include: (1) Source Input: Report text and a suggested symptom list; (2) TACO Prompting: Guides LLMs in symptom coding; (3) Model Output Distillation: Refines LLM outputs; (4) SYMPCODER Dataset: Contains human annotations; and (5) Two-Stage Evaluation: LINK for matching extracted symptoms with annotations, and MATCH for assessing the contextual accuracy of symptom mentions.**

diverse cases—especially rare or complex adverse events—remains an open problem [15]. The need for a unified framework that preserves the relationships between symptoms and their corresponding codes while maintaining efficiency is paramount.

To address these challenges, we propose the **Task as Context Prompting (TACO)**, a novel approach leveraging LLMs to perform symptom extraction and linking as interdependent tasks. Unlike traditional methods that treat these processes in isolation [19], TACO embeds task-specific context directly within LLM prompts, enabling the model to understand and maintain relationships between symptoms and standardized codes throughout the process. By unifying extraction and linking, TACO reduces information loss and enhances the flexibility and accuracy of symptom coding. Additionally, to the best of our knowledge, there is no existing dataset specifically designed for symptom coding in the medical domain. To support the training and evaluation of models for medical symptom coding, we introduce **SYMPCODER**, a human-annotated dataset derived from VAERS reports. This dataset encompasses three variants, SYMPCODER-Full, SYMPCODER-Common-50, and SYMPCODER-Rare-50, providing a comprehensive benchmark for assessing model performance across diverse cases, including both frequently reported and rare adverse events. By offering detailed annotations for both symptom extraction and linking, SYMPCODER enables systematic evaluation of models and facilitates further advancements in medical symptom coding research.

Our study further introduces a two-stage evaluation framework. The **Linking Integrity and Knowledge (LINK)** stage evaluates the accuracy of linking extracted symptoms to standardized codes,

while the **Mention Accuracy and Textual Coherence (MATCH)** stage focuses on the fidelity and coherence of the original mentions extracted from clinical narratives. This dual-phase evaluation framework provides a granular analysis of model performance across both common and rare cases, offering comprehensive insights into their capabilities. Figure 1 illustrates our TACO prompting framework, detailing the workflow from input processing to final evaluation. In summary, our study introduces the following contributions:

- **SYMPCODER Dataset**: A human-annotated dataset based on VAERS. This dataset includes three variants and provides a benchmark to evaluate the capabilities of various methods for symptom coding, including LLMs, across diverse cases, ranging from common to rare adverse events.
- **TACO Prompting Framework**: A novel approach that unifies symptom extraction and linking by embedding task-specific context within prompts, improving flexibility and accuracy.
- **Comprehensive Two-Stage Evaluation**: A dual-phase framework (LINK and MATCH) that assesses both the linking accuracy of extracted symptoms and the coherence of their original mentions, providing a detailed understanding of model capabilities.

## 2 RELATED WORK

### 2.1 Entity Extraction and Linking

Entity extraction and linking are fundamental tasks in natural language processing (NLP) that facilitate the transformation of unstructured text into structured, actionable data. These tasks are

widely used in domains such as general text processing, biomedical information retrieval, and clinical text mining [17].

Entity extraction focuses on identifying relevant entities (e.g., people, places, symptoms) in text. Earlier approaches to this task relied on rule-based systems and statistical models, such as Hidden Markov Models (HMMs) [3] and Conditional Random Fields (CRFs) [33]. While these models were effective in specific scenarios, they struggled with generalizing across diverse contexts and handling the inherent variability of natural language [21]. The introduction of deep learning, particularly Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, improved the ability to capture sequential dependencies and contextual nuances [1]. Recently, transformer-based models such as BERT [11] have revolutionized entity extraction by leveraging self-attention mechanisms, enabling the capture of long-range dependencies and complex contextual relationships [34].

Entity linking complements extraction by resolving ambiguities and connecting extracted entities to predefined ontologies or knowledge bases such as Wikipedia [12], DBpedia [2], or medical databases like SNOMED [10] and UMLS [4]. Traditional methods typically employed a two-step approach: extracting handcrafted features and using entity-ranking models for linking predictions. These methods were limited by their reliance on labor-intensive feature engineering and poor generalizability across different KBs and domains [37]. Modern approaches such as embedding-based method, powered by models like BERT, have significantly enhanced linking by mapping textual entities and knowledge base entries into a shared vector space [6]. Despite advancements, embedding-based methods still face challenges such as reliance on large annotated datasets, difficulty generalizing to unseen entities or domains, and computational inefficiency at scale, while often struggling with nuanced contextual relationships in specialized domains [37].

In the biomedical domain, extraction and linking tasks are often addressed independently, overlooking their inherent interdependencies and potential benefits of integrating them. However, combining both extraction and linking tasks is critical for applications like ICD coding and adverse event detection, where seamless integration not only enhances the accuracy but also improves the overall efficiency by providing essential contextual information to each other [13, 20]. BioBERT-based Named Entity Recognition and Normalization (BERN) [22] marked a milestone in this field by unifying entity extraction and linking into a single, streamlined framework. Leveraging a fine-tuned version of BioBERT [23], BERN integrates these interdependent tasks and eliminates the need for separate modules, achieving state-of-the-art performance at the time. This integrated approach laid the groundwork for further advancements in medical coding. However, significant gaps and challenges remain, highlighting the need for further investigation.

## 2.2 Advances in Contextual Models and LLMs
Contextual language models like BERT and its domain-specific adaptations, such as BioBERT and ClinicalBERT [16], have significantly improved tasks involving complex biomedical text. By incorporating domain-specific pretraining on large biomedical corpora, these models enhanced the accuracy of tasks like symptom extraction, diagnosis identification, and treatment mapping [18, 28].

Despite these advances, they continued to treat extraction and linking as distinct tasks, requiring additional post-processing steps that often introduced errors, particularly for ambiguous or infrequent cases [19]. Moving beyond these limitations, CNN-based architectures such as MultiResCNN [24] further advanced medical coding by incorporating residual blocks and multi-resolution filters to capture complex patterns in clinical notes. However, these models required explicit alignment of input features with predefined codes, limiting their adaptability to unseen data or emerging clinical terms.

The emergence of large language models (LLMs) like GPT-3 and GPT-4 has transformed NLP, enabling the integration of extraction and linking tasks within a single framework. For example, LLM-Codex [40] utilized a two-stage pipeline to improve the reliability of ICD coding predictions, with an LLM in the first stage and a verifier model in the second stage. Similarly, Boyle et al. [5] proposed a zero-shot and few-shot ICD coding method using pre-trained LLMs, framing the task as an information extraction problem and leveraging the hierarchical structure of the ICD ontology for efficient code assignment. While these approaches demonstrated state-of-the-art performance, their reliance on predefined hierarchies and focus on ICD coding distinguishes them from our work.

## 3 THE SYMPCODER DATASET CREATION

### 3.1 VAERS Dataset
The Vaccine Adverse Event Reporting System (VAERS) dataset, managed by the Centers for Disease Control and Prevention (CDC) and the Food and Drug Administration (FDA), is a critical resource for monitoring vaccine safety. VAERS operates as a passive reporting system where individuals, including healthcare professionals and the general public, can submit reports of adverse events following immunization. Healthcare professionals must report certain adverse events, and vaccine manufacturers must report all adverse events they become aware of. VAERS does not determine causality but helps detect unusual or unexpected patterns in adverse event reports that may suggest safety issues, aiding the CDC and FDA in identifying areas that need more evaluation and assessment. Since 1990, VAERS has collected millions of reports, each containing detailed information such as demographics, vaccination details, descriptions of adverse events, standard symptom lists, and medical history [8]. This extensive dataset serves as a valuable resource for medical symptom coding, facilitating ongoing research and analysis to enhance our understanding of vaccine safety and address emerging concerns.

### 3.2 Entity Mention Annotation
We randomly selected 500 VAERS reports (1990–2023) and assembled a team of three annotators plus one validator to ensure accuracy. Each report included suggested symptom terms (MedDRA-encoded), and our annotation process marked all symptom mentions related to vaccine adverse events. This approach captures the variability of symptom expression and yields a robust dataset for model training and evaluation. The annotation process consists of two key phases, including **annotation** and **validation**, designed to ensure both thoroughness and accuracy.

**Table 1: Basic Data Statistics of SYMPCODER. Note that "Suggested Symptoms" refers to the labeled symptoms in the SYMPCODER dataset (not the "suggested list" from VAERS), and "Extracted Symptoms" are those automatically identified by LLMs.**

|  | Clinical Text | Suggested Symptoms | Extracted Symptoms |
|---|---|---|---|
| **SYMPCODER-Full** (# of Reports: 487) | | | |
| Average Length | 843 | 8 | 6 |
| Median Length | 321 | 5 | 5 |
| Min Length | 9 | 1 | 0 |
| Max Length | 11834 | 105 | 41 |
| **SYMPCODER-Common-50** (# of Reports: 427) | | | |
| Average Length | 873 | 9 | 6 |
| Median Length | 326 | 6 | 5 |
| Min Length | 9 | 1 | 0 |
| Max Length | 11834 | 105 | 41 |
| **SYMPCODER-Rare-50** (# of Reports: 22) | | | |
| Average Length | 1659 | 9 | 8 |
| Median Length | 1000 | 8 | 7 |
| Min Length | 30 | 2 | 0 |
| Max Length | 8555 | 30 | 21 |

*3.2.1 Annotation Phase.* The annotation phase was meticulously carried out by three graduate students selected for their academic background, annotation skills, English proficiency, and basic clinical knowledge, ensuring they could handle the linguistic and domain-specific nuances required for accurate annotation. Each annotator was responsible for annotating approximately one-third of the total 500 reports for annotations by following a clearly defined set of guidelines to ensure consistency and reliability across all reports. During the annotation, (1) the annotators used a home-maintained data annotation tool designed specifically for this task. This tool ensured a seamless workflow, focusing solely on adverse events while disregarding procedural details and negative test results to eliminate any irrelevant information. (2) Each report was annotated one label at a time, using a list of suggested terms for symptoms provided in the original VAERS data, which are encoded according to the MedDRA terminology. This step-by-step process allowed the annotators to focus on one potential adverse event at a time, ensuring accuracy and precision. (3) In cases where annotators encountered uncertainties or ambiguities in the text, they marked the annotation as "uncertain" for further validation. This precautionary measure allowed a second layer of scrutiny to improve the reliability of the dataset.

*3.2.2 Validation Phase.* The validation phase was conducted by a senior annotator specializing in natural language processing for the medical domain, selected for his or her extensive annotation experience. (1) The validator meticulously reviewed all annotations, especially those marked as "uncertain", to ensure flagged annotations received additional scrutiny and maintained high data integrity. (2) For each uncertain annotation, the validator engaged in discussions with the original annotator to understand their reasoning and reach a consensus. Persistent ambiguities were resolved collaboratively within the team to ensure alignment with the annotation guidelines. (3) When discrepancies or inconsistencies were identified, the validator suggested revisions, which were reviewed and discussed with the annotators. Final decisions on adjustments

were made collaboratively to ensure consistency and transparency throughout the validation process.

The resulting human-annotated dataset, **SYMPCODER**, stands as a fundamental benchmark for medical symptom coding tasks. Beyond its utility in our study, this dataset offers valuable insights for advancing research in this field and exploring the capabilities of various methods for symptom coding in future investigations. This rigorous annotation and validation process ensured high data integrity and reliability, making SYMPCODER a robust resource for evaluating the performance of LLMs in the symptom coding task presented in this study.

*3.2.3 SYMPCODER Dataset.* We constructed three subsets of the annotated dataset: SYMPCODER-Full, SYMPCODER-Common-50, and SYMPCODER-Rare-50 to facilitate focused analyses. These subsets were created to evaluate model performance across symptoms with different frequencies, targeting variations between common cases and rare or ambiguous cases. Additional details about the dataset and project can be found in https://github.com/LEAF-Lab-Stevens/TACO-Prompting.

- **SYMPCODER-Full**: This comprehensive dataset includes all annotated symptom mentions from the VAERS reports, providing a holistic benchmark for symptom extraction and linking tasks.
- **SYMPCODER-Common-50**: This subset focuses on the 50 most frequently mentioned symptoms in the dataset. Symptoms were ranked by their frequency of occurrence within SYMPCODER-Full. The top 50 symptoms were selected, and reports containing at least one of these symptoms were included in this subset.
- **SYMPCODER-Rare-50**: This subset highlights the 50 least frequently mentioned symptoms. Following a similar process to the Common-50 subset, symptoms were ranked by frequency, and reports containing at least one of these rare symptoms were included in this subset.

## 3.3 Basic Statistics of SYMPCODER

Of the initial 500 annotations, 487 distinct reports remained after removing 13 uncertain results, which annotators and validator could not verify as symptoms due to limited domain knowledge or complex contexts. The SYMPCODER datasets, comprising SYMPCODER-Full, SYMPCODER-Common-50, and SYMPCODER-Rare-50, provide a robust foundation for further studies. As shown in Table 1, the SYMPCODER-Full dataset offers a comprehensive resource, capturing diverse adverse event descriptions with varying lengths and complexities. Table 1 provides additional details, such as the average and median lengths of clinical text, suggested symptoms, and extracted symptoms. These metrics reveal consistent gaps between human-labeled and model-generated symptoms across all subsets, highlighting the challenges of achieving complete extraction. By focusing on common and rare subsets, the SYMPCODER-Common-50 and SYMPCODER-Rare-50 datasets provide targeted benchmarks for evaluating model performance across both frequent and infrequent adverse events.

Figure 2a shows that most reports in the SYMPCODER-Full dataset contain between 0 and 10 symptoms, with a sharp decline for higher counts. A similar trend is observed in Figure 2b for the SYMPCODER-Common-50 dataset. Figure 2c highlights a steeper drop after 10 symptoms in the SYMPCODER-Rare-50 dataset, with
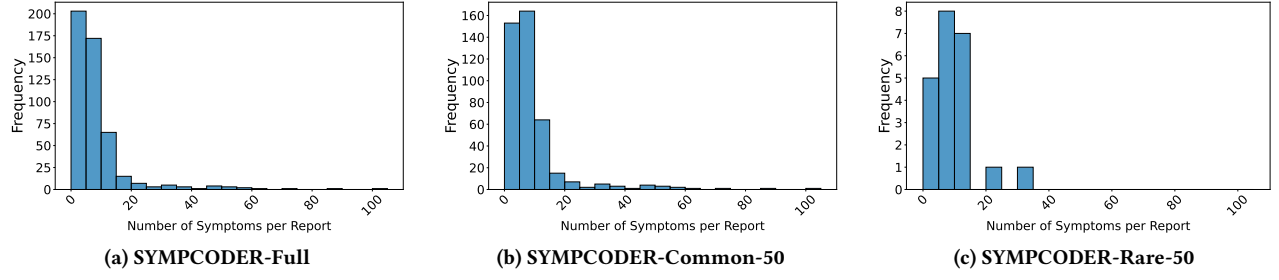
Figure 2: Distributions of the number of symptoms for different datasets in SYMPCODER.

| Prompting Method Structure | | |
|---|---|---|
| | **TASI** | **TACO** |
| Clinical Text | {'Headaches for 2 weeks tired for 2 weeks sinus congestion for 2 weeks left side of throat sore month.',} | |
| Prompt Header | First, extract a symptom list including all symptoms from the clinical text above<br>Then, extract the symptoms that indicating each of the suggested terms below from the symptom list in previous step. | Extract the terms mentioned in the clinical text above that indicating each of the following terms |
| Prompt Body | Include the terms in the output even if the terms are not explicitly mentioned in the provided report, just provide 'none' as the result.<br><br>Please follow the order of this list: {suggested symptom list} and generate output by following the requirements below:<br>Requirements:<br>1. (Define adverse event symptom)<br>2. (Return "none" for non-symptom terms)<br>3. (Use JSON as shown in the example) | |
| Output Instruction | {{"Symptom1": ["redness", "redness"], "Symptom2": ["fever"], "Symptom3": ["sore", "arm", "soreness"], "Symptom4": ["heartfailure"]}}<br>{"Erythema": ["redness", "redness"], "Pain in extremity": ["sore", "arm", "soreness"], "Pruritus": ["none"]}} | {{"Erythema": ["redness", "redness"], "Pain in extremity": ["sore", "arm", "soreness"], "Pruritus": ["none"]}} |

Figure 3: The structure of TASI and TACO prompts, detailing clinical input, task instructions, and output format. The provided output examples are for format demonstration purposes only and do not align with the clinical input text, which is taken from real clinical reports.

a comparative analysis of the two approaches. Throught this comparison, our study highlights how embedding interrelated tasks, as proposed by the Task as Context strategy, enhances the performance and adaptability of LLMs for automated medical symptom coding.

## 4.1 Task as Context Prompting

*4.1.1* **Inspiration**. The Task as Context concept, originally designed for annotation workflows with non-experts involving interdependent tasks, emphasizes leveraging contextual relationships to improve accuracy and adaptability. This strategy integrates related tasks into a single process, reducing inefficiencies and error propagation [25]. In the context of symptom coding, two highly interdependent tasks, symptom extraction and linking symptoms to standardized codes, are traditionally solved independently. However, this modular approach suffers from limitations, including the loss of contextual cues between tasks, error propagation from one task to another, and scalability challenges due to the need for separate training processes.

Inspired by the Task as Context strategy, we address these limitations by designing TACO prompting, which explicitly embeds the interdependence of extraction and linking into a unified prompting framework. TACO integrates the tasks within the prompt, enabling simultaneous learning and execution. By treating the two tasks holistically, our approach improves contextual understanding, minimizes error propagation, and enhances both accuracy and scalability, providing a streamlined and adaptive solution for symptom coding challenges.

*4.1.2* **TACO Prompting Design**. To clearly present the structure of TACO prompting, we outline its four core components in Figure 3.

- **Clinical Text:** The clinical text serves as the raw, unstructured input data from the VAERS reports. It mirrors real-world scenarios where models must process free-text narratives to extract and link relevant symptoms. This component provides the context required for understanding the input data and serves as the foundation for subsequent tasks.
- **Prompt Header:** The prompt header provides explicit instructions for task execution. In TACO, this component integrates both symptom extraction and linking tasks into a single step. It instructs the model to directly extract symptoms from the clinical text and map them to the provided suggested terms, leveraging the broader task context. In contrast, the benchmark prompt TASI separates these tasks into two distinct phases, requiring the

a few outliers reaching up to 40 symptoms, reflecting the rarity and complexity of such cases.

## 4 THE PROPOSED TACO PROMPTING

This paper evaluates the efficacy of various LLMs in extracting vaccine adverse event symptoms from VAERS reports and linking to formal MedDRA codes in a predefined list. To address this task, we designed **T**ask **a**s **Co**ntext (TACO) prompting, a novel prompting method that integrates symptom extraction and linking into a unified framework. By embedding the interrelated tasks within a single prompt, TACO leverages the broader task context to enhance model performance and adaptability for automated symptom coding. To benchmark its effectiveness, we also designed **T**ask **a**s **S**equential (TASI), which separates both tasks into distinct phases, allowing for

model to first extract all symptoms and subsequently link them to suggested terms.

- **Prompt Body:** The prompt body contains detailed guidelines for task execution and ensures that both tasks are carried out as intended. For TACO, the instructions guide the model to simultaneously address symptom extraction and linking, thereby reducing redundancy and improving efficiency. TASI, however, employs sequential instructions, where symptom extraction is followed by linking. This distinction reflects the core advantage of TACO, which embeds the broader task context into a cohesive framework to improve task understanding and execution.
- **Output Instruction:** The output instruction specifies the expected output format to ensure consistency and clarity. Both TACO and TASI utilize a structured JSON format; however, their approaches differ. In TACO, the format unifies the output by directly mapping symptoms to suggested terms, aligning with its integrated task-solving approach. In TASI, the output separates extraction and linking results, reflecting its sequential methodology. The inclusion of illustrative examples further clarifies the expected output structure for the model.

To comprehensively evaluate the effectiveness of TACO, we compare it against the benchmark prompt TASI, which adheres to traditional sequential task-solving methodologies. While TASI serves as a baseline, TACO's integrated and context-rich design seeks to overcome the limitations of traditional approaches by streamlining task execution and enhancing contextual understanding.

*4.1.3 **Model Output Distillation and Evaluation Tasks**.* To address inconsistencies in model outputs, we incorporate a post-processing pipeline during the model output distillation phase, leveraging Regular Expression (regex) [38] syntax to systematically capture the information needed. These inconsistencies include incomplete model outputs, unnecessary descriptive text at the beginning of responses, and nonsensical outputs that deviate from task requirements. The distillation process ensures uniformity and precise extraction of relevant information from the varied responses generated by LLMs. As shown in Figure 1, the TACO outputs undergo this refinement step, eliminating irrelevant details and preparing them for structured evaluation.

The evaluation phase comprises two stages. In the first stage, known as **L**inking **In**tegrity and **K**nowledge (LINK), we assess the models based on how accurately they link extracted symptoms to the corresponding terms from the Suggested List. In this stage, as Figure 1 shows, the model must first identify the correct symptoms (e.g., "injection site erythema," "vomiting") from the clinical text using the distilled output from the previous phase. The focus here is on ensuring that each relevant symptom has been accurately identified and linked to the correct terminology. This stage evaluates the model's ability to extract and link adverse events, providing insights into how well it handles clinical information within the context of symptom extraction.

In the second stage, termed **M**ention **A**ccuracy and **T**extual **Co**herence (MATCH), the focus shifts to the quality of the original mentions generated by the model for each linked symptom. As depicted in Figure 1, this involves comparing the generated model results to the human-annotated gold standard in terms of specificity and precision. For example, while LINK verifies if the model

has correctly linked "injection site erythema," MATCH evaluates whether the model-generated mention—such as "redness at the injection site"—is semantically similar to the original clinical report, which is annotated simply as "redness." This two-tiered evaluation provides a deeper analysis of how well the models not only identify but retain the original context and details from the clinical text.

## 4.2 Benchmark LLMs

We aim to harness the capabilities of several cutting-edge Large Language Models (LLMs) for the extraction and linking of adverse events on the SYMPCODER dataset. Each model offers unique strengths and characteristics, making them suitable for various aspects of this complex task.

- **Jackalope-7b** [26]: The smallest open-sourced model in our investigation, Jackalope-7b is fine-tuned by SlimOrca using several open datasets on top of Mistral 7B. Despite its smaller size, Jackalope-7b is optimized for specific NLP tasks and demonstrates a balanced trade-off between performance and computational efficiency. Its architecture allows for quicker adjustments and fine-tuning, making it particularly effective in scenarios requiring high precision within a smaller model footprint.
- **Llama2-13b-chat** [39]: Developed by Meta AI, Llama2-13b-chat is a medium-sized model based on the widely recognized Llama2. Known for its advanced natural language understanding capabilities, Llama2-13b-chat is designed to handle complex language tasks with a focus on generating human-like responses. Its medium size strikes a balance between computational demand and performance, making it a versatile choice for a variety of NLP applications.
- **GPT-3.5-Turbo** [29]: An efficient variant of OpenAI's GPT-3, GPT-3.5-Turbo is designed to excel in natural language processing tasks with optimized performance. It leverages the strengths of GPT-3 while incorporating enhancements that improve response coherence, accuracy, and efficiency. This model is well-suited for applications requiring robust language comprehension and generation capabilities.
- **GPT-4-Turbo** [30]: An enhanced version of GPT-3.5-Turbo, GPT-4-Turbo offers comprehensive improvements in terms of model architecture, training data diversity, and computational efficiency. It is designed to handle more complex language tasks with greater accuracy and faster response times. GPT-4-Turbo's advancements make it a powerful tool for sophisticated NLP applications, including detailed extraction and linking tasks.
- **GPT-4o** [31]: A faster and more cost-effective variant of GPT-4-Turbo, GPT-4o provides efficient performance at a reduced cost. It retains many of the advanced features of GPT-4-Turbo while optimizing for speed and resource utilization. GPT-4o is particularly advantageous in environments where computational resources are limited, but high performance is still required. Its ability to balance cost and efficiency makes it a practical choice for extensive NLP tasks.

## 5 EXPERIMENTS

Based on the SYMPCODER dataset, three research questions (RQs) are studied in this paper, including

**Table 2: LINK stage results assessing the accuracy of linking extracted symptoms to standard medical codes. Bold values indicate the best performance for each prompting method, while italic values show the overall best performance across methods.**

| Prompt Type | Models | EM-Precision | EM-Recall | Fuzzy-Precision | Fuzzy-Recall | EM-Fuzzy-Precision | EM-Fuzzy-Recall |
|---|---|---|---|---|---|---|---|
| TASI | Jackalope-7b | 0.874 | 0.841 | 0.876 | 0.843 | 0.877 | 0.844 |
|  | Llama-2-13b-chat | 0.765 | 0.582 | 0.769 | 0.584 | 0.772 | 0.585 |
|  | gpt-3.5-turbo | 0.838 | 0.814 | 0.846 | 0.821 | 0.853 | 0.827 |
|  | gpt-4-turbo | 0.887 | 0.867 | 0.892 | 0.872 | 0.895 | 0.875 |
|  | gpt-4o | **0.912** | **0.896** | **0.918** | **0.899** | **0.921** | **0.902** |
| TACO | Jackalope-7b | 0.763 | 0.827 | 0.767 | 0.831 | 0.774 | 0.840 |
|  | Llama-2-13b-chat | 0.886 | 0.875 | 0.893 | 0.881 | 0.902 | 0.891 |
|  | gpt-3.5-turbo | 0.844 | 0.867 | 0.844 | 0.867 | 0.847 | 0.869 |
|  | gpt-4-turbo | *0.999* | *0.998* | *0.999* | *0.998* | 0.999 | *0.998* |
|  | gpt-4o | 0.995 | 0.994 | 0.997 | 0.995 | 0.999 | 0.996 |

**Table 3: MATCH stage results assessing the quality and semantic accuracy of original symptom mentions extracted by different models. Bold values indicate the best performance for each prompting method, while italic values show the overall best performance across methods.**

| Prompt Type | Models | BLEU | Fuzzy | OpenAI Similarity |
|---|---|---|---|---|
| TASI | Jackalope-7b | 0.238 | 0.728 | 0.604 |
|  | Llama-2-13b-chat | 0.197 | 0.638 | 0.501 |
|  | gpt-3.5-turbo | 0.320 | 0.725 | 0.665 |
|  | gpt-4-turbo | 0.377 | 0.770 | 0.721 |
|  | gpt-4o | **0.382** | **0.788** | **0.725** |
| TACO | Jackalope-7b | 0.182 | 0.621 | 0.501 |
|  | Llama-2-13b-chat | 0.214 | 0.717 | 0.541 |
|  | gpt-3.5-turbo | 0.300 | 0.710 | 0.625 |
|  | gpt-4-turbo | *0.465* | *0.862* | *0.775* |
|  | gpt-4o | 0.435 | 0.856 | 0.766 |

(1) **RQ1**: How do the TACO and TASI prompting strategies impact the performance of LLMs in the symptom coding task?

(2) **RQ2**: How do different LLMs perform on the symptom coding task when evaluated with the same prompt design?

(3) **RQ3**: How do LLMs perform on the top 50 common and rare symptom datasets, and how does performance vary across these cases?

## 5.1 Experimental Setting

*5.1.1 Dataset Description.* In the evaluation, we utilized the SYMPCODER dataset, comprising SYMPCODER-Full, SYMPCODER-Common-50, and SYMPCODER-Rare-50, as introduced in Section 3. These subsets enable comprehensive evaluation of LLM performance across both frequently and infrequently observed symptoms, providing insights into model robustness and adaptability. More details on the dataset and its creation can be found in Section 3.

*5.1.2 Evaluation Metrics.* The evaluation metrics used to assess model performance are structured according to the two stages of the evaluation process defined in Section 4.1.3. Across both stages, we employ the following metrics: Exact Match (EM) [36], Fuzzy

Match [9], Precision [14], Recall [14], BLEU Score [32], and Cosine Similarity [35]. Each metric provides a unique perspective on the models' performance in extracting and linking adverse event symptoms and their original mentions.

In the first stage (LINK), the evaluation focuses on term linking, with Exact Match (EM), Fuzzy Match, Precision, and Recall serving as the primary metrics. EM measures accuracy by verifying if the extracted terms exactly match the human-annotated terms. Fuzzy Match accounts for minor discrepancies by allowing approximate matches between terms. The combined EM and Fuzzy Match approach is applied ultimately, where Exact Match is followed by Fuzzy Match for unmatched terms, ensuring a more comprehensive and accurate evaluation. Precision and recall metrics quantify the accuracy and completeness of term linking, providing insights into the models' ability to capture and associate relevant terms with their respective medical codes.

In the second stage (MATCH), the evaluation centers on the quality of the original mentions generated by the models. we assess mention quality using BLEU, Fuzzy Match, and Cosine Similarity—measuring n-gram precision, tolerating minor wording differences, and evaluating semantic coherence via vector embeddings, respectively. The embeddings for cosine similarity are derived from OpenAI Embedding models, ensuring robust semantic comparisons.

*5.1.3 Implementation Details.* In our model implementation, we utilized the Jackalope-7B and Llama2-13B architectures, employing models sourced from The Bloke via Huggingface. Additionally, for GPT models, we accessed OpenAI APIs to obtain embeddings and inference results. To regulate the output length, we configured the parameter max_new_token to a value of 256, while adjusting the temperature within the range of 0.3 to 0.5 to optimize performance. Our experiments were conducted on hardware comprising two Nvidia RTX A5000 GPUs with 298.5GB disk space and 104GB RAM. The processing time for each model, spanning from obtaining inference results to generating evaluations, remained within 2 hours, ensuring timely and efficient execution.
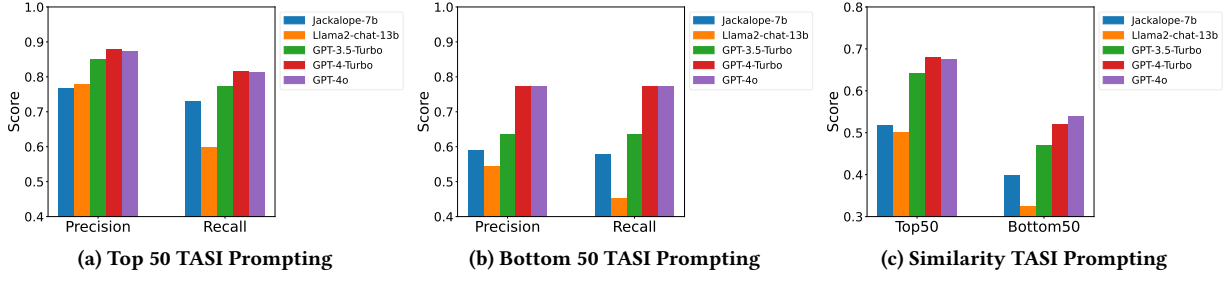
(a) Top 50 TASI Prompting     (b) Bottom 50 TASI Prompting     (c) Similarity TASI Prompting

**Figure 4: Common Rare Case Analysis with TASI Prompting on SYMPCODER**



(a) Top 50 TACO Prompting     (b) Bottom 50 TACO Prompting     (c) Similarity TACO Prompting
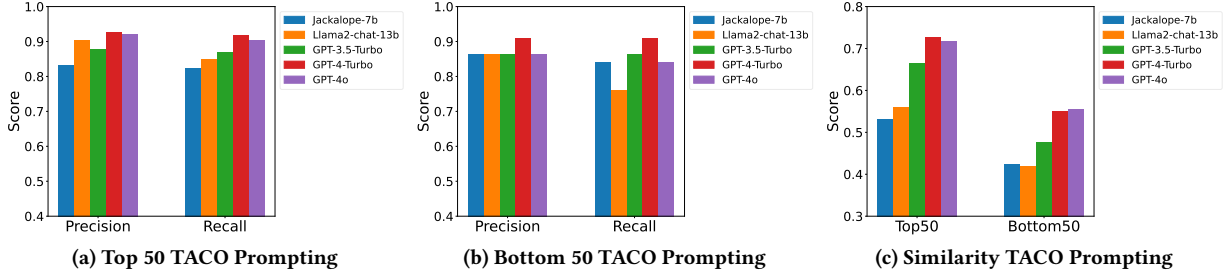
**Figure 5: Common Rare Case Analysis with TACO Prompting on SYMPCODER**

## 5.2 Experimental Results

*5.2.1* ***Prompt Variation (RQ1).*** To investigate the capacities of
TACO prompting method, we compare TACO and TASI perfor-
mances across both stages of evaluation. The results from Table 2
and Table 3 consistently demonstrate TACO's overall superiority in
enhancing performance across most models in handling complex
linking and extraction tasks.

In the *LINK stage* shown in Table 2, TACO achieves higher pre-
cision and recall scores for most models, with GPT-4o and GPT-4-
Turbo consistently leading in performance. These results underline
TACO's ability to handle complex linking tasks effectively. However,
Jackalope-7b exhibits a unique trend, where TASI slightly outper-
forms TACO for precision. This anomaly suggests that smaller
models like Jackalope-7b may benefit from the sequential nature of
TASI due to their limited capacity to handle the integrated context
provided by TACO.

In the *MATCH stage* displayed in Table 3, TACO continues to
outperform TASI across BLEU, fuzzy match, and similarity scores,
particularly for advanced models like GPT-4o and GPT-4-Turbo.
For instance, GPT-4-Turbo achieves a BLEU score of 0.465 and
a fuzzy match score of 0.862, showcasing its ability to generate
coherent and accurate mentions under TACO prompting method.
However, models such as GPT-3.5-Turbo and Jackalope-7b exhibit a
slight decline in BLEU and similarity scores with TACO. While the
reasons for this decline are not entirely clear, it suggests potential
limitations in how these models process the additional context
provided by TACO.

Overall, TACO emerges as the more effective strategy, partic-
ularly for advanced LLMs like GPT-4o and GPT-4-Turbo, which
leverages its integrated design to achieve robust and accurate perfor-
mance. The results reaffirm TACO's ability to streamline complex

tasks and enhance both precision and recall, making it a superior
prompting method for our task.

*5.2.2* ***Model Performance (RQ2).*** To evaluate the impact of
different LLMs on vaccine adverse symptom coding, we compared
their performance across two evaluation stages: the LINK stage
and the MATCH stage. As shown in Tables 2 and 3, advanced
LLMs such as GPT-4-Turbo and GPT-4o consistently outperform
smaller models like Jackalope-7b and Llama2-13b, demonstrating
their superior ability to handle the complexities of coding vaccine
adverse symptoms.

In the *LINK stage*, which evaluates how well models align symp-
toms with suggested terms, GPT-4-Turbo achieves near-perfect
scores under TACO prompting, with an EM Precision of 0.999 and
EM Recall of 0.998, reflecting its ability to both accurately and
comprehensively code symptoms. Similarly, GPT-4o achieves ex-
ceptional performance under TASI prompting, with the highest
EM-Fuzzy Precision (0.921) and EM-Fuzzy Recall (0.902). These
results highlight the advanced contextual understanding of these
models, enabling them to reliably handle the linking of symptoms
to standardized terms.

In the *MATCH stage*, which assesses the semantic accuracy and
coherence of original mentions, GPT-4-Turbo again leads under
TACO prompting with a BLEU score of 0.465 and a cosine similar-
ity score of 0.775. These findings confirm that advanced models
not only excel at linking symptoms to standardized terms but also
maintain semantic accuracy and coherence in their outputs. In con-
trast, smaller models like Jackalope-7b and Llama2-13b demonstrate
consistently lower performance, with Jackalope-7b showing signif-
icant difficulty in generating coherent mentions, particularly under
TACO prompting.

**Table 4: Original Mention vs Model Output from GPT4-turbo. This table compares human-annotated and model-extracted symptoms for six selected standard symptoms, with three examples each randomly chosen from the top 50 common and rare symptoms. The counts in parentheses indicate the frequency of mentions for each term.**

| Standard Symptoms | Human Annotated Symptoms | Model Extracted Symptoms |
|---|---|---|
| Fatigue | fatigue (59), tiredness (20), exhaustion (3) | fatigue (55), tiredness (23), exhaustion (6), weakness (3), wiped out (1) |
| Pyrexia | pyrexia (9), fever (102), elevated/inc temp (1) | pyrexia (2), fever (108), elevated/inc temp (1), low grade temp (1), Temp of 103 degrees (3) |
| Dizziness | Dizziness (33), light headed (5), woozy (2), vertigo (2) | Dizziness (33), light headed (5), woozy (2), wobbly legs (1) |
| Eye Irritation | eye irritation (1), burning eyes (1) | burning eyes (1) |
| Facial Spasm | facial muscle spasm (1) | facial muscle spasm (1) |
| Blister | blisters (1), blister (1) | blisters (1) |

An interesting observation is the relative performance of Jackalope-7b, which surpasses Llama2-13b in the LINK stage under TASI. This unexpected trend could stem from Jackalope-7b's simpler architecture and focused training, which might align better with the structured nature of TASI prompts. However, this advantage diminishes in the TACO scenario, where the added contextual complexity benefits more advanced models like GPT-4-Turbo and GPT-4o. This suggests that larger and more advanced LLMs are better equipped to handle the challenges of contextual integration inherent in TACO prompting.

Overall, the results demonstrate that GPT-4-Turbo and GPT-4o are the most effective models for vaccine adverse symptom coding. Their advanced architectures and contextual capabilities enable them to achieve consistently high performance across both evaluation stages, making them the most reliable choices for addressing the complexities of this task.

*5.2.3 **Common Rare Symptoms Analysis (RQ3)**.* To evaluate the performance of LLMs on datasets containing the top 50 most common and bottom 50 least common symptoms, we conducted analyses using both TASI and TACO prompting methods. Figures 4 and 5 provide insights into how models handle frequent versus infrequent adverse symptoms, with a focus on model performance under each prompting strategy before comparing the methods.

Under *TACO prompting*, as shown in Figure 5, models like GPT-4o and GPT-4-Turbo consistently exhibit higher precision and recall for the top 50 most common symptoms (Figure 4a). This demonstrates their ability to accurately and comprehensively capture frequent symptoms due to their advanced architectures and extensive training. However, for the bottom 50 least common symptoms (Figure 4b), performance varies more significantly. GPT-4-Turbo maintains relatively robust recall and precision, while GPT-4o and Llama2-13b-chat experience a noticeable decline in both metrics. This decrease suggests that rare symptoms, despite their distinctiveness, are challenging to extract consistently under TACO prompting, likely due to their sparse representation in training data. The similarity scores (Figure 4c) further support these observations, with GPT-4o and GPT-4-Turbo achieving the highest semantic accuracy

for both common and rare symptoms, while smaller models like Llama2-13b-chat perform less effectively.

Under *TASI prompting*, as depicted in Figure 4, similar trends emerge, but the recall values for rare symptoms exhibit a sharper decline compared to TACO prompting (Figure 5b). For the top 50 most common symptoms (Figure 5a), all models maintain high precision, with GPT-4o and GPT-4-Turbo again leading in recall. However, for the bottom 50 rare symptoms, recall values decrease more dramatically for models like Llama2-13b-chat and GPT-4o, indicating that TASI's sequential structure is less effective at capturing rare terms. Precision scores for rare symptoms also decline under TASI, although not as significantly as recall. The similarity scores (Figure 5c) show consistent results, with GPT-4o and GPT-4-Turbo outperforming smaller models, particularly in the rare case analysis.

When comparing the two prompting methods, TACO consistently demonstrates advantages over TASI for capturing common symptoms, with higher recall and precision for most models. For rare symptoms, TACO prompting provides better consistency in recall and precision across advanced models like GPT-4-Turbo and GPT-4o, although challenges remain for smaller models. The integrated contextual design of TACO prompting likely allows models to handle both frequent and infrequent patterns more effectively, while TASI's sequential structure introduces limitations, particularly for rare cases.

Overall, the comparison reveals that advanced models such as GPT-4o and GPT-4-Turbo consistently outperform smaller models under both prompting methods. However, TACO prompting offers clear advantages in handling the variability of both common and rare adverse symptoms, highlighting its effectiveness as a strategy for vaccine adverse symptom coding tasks.

*5.2.4 **Case Study: Common vs Rare Mention**.* Table 4 evaluates the ability of GPT-4 Turbo to capture common and rare terms in clinical text. The selected terms—"Eye Irritation," "Facial Spasm," "Blister," "Fatigue," "Pyrexia," and "Dizziness"—are derived from the top 50 (common) and bottom 50 (rare) terms in the dataset. This selection provides a balanced analysis of the model's performance across varying levels of term frequency.

**Table 5: Illustrative Example of Model Prediction Comparison. This table compares the outputs of various models against the gold labels for a selected VAERS report, illustrating the models' abilities to identify and link symptoms from the given text description to standardized terms. Text in the same color refers to the same symptom, while red text indicates incorrect predictions compared to the gold labels.**

| Text Descriptions | 24 hours after vaccines had temp of 103.2 degrees. Gave Tylenol. On Tues., lack of appetite. On Wed, blotchy rash on face & body. On Thurs, rash lessened. On Fri, no rash. Fever lasted about 24 hours. Has had cold for approx. 3 weeks/runny nose, occasional fever). Directed to inform physician of vaccines and events. |
|---|---|
| Gold Labels | Pyrexia: [Fever], Decreased appetite: [lack, of, appetite], Rhinorrhoea: [weeks/runny, nose], Rash: [blotchy, rash, on, face, & body], Rash macular: [blotchy, rash] |

| Model | Output |
|---|---|
| Jackalope-7b | Pyrexia: [fever, temp of 103.2 degrees, lack of appetite, on thursday, no rash, on friday], Decreased appetite: [on tuesday, on wednesday, on friday], Rhinorrhoea: [cold for approx. 3 weeks/runny nose], Rash: [blotchy rash on face & body, on wednesday, on thursday, on friday], Rash macular: [rash on face & body, rash on wednesday, rash on thursday, rash on friday] |
| Llama2-chat-13b | Pyrexia: [fever], Decreased appetite: [lack of appetite], Rhinorrhoea: [none], Rash: [blotchy rash, rash lessened, no rash], Rash macular: [none] |
| GPT-3.5-Turbo | Pyrexia: [temp, fever], Decreased appetite: [lack of appetite], Rhinorrhoea: [cold, runny nose], Rash: [blotchy rash], Rash macular: [none] |
| GPT-4-Turbo | Pyrexia: [temp of 103.2 degrees, Fever lasted about 24 hours], Decreased appetite: [lack of appetite], Rhinorrhoea: [runny nose], Rash: [blotchy rash on face & body], Rash macular: [none] |
| GPT-4o | Pyrexia: [temp of 103.2 degrees, fever lasted about 24 hours, occasional fever], Decreased appetite: [lack of appetite], Rhinorrhoea: [runny nose], Rash: [blotchy rash on face & body, rash lessened, no rash], Rash macular: [none] |

For common terms like "Fatigue" and "Dizziness," GPT-4 Turbo enriched its outputs with additional contextual details, such as "wiped out" for "Fatigue" and "wobbly legs" for "Dizziness." While this demonstrates the model's ability to infer related terms, the over-generalization occasionally introduced extraneous phrases that deviated from the original clinical mention. In the case of "Pyrexia," the model generated descriptors such as "low-grade temp" and "temp of 103 degrees," which, while adding specificity, risked fragmenting data due to overly detailed outputs.

Rare terms, on the other hand, posed greater challenges. For example, "Eye Irritation" was simplified to "burning eyes," reflecting a preference for informal phrasing over formal clinical terminology. Similarly, terms like "Facial Spasm" and "Blister" were extracted correctly but lacked variation, suggesting limited diversity in the model's outputs for rare terms. These issues may stem from the sparse representation of rare cases in the training data, causing the model to favor familiar informal phrases and rely on heuristics that prioritize common terms over rare, specific ones.

In summary, GPT-4 Turbo performs well with common terms but struggles to preserve formal phrasing for rare ones. This reliance on sparse data underscores the need to refine prompts and add post-processing to better handle underrepresented terms.

*5.2.5* **Case Study: Mention Discrepancy**. Table 5 compares outputs from five models(Jackalope-7b, Llama2-chat-13b, GPT-3.5-Turbo, GPT-4 Turbo, and GPT-4o) with gold-standard annotations, revealing significant differences in capturing nuanced and rare mentions.

Advanced models like GPT-4 Turbo and GPT-4o consistently provided detailed and contextually enriched outputs. For instance, GPT-4o added specific details such as "occasional fever," closely aligning with the gold standard while offering richer context. Jackalope-7b, on the other hand, tended to over-annotate by introducing temporal phrases like "on Wednesday" and "on Friday," which, while detailed, often introduced unnecessary noise. Llama2-chat-13b struggled significantly, failing to identify certain mentions, particularly less frequent ones like "Rash Macular," highlighting its limitations in capturing rare or complex terms.

For common terms like 'Decreased Appetite,' all models except Jackalope-7b closely matched the gold standard. However, performance on rare terms like 'Rash Macular' varied: Jackalope-7b offered multiple details, while Llama2-chat-13b and GPT-3.5-Turbo often returned 'none.'

This analysis reinforces a clear trend: higher-capacity models generally excel in capturing nuanced terms, whereas smaller models struggle with rare and complex ones. Even the most advanced models miss infrequent terms occasionally, highlighting an ongoing need for refined prompting strategies and post-processing pipelines to ensure clinical relevance and consistency.

## 6 CONCLUSIONS

In this paper, we addressed the challenges of accurate symptom coding from unstructured clinical data, particularly focusing on the nuanced extraction and linking of symptoms in vaccine safety reports such as those in VAERS. To benchmark the performance of LLMs on this tailored task, we introduced SYMPCODER, a human-annotated dataset that captures both common and rare symptoms, providing a robust foundation for evaluating adverse event extraction and linking tasks. Our proposed TACO prompting framework unifies symptom extraction and linking into a single process, significantly improving contextual accuracy and reducing information loss typically associated with traditional separate workflows. In addition, we introduced a two-stage evaluation framework, comprising the LINK and MATCH phases, which enables a detailed assessment of model performance by focusing on both symptom linking and original

mention matching. This framework revealed key differences in how models handle common versus rare symptoms, offering a nuanced understanding of LLM capabilities. Through extensive experiments with several state-of-the-art LLMs, we demonstrated the superior performance of TACO, emphasizing the impact of task-specific prompt design on model accuracy. Our findings not only highlight the effectiveness of TACO prompting in enhancing model performance but also underscore its potential as a flexible framework for developing tailored coding tasks in clinical natural language processing. Future work will explore broader applications of TACO, aiming to develop more innovative and impactful solutions in the medical field.

## ACKNOWLEDGMENTS

## REFERENCES

[1] Sajid Ali, Khalid Masood, Anas Riaz, and Amna Saud. 2022. Named Entity Recognition using Deep Learning: A Review. *2022 International Conference on Business Analytics for Technology and Security (ICBATS)* (2022), 1–7. https://api.semanticscholar.org/CorpusID:248408008

[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *international semantic web conference*. Springer, 722–735.

[3] Leonard E Baum and Ted Petrie. 1966. Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics* 37, 6 (1966), 1554–1563.

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

[5] Joseph S Boyle, Antanas Kascenas, Pat Lok, Maria Liakata, and Alison Q O'Neil. 2023. Automated clinical coding using off-the-shelf large language models. *arXiv preprint arXiv:2310.06552* (2023).

[6] Samuel Broscheit. 2020. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. *arXiv preprint arXiv:2003.05473* (2020).

[7] Elizabeth G Brown, L Wood, and S Wood. 1999. The medical dictionary for regulatory activities (MedDRA). *Drug Safety* 20, 2 (1999), 109–117. https://doi.org/10.2165/00002018-199920020-00002

[8] Centers for Disease Control and Prevention (CDC), Food and Drug Administration (FDA), agencies of the U.S. Department of Health and Human Services (HHS). 1990. Vaccine Adverse Event Reporting System. https://vaers.hhs.gov/data.html.

[9] Surajit Chaudhuri, Kris Ganjam, Venkatesh Ganti, and Rajeev Motwani. 2003. Robust and efficient fuzzy match for online data cleaning *(SIGMOD '03)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/872757.872796

[10] Roger A. Côté and Stanley Robboy. 1980. Progress in Medical Information Management: Systematized Nomenclature of Medicine (SNOMED). *JAMA* 243, 8 (02 1980), 756–762.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[12] Wikimedia Foundation. 2001. *Wikimedia Downloads.* https://dumps.wikimedia.org

[13] Evan French and Bridget T McInnes. 2023. An overview of biomedical entity linking throughout the years. *Journal of biomedical informatics* 137 (2023), 104252.

[14] Cyril Goutte and Eric Gaussier. 2005. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In *European conference on information retrieval*. Springer, 345–359.

[15] Yu Gu, Sheng Zhang, Naoto Usuyama, Yonas G. Woldesenbet, Cliff Wong, Praneeth Sanapathi, Mu-Hsin Wei, Naveen Valluri, Erika Strandberg, Tristan Naumann, and Hoifung Poon. 2023. Distilling Large Language Models for Biomedical Knowledge Extraction: A Case Study on Adverse Drug Events. *ArXiv* abs/2307.06439 (2023). https://api.semanticscholar.org/CorpusID:259847622

[16] Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2019. ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission. *arXiv:1904.05342* (2019).

[17] Basra Jehangir, Saravanan Radhakrishnan, and Rahul Agarwal. 2023. A survey on Named Entity Recognition — datasets, tools, and methodologies. *Natural Language Processing Journal* 3 (2023), 100017. https://doi.org/10.1016/j.nlp.2023.100017

[18] Shaoxiong Ji, Matti Hölttä, and Pekka Marttinen. 2021. Does the magic of BERT apply to medical code assignment? A quantitative study. *Computers in biology and medicine* 139 (2021), 104998.

[19] Shaoxiong Ji, Xiaobo Li, Wei Sun, Hang Dong, Ara Taalas, Yijia Zhang, Honghan Wu, Esa Pitkänen, and Pekka Marttinen. 2024. A Unified Review of Deep Learning for Automated Medical Coding. *ACM Comput. Surv.* (may 2024). https://doi.org/10.1145/3664615 Just Accepted.

[20] David Kartchner, Jennifer Deng, Shubham Lohiya, Tejasri Kopparthi, Prasanth Bathala, Daniel Domingo-Fernández, and Cassie S Mitchell. 2023. A Comprehensive Evaluation of Biomedical Entity Linking Models. In *The 2023 Conference on Empirical Methods in Natural Language Processing.* https://openreview.net/forum?id=5Ob6DsDv2V

[21] Imed Keraghel, Stanislas Morbieu, and Mohamed Nadif. 2024. A survey on recent advances in named entity recognition. arXiv:2401.10825 [cs.CL] https://arxiv.org/abs/2401.10825

[22] Donghyeon Kim, Jinhyuk Lee, Chan Ho So, Hwisang Jeon, Minbyul Jeong, Yonghwa Choi, Wonjin Yoon, Mujeen Sung, and Jaewoo Kang. 2019. A Neural Named Entity Recognition and Multi-Type Normalization Tool for Biomedical Text Mining. *IEEE Access* 7 (2019), 73729–73740. https://doi.org/10.1109/ACCESS.2019.2920708

[23] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* 36, 4 (2020), 1234–1240.

[24] Fei Li and Hong Yu. 2019. ICD Coding from Clinical Text Using Multi-Filter Residual Convolutional Neural Network. *Proceedings of the ... AAAI Conference on Artificial Intelligence. AAAI Conference on Artificial Intelligence* 34 5 (2019), 8180–8187. https://api.semanticscholar.org/CorpusID:208527485

[25] Tianyi Li, Ping Wang, Tian Shi, Yali Bian, and Andy Esakia. 2023. Task as Context: A Sensemaking Perspective on Annotating Inter-Dependent Event Attributes with Non-Experts. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*.

[26] Wing Lian, Bleys Goodson, Guan Wang, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". 2023. *Jackalope 7B: Mistral-7B Model Multi-Turn Chat tuned on Filtered OpenOrcaV1 GPT-4 Dataset.*

[27] Chang Lu, Chandan Reddy, Ping Wang, and Yue Ning. 2023. Towards Semi-Structured Automatic ICD Coding via Tree-based Contrastive Learning. In *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.), Vol. 36. Curran Associates, Inc., 68300–68315. https://proceedings.neurips.cc/paper_files/paper/2023/file/d74f9efa1d8ca30b31d65cef8de7c2bf-Paper-Conference.pdf

[28] Darshini Mahendran and Bridget T. McInnes. 2021. Extracting Adverse Drug Events from Clinical Notes. arXiv:2104.10791 [cs.CL] https://arxiv.org/abs/2104.10791

[29] OpenAI. 2023. *GPT-3.5 Turbo.*

[30] OpenAI. 2023. *GPT-4 Turbo.*

[31] OpenAI. 2023. *GPT-4o.*

[32] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation *(ACL '02)*. Association for Computational Linguistics, USA. https://doi.org/10.3115/1073083.1073135

[33] Nita Patil, Ajay Patil, and B.V. Pawar. 2020. Named Entity Recognition using Conditional Random Fields. *Procedia Computer Science* 167 (2020), 1181–1188. https://doi.org/10.1016/j.procs.2020.03.431 International Conference on Computational Intelligence and Data Science.

[34] Bo Qiao, Zhuoyang Zou, Yu Huang, Kui Fang, Xinghui Zhu, and Yiming Chen. 2022. A joint model for entity and relation extraction based on BERT. *Neural Computing and Applications* (2022), 1–11.

[35] Faisal Rahutomo, Teruaki Kitasuka, Masayoshi Aritsugi, et al. 2012. Semantic cosine similarity. In *The 7th international student conference on advanced science and technology ICAST*, Vol. 4. University of Seoul South Korea, 1.

[36] Simone Santini and Ramesh Jain. 1996. Similarity matching. In *Recent Developments in Computer Vision: Second Asian Conference on Computer Vision, ACCV'95 Singapore, December 5–8, 1995 Invited Session Papers 2*. Springer, 571–580.

[37] Wei Shen, Yuhan Li, Yinan Liu, Jiawei Han, Jianyong Wang, and Xiaojie Yuan. 2023. Entity Linking Meets Deep Learning: Techniques and Solutions. *IEEE Transactions on Knowledge and Data Engineering* 35, 3 (2023), 2556–2578. https://doi.org/10.1109/TKDE.2021.3117715

[38] Ken Thompson. 1968. Programming techniques: Regular expression search algorithm. *Commun. ACM* 11, 6 (1968), 419–422.

[39] Hugo Touvron, Louis Martin, Kevin Stone, et al. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288* (2023).

[40] Zhichao Yang, Sanjit Singh Batra, Joel Stremmel, and Eran Halperin. 2023. Surpassing GPT-4 Medical Coding with a Two-Stage Approach. *arXiv preprint arXiv:2311.13735* (2023).