LLMs Accelerate Annotation for Medical Information Extraction

Akshay Goel* GOELAK@GOOGLE.COM
Almog Gueta* ALMOGGUETA@GOOGLE.COM
Omry Gilon* OMRYGILON@GOOGLE.COM
Chang Liu CHANGLIUSTAT@GOOGLE.COM
Sofia Erell ROVINSKY@GOOGLE.COM
Lan Huong Nguyen LANHUONGNGUYEN@GOOGLE.COM
Xiaohong Hao HAOXH@GOOGLE.COM
Bolous Jaber BOLOUS@GOOGLE.COM
Shashir Reddy SHASHIR@GOOGLE.COM
Rupesh Kartha RUPESHKARTHA@GOOGLE.COM
Jean Steiner JSTEINER@GOOGLE.COM
Itay Laish ITAYLAISH@GOOGLE.COM
Amir Feder AFEDER@GOOGLE.COM

Google Research

Abstract

The unstructured nature of clinical notes within electronic health records often conceals vital patient-related information, making it challenging to access or interpret. To uncover this hidden information, specialized Natural Language Processing (NLP) models are required. However, training these models necessitates large amounts of labeled data, a process that is both time-consuming and costly when relying solely on human experts for annotation. In this paper, we propose an approach that combines Large Language Models (LLMs) with human expertise to create an efficient method for generating ground truth labels for medical text annotation. By utilizing LLMs in conjunction with human annotators, we significantly reduce the human annotation burden, enabling the rapid creation of labeled datasets. We rigorously evaluate our method on a medical information extraction task, demonstrating that our approach not only substantially cuts down on human intervention but also maintains high accuracy. The results highlight the potential of using LLMs to improve the utilization of unstructured clinical data, allowing for the swift deployment of tailored NLP solutions in healthcare.

Keywords: Medical NLP, Large Language Models, Data Annotation

* These authors contributed equally.

1. Introduction

Medical NLP holds promise for transforming healthcare by extracting valuable insights from vast amounts of unstructured clinical data (Feder et al., 2022; Zhang et al., 2022; Liu et al., 2023). These NLP insights can improve data organization and highlight critical clinical information for users (Landolsi et al., 2022). However, a significant impediment to this transformation is the challenge of obtaining high-quality labeled data for training and evaluating machine learning models specific to medical NLP tasks (Hakala and Pyysalo, 2019). The crux of the issue resides in the human labeling process. Manual annotation, especially when it necessitates the expertise of medical professionals, is labor-intensive and expensive. Given the intricacies of medical data, achieving a comprehensive and accurate set of text spans corresponding to specific medical entities often requires multiple annotators or iterative rounds of annotation.

To alleviate this bottleneck, in this work, we test the potential of LLMs to streamline the labeling process (Ding et al., 2022; Liu et al., 2022). LLMs have recently gained immense popularity due to their generalizability across various tasks (Zhou et al., 2023b). *Prompt engineering* is the optimization of natural language instructions to effectively condition an LLM (Reynolds and McDonell, 2021). When incorporating examples of the task, this is termed *few-shot learning* (Brown et al., 2020). Notably, *prompt engineering*

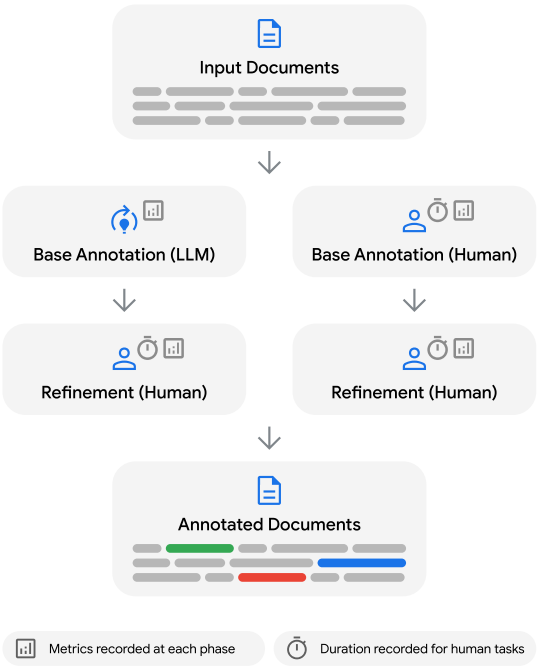

Figure 1: Evaluation overview comparing annotation efficiency between LLM-assisted (left) and human-only (right) annotation workflows using the identical document set. Duration and metrics of each phase are benchmarked against *i2b2* ground-truth labels.

has shown resilience against overfitting (Akinwande et al., 2023). Although LLMs offer the allure of automating data labeling, they have two main limitations: their propensity for generating unconstrained outputs and their inability to guarantee expert-level performance (Agrawal et al., 2022).

Acknowledging this, we introduce a two-step method that combines the capabilities of LLMs with human expertise. Initially, the LLM, conditioned in a *few-shot learning* setup, generates *Base Annotations*. Subsequently, these annotations are refined by medical annotation experts to produce *Refined Annotations*.

To validate the efficacy of our approach, we compare it against a standard annotation pipeline, which utilizes a round of human Base Annotations followed by expert human Refinement Annotations. Figure 1 provides an overview of our evaluation, comparing an LLM-assisted annotation process to a human-only annotation process. Our empirical evaluation, focused on the medication-extraction task, highlights the potential of our approach to significantly reduce the human annotation workload while maintaining expert-level high-quality annotations. Through this study, we aim to establish a more efficient route for labeling data to deploy robust medical NLP solutions, ultimately driving advancements in patient care and healthcare operations.

Our contributions are:

1. We empirically test the utility of LLMs in annotating clinical narratives, a core bottleneck in Medical NLP (see Figure 1).
2. We present a method that significantly accelerates the annotation process while maintaining expert-level quality.
3. We produce a set of labels for medication extraction using a public medical dataset (MIMIC-IV-Note) (Johnson et al., 2023).[1]

The paper is structured as follows. In Section 2, we discuss prior work on clinical information extraction and the challenges these systems face. In Section 3, we present the data and task with which we are experimenting. Subsequently, in Section 4 and Section 5, we describe the human annotators and the LLM pipeline that supports them, respectively. Using these annotators, we then present in Section 6 our empirical findings and discuss the efficiency gains from using our annotation approach. Finally, in Section 7 we discuss paths forward for utilizing LLMs to enhance Medical NLP while ensuring reliability.

2. Related Work

Information Extraction from Clinical Notes. With the rise of Electronic Health Records (EHRs), there is a pressing need for efficient clinical information extraction systems (Dash et al., 2019; Birkhead et al., 2015). Extracting patient information from medical notes is pivotal in clinical-NLP, with many standardized datasets and competitions (Uzuner, 2009; Savova et al., 2010; Jensen et al., 2012; Ford et al., 2016; Zhu et al., 2018). Modern solutions often employ the transformer architecture for these tasks (Vaswani et al., 2017; Peng et al., 2019; Yadav and Bethard, 2019; Si et al., 2019; Lee et al., 2020). These approaches have shown promise in tasks like biomedical Named Entity Recognition (NER) but are constrained by their reliance on extensive domain-specific labels (Hakala and Pyysalo, 2019).

1. The labels can be accessed at PhysioNet (Goel et al., 2023; Goldberger et al., 2000 (June 13)).

LLMs in Medical NLP. LLMs have transformed medical NLP, excelling in tasks such as medical question-answering and unsupervised clinical information extraction (Driess et al., 2023; Singhal et al., 2022; Agrawal et al., 2022). Their adaptability allows for extension across various medical domains, from oncology to telemedicine, and their ability to be distilled into task-specific models, such as UniversalNER, highlights their versatility (Zhou et al., 2023a).

Active Learning in Medical NLP. Active learning has emerged as a solution to reduce manual labeling in medical NLP. For instance, Kholghi et al. (2015) utilized active learning for medical concept extraction, while Wei et al. (2019) introduced a cost-aware method. Feder et al. (2020) combined BERT (Devlin et al., 2019) with active learning, emphasizing iterative manual refinement. While promising, these methods require explicit model parameter tuning to fit a data distribution and task.

In this work, we propose an LLM-based approach that significantly enhances medical information extraction over traditional methods, without necessitating model parameter tuning.

3. Data and Task

3.1. i2b2 Dataset

Our experiment utilizes the 2009 *i2b2* Workshop on NLP Challenges dataset (Uzuner et al., 2010), consisting of de-identified clinical discharge summaries (*documents*). Each document contains labels of medication *fields*, which includes the medication's *name*, *dosage*, *mode* (route) of administration, *frequency*, *duration* of administration, and the *reason* for administration.[2] The fields of a single medication are linked together to create a *medication entry*.

3.2. Annotation Task

The *i2b2* dataset provides high-quality labels for a medication-extraction task, divided into two subtasks: Named Entity Recognition (NER) for medication fields and Relationship Extraction (RE) to associate these fields with specific medication entries. This involves a two-step process:

1. NER Task: Identifies the textual spans corresponding to each medication field, marking the start and end token indices.
2. RE Task: Forms medication entries by linking related fields for each medication mention.

A visual representation of the labels and task can be seen in Figure 2.

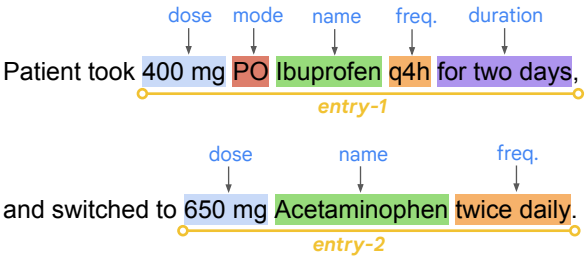

Figure 2: Example of a medication-extraction annotation, showcasing two medication entries underlined in yellow. Field types are highlighted by color.

3.3. Data Splits & Pre-Processing

The dataset, comprising 261 documents, was randomly divided into three subsets: the 'pilot set' (16 documents, 6%), used to ensure human annotators' understanding of the guidelines and task and to select expert raters for the refinement task (see §4); the 'development set' (29 documents, 11%), for selecting optimal prompts and parameters for the LLM pipeline; and a 'test set' (216 documents, 83%), for measuring the performance of both annotation methods, with human or LLM Base Annotations. Table 1 presents the number of occurrences of each field in each dataset subset.

	Name	Dose	Mode	Frequency	Duration
Total	9,237	4,646	3,491	4,182	562
Pilot	511	300	180	282	38
Development	1,123	523	437	452	57
Test	7,603	3,823	2,874	3,448	467

Table 1: Number of occurrences for each labeled field across all data subsets of the *i2b2* 2009 dataset.

Minimal preprocessing was performed on the dataset to remove leading and trailing punctuation, spaces, and tabs from label spans. In addition, discontinuous medication fields were infrequent, consti-

2. Uzuner et al. (2010) note a *narrative* field, distinguishing list structure from narrative text in discharge summaries. This was omitted as our focus is classification, not extraction. Additionally, the *reason* field was excluded due to its high variance in human labeling (Uzuner et al., 2010).

tuting less than 1% of the data, and were therefore excluded to streamline the preprocessing stage.

4. Human Annotations

Human annotations in the medical domain often necessitate medical expertise and extensive training, making the process both time-consuming and costly. Annotation typically involves marking spans of text in lengthy documents, requiring high focus to avoid errors. Rater fatigue is thus a significant concern.

Annotation Workflow: To mitigate the above challenges, our annotation process comprises two sequential phases:

1. **Base Annotation Phase**: Initial labels are added to blank documents by a rater.

2. **Refinement Annotation Phase**: The base labels are refined and corrected by specialized *expert raters*.

Annotation Platform and Guidelines Annotations were carried out on a text labeling platform, configured to support the medication-extraction schema detailed in §3.2. The platform consists of a viewing panel to display raw medical documents and a question panel to guide the annotation process. Special cases like ambiguity between frequency and duration were handled according to specific rules based on the original *i2b2* guidelines.

Raters Team Composition and Expertise Our annotation team was composed of seven raters, each with a medical background and an extensive experience in medical NLP tasks. Their medical training ranged from medical assistant to attending physician.

A pilot task was conducted to select the *expert raters* based on their performance in annotating a pilot set of documents, resulting in the selection of three expert raters. To optimize the use of rater time and expertise, the more exact Refinement Annotations were performed solely by *expert raters*. Additionally, both *expert raters* and *non-expert raters* contributed to Base Annotations. This setup allowed us to perform a subgroup analysis on expertise-level for the annotation pipeline.

Rater Tasks Raters were responsible for three types of tasks, where documents were randomly allocated to ensure no rater annotated a document twice:

1. **Base Rater Label**: Base Annotation of medical documents (labeling documents from scratch). Performed by *all raters*.

2. **Refined Rater Label**: Refinement Annotation of labeled documents produced by another rater. Performed by *expert raters*.

3. **Refined LLM Label**: Refinement Annotation of labeled documents produced by the LLM pipeline. Performed by *expert raters*.

5. LLM Annotations

To assess the efficiency gains from LLM annotation in medication extraction, we devised an LLM labeling pipeline that consists of three key components: (1) an LLM, (2) a task-specific prompt with an input chunk of text, and (3) a post-processing *Resolver Module* that converts the generative outputs into NER-RE structured objects.

The steps for the LLM medication-extraction annotation pipeline are:

1. Input document is segmented into text chunks, which are delineated by textual breaks such as new lines and end-of-sentence markers.

2. Each text chunk is incorporated into a task-specific *few-shot prompt* designed for medication extraction.

3. The LLM performs inference on this prompt, generating textual outputs.

4. Generative textual outputs are converted into structured medication-extraction objects by the resolver module. Each object encapsulates the relevant medication fields along with their associated RE information.

5. Optionally, human reviewers can refine these annotations to enhance their quality.

Figure 3 provides an illustration of the pipeline.

PaLM 2 Language Model. To construct the LLM medical-extraction annotation pipeline, we utilized Google's latest-generation language model, PaLM 2 (Anil et al., 2023). This model has achieved state-of-the-art performance across a broad spectrum of tasks. While employing a fine-tuned version of PaLM 2 for the NER-RE task might yield even better results, our primary objective was to demonstrate a proof-of-concept using an LLM, without explicit tuning.

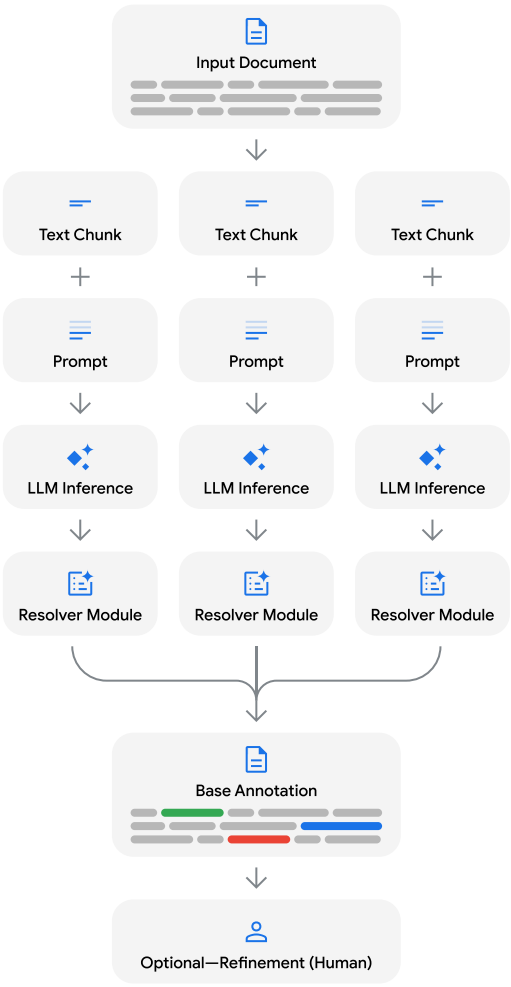

Figure 3: Overview of the LLM medication-extraction annotation pipeline. Our labeling pipeline consists of a task-specific prompt with an input chunk of text, an LLM, and a post-processing Resolver Module that converts the generative outputs into NER-RE structured objects.

5.1. Prompt Structure

To apply a generalized LLM to the specific medication-extraction task, we utilize *in-context* or *few-shot learning* (Brown et al., 2020). This approach leverages prompts containing: contextual information, a description of the task, and one or more *few-shot examples* with input-output pairs. During inference, the input chunk text is integrated into the prompt, as illustrated in Figure 4. (See appendix F for example prompt implementations).

Structure of Prompt

Contextual Information:
"You are a medical assistant..."

Task Description:
"Tag groups of entities related to medications using YAML format. Include drug names, dose, etc..."

Examples:
Question: "Patient is on Prozac 20 mg..."
Answer:

```
entities:
  - group: 1
    medication_name:
      text: Prozac
    ...
```

Question/Answer Pairs Continued...

Question: {input_chunk_text}
Answer:

Figure 4: Prompt structure for LLM annotation: includes contextual information, task description, and curated examples to illustrate task intricacies.

5.2. Medication-Extraction Prompts Schema

To leverage in-context learning for medication extraction, the task should be precisely described and defined within the prompt. This information shapes the generative outputs from the LLM. We evaluated two prompt schemas, and generated annotations of both of which were ultimately ensembled to achieve optimal performance. The first schema, *IOB-Token*, utilizes a modified version of the Inside-Outside-Beginning (IOB) token-tagging technique (Ramshaw and Marcus, 1999), using medication fields as possible classes. The IOB method classifies each token into a specific class. Our modification introduces an additional group tag to retain RE information, as depicted in Figure 5.

The second schema, *Direct Chunk*, directly produces the fields labels and grouping for the input chunk of text which may contain multiple words (e.g., '20 mg'), without requiring intermediate tokens. Figure 6 provides an illustration of this schema.

Once generative outputs are produced, the *Resolver Module* transforms unstructured output data into medication-extraction objects. The object definitions of the output are consistent with the human

IOB-Token Annotation

Chunk Input: "Patient takes Prozac 20 mg and Zoloft 50 mg..."

Token	IOB Tag-Label	Group Tag
'Patient'	O	None
'takes'	O	None
'Prozac'	B-MEDICATION	'entity_1'
'20'	B-DOSE	'entity_1'
'mg'	I-DOSE	'entity_1'
'and'	O	None
'Zoloft'	B-MEDICATION	'entity_2'
'50'	B-DOSE	'entity_2'
'mg'	I-DOSE	'entity_2'

Figure 5: Example of IOB-token annotations for medication information extraction. In the prompt implementation, the tokenized text, IOB tag, and group entity are comma-delimited.

Direct Chunk Annotation

Chunk Input: "Patient takes Prozac 20 mg and Zoloft 50 mg..."

Field Type	Entity Text	Position (start-end)
Group: 1		
MEDICATION:	Prozac	18-24
DOSE:	20 mg	25-30
FREQUENCY:	-	
DURATION:	-	
REASON:	-	
MODE:	-	
Group: 2		
MEDICATION:	Zoloft	35-41
DOSE:	50 mg	42-47
FREQUENCY:	-	
DURATION:	-	
REASON:	-	
MODE:	-	

Figure 6: Example of Direct Chunk annotations, demonstrating the extraction of medication fields from the input by groups. In the prompt implementation, each group is represented in YAML format.

labeling pipeline, facilitating seamless integration between LLM and human annotations. Both the IOB-Token and Direct Chunk schemas have a dedicated resolver module. For *IOB-Token resolver*, lines are parsed using a comma as the delimiter, while the *Direct Chunk resolver* parses output using a Python YAML library. See appendix A.1 for further details.

5.3. Prompt Engineering & Ensembling

With an established prompt structure and schema in place, we performed iterative experiments to identify the optimal LLM prompts. To identify areas for prompt improvement, we evaluated errors on the 'dev set'. The most significant error patterns were corrected and incorporated as instructions and examples into subsequent prompt versions (See appendix A.2 for further details). We ultimately developed 10 versions of the IOB-Token schema prompt and 5 versions of the Direct Chunk prompt.

We also evaluated other key parameters: Our early observations indicated that a temperature of 0.0 offered the best performance. Max-decoding-steps above 1024 tokens had minimal impact, so this value was fixed. Chunk sizes of 250 and 180 offered the best performance for IOB-Token and Direct Chunk, respectively. See appendix A.3 for further discussion.

We placed a higher priority on annotation recall over precision because our empirical experience suggests that deleting annotations is less time consuming than producing them from scratch. To select for a weighted balance of recall over precision, we used an F2 score. Appendix D.2 quantifies the benefit of prioritizing recall over precision, showing the lower human time cost for deleting a label compared to adding or modifying one.

Lastly, we evaluated the performance of an ensemble that combines generated annotations from the best configurations from both prompt schemas. Pseudo-code for the ensemble procedure is presented in appendix A.4. The ensemble had the highest F2 score and was therefore chosen for the LLM Base Annotations on the 'test set'. See appendix A.5 for experiment results of best performing configurations.

6. Results

We focus on three key aspects: time savings, label quality, and the performance differentiation between expert and non-expert raters for the Base Annotations phase. Our main findings are as follows:

- **Time Efficiency:** Using the LLM pipeline to replace the human rater in the initial Base Annotations phase reduces human annotation total time by 58% on average.
- **Label Quality:** Comparative evaluations between human raters and the LLM pipeline show that the label quality at the Base Annotations phase is comparable. Additionally, the expert Refinement phase achieves the same quality level regardless of the source of Base Annotations — whether from human raters or the LLM pipeline.
- **Expert vs. Non-Expert Performance:** Subgroup analysis focusing on *expert raters* demonstrates that LLMs improve Base Annotations efficiency by 26% on average, underlining the utility of LLMs even within exceptionally skilled raters. Furthermore, the benefit of utilizing LLMs is even more pronounced compared to *non-expert raters*.

In addition, to substantiate these findings, we have employed statistical tests for the main experiments. The details of which can be found in Table 4.

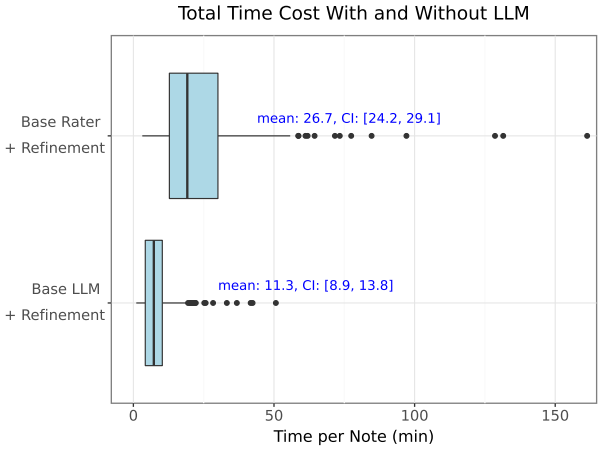

Figure 7: Total human time cost for producing refined labels with and without LLM. Using LLMs reduces 58% of the annotation time (11.3 vs. 26.7).

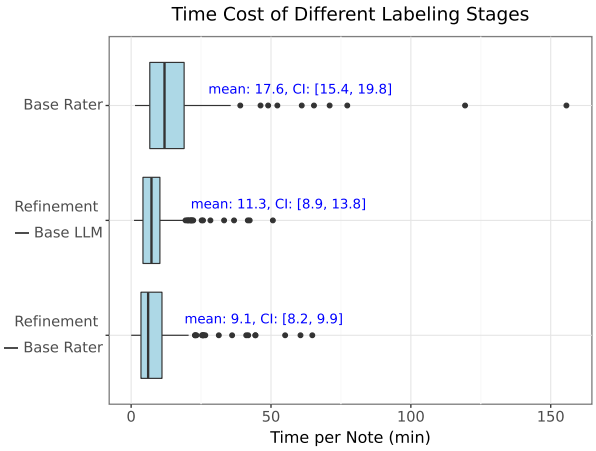

Figure 8: Human time cost at the two labeling phases- Base Annotations and Refinement. Refining annotations reduces time costs by 42% on average compared to starting from scratch.

6.1. Efficiency Metrics.

Figure 7 and Table 2 illustrate that the integration of LLMs leads to a substantial decrease in the human time cost for the production of *expert-level* labels. Using LLMs, the mean time required to produce refined labels for a document decreased by 58%, from 26.67 minutes to 11.32 minutes ($p < 1 \times 10^{-18}$). Figure 8 provides a more granular view of the time costs of each annotation phase. The *Base Rater* has the highest time cost, with a mean of 17.6 minutes per document, compared to the *Refinement over Base LLM* and *Refinement over Base Rater*, which have means of 11.3 and 9.1 minutes per document, respectively.

Details on LLM annotation time in appendix B.

6.2. Quality Metrics.

To evaluate label quality, we utilize phrase-level evaluation [3] based on vertical and horizontal metrics defined by (Uzuner et al., 2010). *Phrase-level* refers to the entire text span of a field, as opposed to *Token-level* which refers to each token in the span, e.g. '20 mg' vs. ['20', 'mg']. *Vertical metrics* evaluate NER with respect to fields, while *horizontal metrics* also evaluate RE between fields and the medication entry. Pseudo-code for these metrics is presented in appendix C. Table 2 outlines the label quality across the two phases of labeling, with and without the LLM.

3. Token-level evaluation shows similar results and is presented in Appendix E.2.

Label Type	Vertical			Horizontal			Time Cost	
	Recall	Precision	F1	Recall	Precision	F1	Mean	Median
Base Rater	0.789	0.893	0.838	0.734	0.821	0.775	17.60	11.93
Base LLM	0.850	0.762	0.804	0.810	0.703	0.752	n/a	n/a
Base Rater + Refinement	0.912	0.907	0.910	0.887	0.879	0.883	26.67	19.18
Base LLM + Refinement	0.921	0.893	0.907	0.892	0.860	0.876	11.32	7.27

Table 2: Labeling Quality and Time metrics for the entire Test Set (n=216 documents). Quality metrics are calculated at the phrase-level. *Time Cost* refers to the total time expended by raters, measured in minutes per document. Refinement methods demonstrate significantly higher quality, with LLM + Refinement method reducing annotation time by 57.6%, while compromising only 1% in quality metrics.

Quality of Base Annotations Phase. Comparing the Quality of *Base Rater Labels* and *Base LLM Labels* in Table 2 indicates that the Base rater outperforms in F1 score (0.838 vs. 0.804 in vertical F1 score, and 0.775 vs. 0.752 in horizontal F1 score), due to the dominance in precision difference, while the *Base LLM* obtained a higher recall rates, as we intended (see Sec 5.3).

Quality of Refinement Annotations Phase. Upon transitioning to the Refinement phase, a striking improvement in label quality is evident from the results presented in Table 2. Both *Base LLM + Refinement* and *Base Rater + Refinement* exhibit substantial advancements in their F1 scores. Specifically, the F1 score for *Base LLM + Refinement* reaches 0.907 in vertical metrics and 0.876 in horizontal metrics. These scores closely mirror the high quality of the *Base Rater + Refinement*, which achieves F1 scores of 0.910 and 0.883 for vertical and horizontal metrics, respectively. An analysis of the precision and recall metrics demonstrates similar alignment.

Results suggest a noteworthy convergence in label quality between the *Base LLM + Refinement* Labels and *Base Rater + Refinement* Labels. The findings also highlight the value of expert rater involvement in elevating both LLM-generated and average-rater annotations to produce high-quality labels.

6.3. Impact of Rater Expertise

To further evaluate the quality and efficiency metrics between expert raters and non-expert raters, we evaluated two subgroups of test documents, differing based on the rater's skill level for the Base Annotation phase.

Table 3 shows that the time cost for producing Base Annotations is similar regardless of the level of expertise. The mean time is 16.17 minutes for experts and 18.67 minutes for non-experts.[4]

Evaluating quality, Table 3 also underscores the strong relationship between rater expertise and labeling quality. *Base expert raters* labels achieve high F1 scores—0.900 in vertical and 0.871 in horizontal metrics—signifying that these labels seldom require refinement. However, the efficiency aspect should not be overlooked. The table demonstrates that even *expert raters* realize significant time savings when assisted by LLMs. Specifically, the average time for labeling reduces from 16.17 minutes with *Base Expert Rater Labels* to 11.97 minutes with *Base LLM + Refinement Labels*, improving efficiency by 26% ($p < 0.01$). Lastly, for the *non-expert raters* subgroup, the benefits of using LLMs are even more pronounced, as can be seen in Table 3.

7. Discussion

Our study demonstrates that LLMs can significantly accelerate the process of medication information extraction, achieving baseline accuracy comparable to that of a trained medical NLP annotator, with superior recall at the expense of precision. Remarkably, the results were achieved solely through prompt engineering, without direct model parameter tuning. Furthermore, we show that when integrated into a human-in-the-loop process, LLMs can facilitate the generation of expert-level annotations while saving considerable human time. This is particularly noteworthy given the traditionally high costs and time investments required for generating high-quality medical NLP labels through human annotation.

4. A Figure displaying the time cost for experts versus non-experts, as well as the time cost per rater, are shown in appendix E.3.

Label Type	Vertical			Horizontal			Time Cost	
	Recall	Precision	F1	Recall	Precision	F1	Mean	Median
Base Expert Rater	0.908	0.893	0.900	0.882	0.859	0.871	16.17	11.13
Base Expert Rater + Refinement	0.923	0.900	0.911	0.898	0.873	0.886	22.25	15.22
Base LLM	0.847	0.762	0.802	0.815	0.706	0.757	n/a	n/a
Base LLM + Refinement	0.914	0.885	0.899	0.896	0.858	0.876	11.97	7.55
Base Non-Expert Rater	0.712	0.892	0.792	0.637	0.789	0.705	18.67	12.36
Base Non-Expert Rater + Refinement	0.905	0.913	0.909	0.880	0.882	0.881	30.41	21.06
Base LLM	0.853	0.763	0.805	0.806	0.700	0.750	n/a	n/a
Base LLM + Refinement	0.925	0.899	0.912	0.890	0.861	0.875	10.83	7.10

Table 3: Comparison of Quality and Time metrics in Base Annotations by Experts and Non-Experts. **Top:** Test documents with Base Annotations performed by experts (n=93). **Bottom:** Test documents with Base Annotations performed by non-experts (n=123). Base LLM and Base LLM + Refinement are included for completeness. Note: Refinements are exclusively expert-performed. *Time Cost* denotes the total time spent by raters, in minutes per document. Notably, the quality of Base annotations by experts without refinement is comparable to that of Base LLM + Refinement, but requires 26% more time.

Label Comparison	Vertical			Horizontal			Time Cost
	Δ_{Recall}	$\Delta_{\text{Precision}}$	Δ_{F1}	Δ_{Recall}	$\Delta_{\text{Precision}}$	Δ_{F1}	Δ_{Median}
Base LLM+Refin. − Base Rater+Refin.	0.009^{**}	-0.014^{**}	-0.003	0.006	-0.018^{**}	-0.007^{*}	-11.90^{**}
Base LLM+Refin. − Base Expert Rater	0.006^{**}	-0.008^{**}	-0.001	0.013^{**}	-0.001	0.006^{*}	-3.58^{**}
Base LLM − Base Non-expert Rater	0.141^{**}	-0.129^{**}	0.013^{**}	0.169^{**}	-0.089^{**}	0.045^{**}	-12.36^{**}

Table 4: Comparing Quality and Time metrics across labeling methods on the Test Set. 'Refin.' denotes Refinement. Differences (Δ) are calculated by subtracting the second method's values from the first. ** and * signify statistical significance at $p < 0.01$ and $p < 0.05$. See Appendix D for analysis details.

In future work, we plan to explore fine-tuning LLMs for specific tasks to improve performance, and the integration of constrained decoding to ensure outputs adhere to predefined schemas. As the field continues to advance rapidly, we expect to see further improvements in performance and utility.

In addition to speeding up the annotation process, our study reveals the importance of expert-level human annotators in achieving high-quality results. We quantified the time-efficiency gains from using LLMs in the annotation pipeline and found that optimizing for recall is more time-efficient as deleting annotations is easier than adding new annotations. Although our study focused on medical information extraction, the flexibility of LLMs suggests their applicability across various information extraction tasks.

However, our study is not without its limitations. The focus on a single domain, namely medication extraction, may limit the generalizability of our findings. As more diverse and validated medical NLP datasets become available, future studies could broaden this scope. While our study concentrated on the feasibility of LLM-based labeling, a key area for further investigation involves directly comparing an LLM annotation pipeline with other ML methods, including active learning. Lastly, our cost analysis solely considers a basic metric of human time cost. It is important to differentiate between expert-level and average raters in future assessments, and to explore a cost-aware approach.

We believe that integrating LLMs into data labeling workflows has potential to change the field, especially in areas where high-quality annotations are required but resources are limited. As LLMs continue to evolve, they are poised to make the challenge of data acquisition increasingly manageable.

Acknowledgments

The authors extend their sincere gratitude to Dem Gerolemou for substantial contributions to the visuals, and to Wei-Hung Weng for offering insightful feedback during the manuscript's review process.

References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. Large language models are few-shot clinical information extractors. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1998–2022, 2022.

Victor Akinwande, Yiding Jiang, Dylan Sam, and J. Zico Kolter. Understanding prompt engineering may not require rethinking generalization, 2023.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Guthrie S Birkhead, Michael Klompas, and Nirav R Shah. Uses of electronic health records for public health surveillance to advance public health. *Annual review of public health*, 36: 345–359, 2015.

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

Sabyasachi Dash, Sushil Kumar Shakyawar, Mohit Sharma, and Sandeep Kaushik. Big data in healthcare: management, analysis and future prospects. *Journal of Big Data*, 6(1):54, Jun 2019. ISSN 2196-1115. doi: 10.1186/s40537-019-0217-0. URL https://doi.org/10.1186/s40537-019-0217-0.

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

Bosheng Ding, Chengwei Qin, Linlin Liu, Lidong Bing, Shafiq Joty, and Boyang Li. Is gpt-3 a good data annotator? *arXiv preprint arXiv:2212.10450*, 2022.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model, 2023.

Amir Feder, Danny Vainstein, Roni Rosenfeld, Tzvika Hartman, Avinatan Hassidim, and Yossi Matias. Active deep learning to detect demographic traits in free-form clinical notes. *Journal of Biomedical Informatics*, 107:103436, 2020. ISSN 1532-0464. doi: https://doi.org/10.1016/j.jbi.2020.103436. URL https://www.sciencedirect.com/science/article/pii/S1532046420300642.

Amir Feder, Itay Laish, Shashank Agarwal, Uri Lerner, Avel Atias, Cathy Cheung, Peter Clardy, Alon Peled-Cohen, Rachana Fellinger, Hengrui Liu, Lan Huong Nguyen, Birju Patel, Natan Potikha, Amir Taubenfeld, Liwen Xu, Seung Doo Yang, Ayelet Benjamini, and Avinatan Hassidim. Building a clinically-focused problem list from medical notes. In *Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis (LOUHI)*, pages 60–68, Abu Dhabi, United Arab Emirates (Hybrid), December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.louhi-1.8. URL https://aclanthology.org/2022.louhi-1.8.

Elizabeth Ford, John A Carroll, Helen E Smith, Donia Scott, and Jackie A Cassell. Extracting information from the text of electronic medical records to improve case detection: a systematic review. *Journal of the American Medical Informatics Association*, 23(5):1007–1015, 2016.

Akshay Goel, Almog Gueta, Omry Gilon, Sofia Erell, and Amir Feder. Medication extraction labels for mimic-iv-note clinical database. PhysioNet, 2023. URL https://doi.org/10.13026/6d3r-w470.

A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000 (June 13). Circulation Electronic Pages: http://circ.ahajournals.org/content/101/23/e215.ful PMID:1085218; doi: 10.1161/01.CIR.101.23.e215.

Kai Hakala and Sampo Pyysalo. Biomedical named entity recognition with multilingual BERT. In *Proceedings of the 5th Workshop on BioNLP Open Shared Tasks*, pages 56–61, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5709. URL https://aclanthology.org/D19-5709.

Peter B Jensen, Lars J Jensen, and Søren Brunak. Mining electronic health records: towards better research applications and clinical care. *Nature Reviews Genetics*, 13(6):395–405, 2012.

Alistair Johnson, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. Mimic-iv-note: Deidentified free-text clinical notes, 2023.

Mahnoosh Kholghi, Laurianne Sitbon, Guido Zuccon, and Anthony Nguyen. Active learning: a step towards automating medical concept extraction. *Journal of the American Medical Informatics Association*, 23(2):289–296, 08 2015. ISSN 1067-5027. doi: 10.1093/jamia/ocv069. URL https://doi.org/10.1093/jamia/ocv069.

Mohamed Yassine Landolsi, Lobna Hlaoua, and Lotfi Romdhane. Information extraction from electronic medical documents: state of the art and future research directions. *Knowledge and Information Systems*, 65, 11 2022. doi: 10.1007/s10115-022-01779-1.

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

Alisa Liu, Swabha Swayamdipta, Noah A Smith, and Yejin Choi. Wanli: Worker and ai collaboration for natural language inference dataset creation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6826–6847, 2022.

Xin Liu, Daniel McDuff, Geza Kovacs, Isaac Galatzer-Levy, Jacob Sunshine, Jiening Zhan, Ming-Zher Poh, Shun Liao, Paolo Di Achille, and Shwetak Patel. Large language models are few-shot health learners. *arXiv preprint arXiv:2305.15525*, 2023.

Yifan Peng, Shankai Yan, and Zhiyong Lu. Transfer learning in biomedical natural language processing: an evaluation of bert and elmo on ten benchmarking datasets. *arXiv preprint arXiv:1906.05474*, 2019.

Lance A Ramshaw and Mitchell P Marcus. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer, 1999.

Laria Reynolds and Kyle McDonell. Prompt programming for large language models: Beyond the few-shot paradigm, 2021.

Guergana K Savova, James J Masanz, Philip V Ogren, Jiaping Zheng, Sunghwan Sohn, Karin C Kipper-Schuler, and Christopher G Chute. Mayo clinical text analysis and knowledge extraction system (ctakes): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5):507–513, 2010.

Yuqi Si, Jingqi Wang, Hua Xu, and Kirk Roberts. Enhancing clinical concept extraction with contextual embeddings. *Journal of the American Medical Informatics Association*, 26(11):1297–1304, 2019.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Nathaneal Scharli, Aakanksha Chowdhery, Philip Mansfield, Blaise Aguera y Arcas, Dale Webster, Greg S. Corrado, Yossi Matias, Katherine Chou, Juraj Gottweis, Nenad Tomasev, Yun Liu, Alvin Rajkomar, Joelle Barral, Christopher Semturs, Alan Karthikesalingam, and Vivek Natarajan. Large language models encode clinical knowledge, 2022.

Özlem Uzuner. Recognizing obesity and comorbidities in sparse data. *Journal of the American Medical Informatics Association*, 16(4):561–570, 2009.

Özlem Uzuner, Imre Solti, and Eithon Cadag. Extracting medication information from clinical text. *Journal of the American Medical Informatics Association*, 17(5):514–518, 2010.

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

Qiang Wei, Yukun Chen, Mandana Salimi, Joshua C Denny, Qiaozhu Mei, Thomas A Lasko, Qingxia Chen, Stephen Wu, Amy Franklin, Trevor Cohen, and Hua Xu. Cost-aware active learning for named entity recognition in clinical text. *Journal of the American Medical Informatics Association*, 26(11):1314–1322, 07 2019. ISSN 1527-974X. doi: 10.1093/jamia/ocz102. URL https://doi.org/10.1093/jamia/ocz102.

Vikas Yadav and Steven Bethard. A survey on recent advances in named entity recognition from deep learning models. *arXiv preprint arXiv:1910.11470*, 2019.

Alexander Yeh. More accurate tests for the statistical significance of result differences. *arXiv preprint cs/0008005*, 2000.

Fan Zhang, Itay Laish, Ayelet Benjamini, and Amir Feder. Section classification in clinical notes with multi-task transformers. In *Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis (LOUHI)*, pages 54–59, 2022.

Wenxuan Zhou, Sheng Zhang, Yu Gu, Muhao Chen, and Hoifung Poon. Universalner: Targeted distillation from large language models for open named entity recognition, 2023a.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers, 2023b.

Henghui Zhu, Ioannis Ch Paschalidis, and Amir Tahmasebi. Clinical concept extraction with contextual word embedding. *arXiv preprint arXiv:1810.10566*, 2018.

Appendix A. LLM Labeling Pipeline Implementation Details

Here we describe implementation details for our LLM pipepline, including our resolver module, configurations and hyper-parameters and error identification and improvement.

A.1. Resolver Module: Parsing Generative Outputs from Unconstrained LLMs

Parsing generative outputs from an unconstrained LLM can present various edge cases that are specific to the annotation schema. In the case of the IOB-Token schema, the outputs from the LLMs were formatted so that each token was separated by a new line. As such, the outputs were processed one line at a time. The individual components of each token were then parsed using a comma-delimited structure and regular expressions. Occasionally, the LLM produced outputs that did not conform to the expected structure, such as an extra comma or a missing new line. When the output structure could not be resolved, it was discarded and logged for further analysis. The 'Group Tag' information is employed to group fields into medication entries.

Similarly, for the Direct-Chunk implementation, the LLM output is produced in YAML format and delimited by backticks (‘‘‘). This output is isolated and then sent to the standard Python YAML parsing library to generate a dictionary representation of the output. If an edge case leads to a YAML parsing error, the output is logged for further analysis. Once the output has been parsed into a dictionary, each entity is mapped back to the original text. Since the fields for a specific medication are all contained within a single YAML output, individual grouping of fields is unnecessary.

During the phase where outputs are matched to the original text, additional considerations come into play, such as applying a "fuzzy match." This allows for some minor differences in characters and helps to improve entity recall (i.e., matching a correctly spelled medication generated by the LLM with a misspelled medication in the raw text). When the outputs could not be resolved to fit the target structure or mapped back to the original text, they were logged for analysis but ultimately not used as labels. This can contribute to potential recall errors.

A.2. Error Identification and Prompt Improvement

Our experiments with the LLM pipeline revealed various types of errors, which have been categorized and addressed to improve in-context learning. Errors spanned from missing annotations to spurious annotations and field-type confusions.

Missing Annotations: The model had difficulty in identifying less frequent or differently formatted fields. We incorporated additional examples to address these omissions, though it led to a trade-off in overall precision.

Spurious Annotations: The model sometimes incorrectly annotated entities that did not belong to medication information. To mitigate this, we added negative examples in the prompt to refine its identification capabilities.

Confusion between Fields: Occasionally, the model confused similar field types. This was particularly evident in distinguishing between 'mode' and 'frequency'. Adjustments were made in the prompt to improve the model's accuracy in this aspect.

Nuanced Errors: Many errors exhibited a reasonable basis, pointing to the complexities in the common definitions used in the medical domain. For example, the model classifying 'oxygen' as a medication could be attributed to the overlap between the common definitions of chemical medications and therapeutic treatments like oxygen therapy. Examples of these nuanced errors were also incorporated into the prompt.

A.3. Configurations and Hyper-Parameters

In addition to experimenting with different prompts, we also examined various hyper-parameters, including chunk size, maximum decoding steps, and temperature. We tested multiple values for each parameter and selected those that performed best. It is worth noting that we conducted a partial grid search and drew our conclusions based on these results. To determine the best configuration we used a balanced F2 score defined as $F_2 = \frac{5}{\frac{4}{\text{recall}} + \frac{1}{\text{precision}}}$. This provided a higher emphasis on recall, which we believed would be more important for the annotation task. For all best-performing models, 1024 maximum decoding steps yielded the best performance. The optimal chunk size varied between YAML-based and IOB-based models, with chunks of 180 and 250 characters performing best, respectively. Different optimal parameters between the two schema were expected

given their different approaches. The YAML-based models directly produce labels and groups for the input chunk which may benefit from shorter chunks, whereas IOB-based models have an intermediate step of outputting per token before grouping them. For all selected models, a temperature of 0.0 proved optimal, supporting that a deterministic setting is optimal for an information extraction task.

A.4. Pseudo-code for the Ensemble Procedure to Combine Annotations

Figure 9 presents pseudo-code for the ensemble procedure that combines generated annotations from both prompt schemas. As described in Section 5.3 the ensemble had the highest F2 score and was therefore chosen for the LLM Base Annotations on the 'test set'.

```
def ensemble_document_annotations(model1_annotations, model2_annotations):
    # Input: Lists of annotations made by each model on the same text document.
    # The annotations for each entry include the annotated fields' types and spans.

    # The function first takes annotations from model1. Then it tries to intersect
    # annotations from model2 - if partially intersected, takes the union.

    # Output: A single list of ensembled annotations.

    all_annotations = []
    aligned_entries = align entries of (model1_annotations, model2_annotations)
                      using partial intersection of the span of 'name' field
    for entry in aligned_entries:
       new_entry = Annotation()
       for field in entry:
           if field['model1']:
              if not field['model2']:
                 new_entry.append(field['model1'])
              else:
                 if field['model1'] ∩ field['model2']: # spans interaction
                    new_entry.append(field['model1'] ∪ field['model2']) # spans union
                 else:
                    new_entry.append(field['model1'])
           elif field['model2']:
               new_entry.append(field['model2'])
        if new_entry:
           all_annotations.append(new_entry)
    return all_annotations
```

Figure 9: Pseudo-code for the ensemble procedure.

A.5. Choosing the final Model for LLM Base Annotations

Table 5 shows the performance of best performing models on the development set. We can see that the ensemble of the two top models outperform the f2-score, meaning the combination of the models balances between increasing metrics without damaging others. Therefore, we chose the ensemble model for the LLM Base Annotations.

Appendix B. Inference Time of LLM Annotations

As the inference process can be scaled across all documents simultaneously with the appropriate infrastructure, comparing LLM inference time to human annotation time directly is not applicable. Nonetheless, performing inference using PaLM 2 on a TPU 8x8 pod, on each document separately, on a random subset of 15% of the documents, revealed an average inference time of 1.5 minutes per 1000 characters, with a standard deviation of 0.77 minutes. This translates to an average of 9.49 minutes per document, with a standard deviation of 5.55 minutes.

Appendix C. Pseudo-code for Vertical and Horizontal Metrics

Figures 10 and 11 present pseudo-code for the Vertical metrics and Horizontal metrics, respectively. Both are based on the metrics defined by (Uzuner et al., 2010). As described in Section 6.2, *Vertical metrics* evaluate NER with respect to fields, while *horizontal metrics* also evaluate RE between fields and the medication entry.

```
def vertical_metric_phrase_level(documents):
   # Input: A list of documents, where each document contains
   # the medication entries of the model and the ground truth (GT)

   # Output: Recall, Precision, F1 for all documents.

   total_tp, total_fn, total_fp = 0, 0, 0

   for doc in documents:
      model_set = set()
      gt_set = set()
      for entry in doc.model_entries:
         for field in entry.fields:
            # field e.g. (Dose, start_pos, end_pos)
            model_set.add(field)
      for entry in doc.gt_entries:
         for field in entry.fields:
            gt_set.add(field)

      total_tp += count elements of (gt_set ∩ model_set)
      total_fn += count elements of (gt_set - model_set)
      total_fp += count elements of (model_set - gt_set)

   R = total_tp/ (total_tp + total_fn)
   P = total_tp/ (total_tp + total_fp)
   F1 = 2/ ((1/P) + (1/R))
   return R, P, F1
```

Figure 10: Pseudo-code for Vertical metrics.

Type	Chunk Size	F1 Score	F2 Score	Recall	Precision
IOB Token	250	**82.0**	84.0	85.5	78.7
Direct Chunk	180	80.9	80.8	80.7	**81.0**
Ensemble	-	81.8	**84.9**	**87.1**	77.1

Table 5: Performance of top experiments on development set for both prompt schemas. F2 score was used to select top experiments. The ensemble version, highlighted in gray, was the final version used for LLM base annotations on the test set.

```
def horizontal_metric_phrase_level(documents):
   # Input: A list of documents, where each document contains
   # the medication entries of the model and the ground truth (GT)

   # Output: Recall, Precision, F1 for all documents.

   total_tp, total_fp, total_fn = 0, 0, 0

   for doc in documents:
      model_entries_map = create_empty_dict()
      gt_entries_map = create_empty_dict()

      for entry in doc.model_entries:
         model_entries_map[entry.medication_name_span].append(entry)
      for entry in doc.gt_entries:
         gt_entries_map[entry.medication_name_span].append(entry)

      for medication_name_span in
            set(gt_entries_map.keys + model_entries_map.keys):
         med_gt_entries = gt_entries_map[medication_name_span]
         med_model_entries = model_entries_map[medication_name_span]

         if med_gt_entries.empty:
           for entry in med_model_entries:
              total_fp += count_fields(entry)
         elif med_model_entries.empty:
           for entry in med_gt_entries:
              total_fn += count_fields(entry)
         else:
           for gt_entry in med_gt_entries:
              best_f1 = -1
              for model_entry in med_model_entries:
                 _, _, f1, tp, fp, fn =
                       vertical_metric({Document(model_entries={model_entry},
                                                 gt_entries={gt_entry})})
                 if f1 > best_f1:
                   best_model_entry, best_f1 = model_entry, f1
                   best_metrics = (tp, fp, fn)

              total_tp += best_metrics.tp
              total_fp += best_metrics.fp
              total_fn += best_metrics.fn

   R = total_tp/ (total_tp + total_fn)
   P = total_tp/ (total_tp + total_fp)
   F1 = 2/ ((1/P) + (1/R))
   return R, P, F1
```

Figure 11: Pseudo-code for Horizontal metrics.

Appendix D. More Details on Statistical Analysis

Here we provide more details on the statistical analysis conducted in Section 6.

D.1. Statistical Significance Test

To compare the performance metrics (precision, recall and F1 score) of various labeling methods, we use the approximated randomization test in (Yeh, 2000), which is more powerful and precise than the method implemented in *i2b2*. For two different labeling methods A and B, denote their output sets as O_A and O_B and the performance metrics as f_A and f_B. Then $\Delta = f_A - f_B$ is the observed performance difference, $Q = O_A \cap O_B$ is the set of common outputs shared by both methods, and $P_A = O_A \setminus O_B$ and $P_B = O_B \setminus O_A$ are the sets of unique outputs only generated from either method A or method B. When the two methods are similar (null hypothesis), the outputs in P_A and P_B should be exchangeable. The implementation details of our approximated randomization test are as follows:

1. Gather unique outputs generated from either A or B, i.e., $P = P_A \cup P_B$.
2. Randomly assign each output in P to new sets P'_A or P'_B with equal probability, and create the set of psedo-outputs for A and B
$$O'_A = P'_A \cup Q,\ O'_B = P'_B \cup Q.$$
3. Compute performance metrics on O'_A and O'_B, and calculate their difference Δ'.
4. Repeat Steps 2-3 for $n = 1,000$ times, and generate a sampling distribution of Δ'.

If the observed metric difference Δ is above the 95% percentile of the sampling distribution, we can claim that method A performs significantly better than B with metric difference Δ.

D.2. Time cost and types of refinement

We conducted further analysis to investigate what type of errors in pre-labels / what type of corrections has greatest impact on time cost. Focusing on text spans only, there are three types of corrections a human refiner can do: addition (adding a missing text span), modification (modify an existing text span) and deletion (delete a spurious text span). First, we look into the distribution of these three types of corrections for refinement based on human pre-labels

and refinement based on LLM pre-labels Figures 12, 13, and 14. Refinement on LLM pre-labels usually requires significant more deletions and modifications, while the human + refinement method involves more additions (all p-values < 0.5). By fitting a regression model on time cost using the number of corrections of each type as predictors, the amount of time needed for adding a missing text span and modify an existing text span are similar, while it is much faster to delete a text span; see regression results in Table 6.

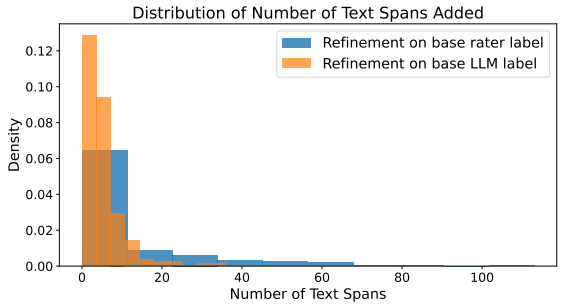

Figure 12: Distribution of additions in the refinement process, LLM vs. rater base generated labels.

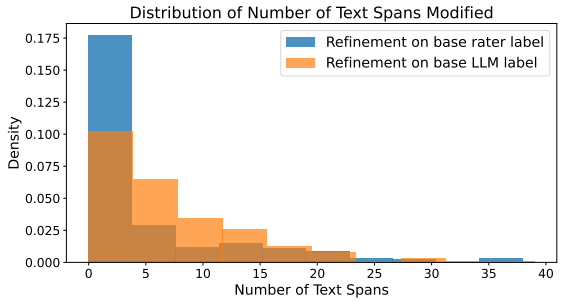

Figure 13: Distribution of modifications in the refinement process, LLM vs. rater base generated labels.

Variable	Coefficient	Std. Error	z score
Intercept	4.327**	0.735	5.888
# Modified X	0.351**	0.076	4.618
# Added	0.288**	0.031	9.158
# Deleted	-0.013	0.235	-0.056

Table 6: Linear mixed effect model regression results: time cost vs. type of corrections. ** denotes $p < 0.01$.

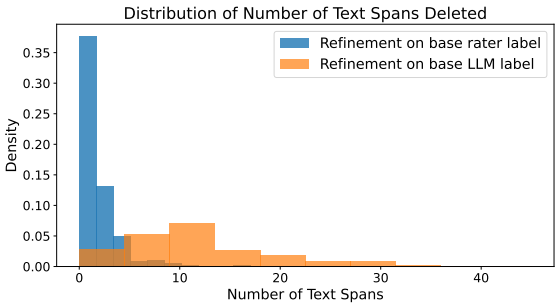

Figure 14: Distribution of deletions in the refinement process, LLM vs. rater base generated labels.

Appendix E. Additional Results

For brevity, we provide here additional results that support the main findings in Section 6.

E.1. Supplemental Results showing performance across each field type

In Table 7, we compare all of the tested approaches in terms of $F1$ scores for each entity type.

E.2. Supplemental Results showing token-level metrics

Table 8 presents token-level labeling quality in terms of horizontal and and vertical metrics on the test set. Results are similar to the ones of phrase-level metrics.

E.3. Time Cost Evaluations

In Figure 15 and Figure 16, we show the time cost for individual raters and based on their level of expertise, respectively.

Method	Medication	Dose	Frequency	Mode	Duration
Base LLM	0.784	0.883	0.787	0.848	0.420
Base Rater	0.776	0.903	0.883	0.907	0.384
Refined Rater	0.908	0.927	0.914	0.952	0.466
Refined LLM	0.907	0.941	0.890	0.947	0.540

Table 7: F1 scores for different types of labels across each medication attribute. The attributes include medication name, dosage, frequency, mode, and duration.

Label Type	Vertical			Horizontal		
	Recall	Precision	F1	Recall	Precision	F1
Base Rater	0.820429	0.924383	0.869309	0.723582	0.756489	0.739669
Base LLM	0.874945	0.769727	0.818970	0.800494	0.678708	0.734587
Refined Rater	0.922830	0.925182	0.924005	0.873006	0.869832	0.871416
Refined LLM	0.939259	0.900770	0.919612	0.886027	0.843396	0.864186

Table 8: *Token-level* labeling Quality and Time Metrics for the Entire Test Set.

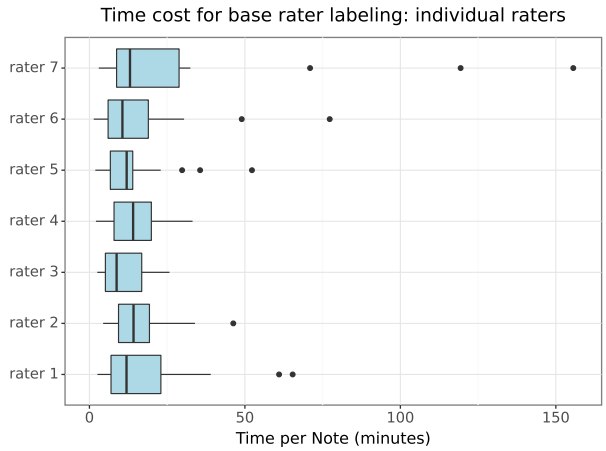

Figure 15: Base labeling time cost of individual raters. Measurements are in minutes per note.

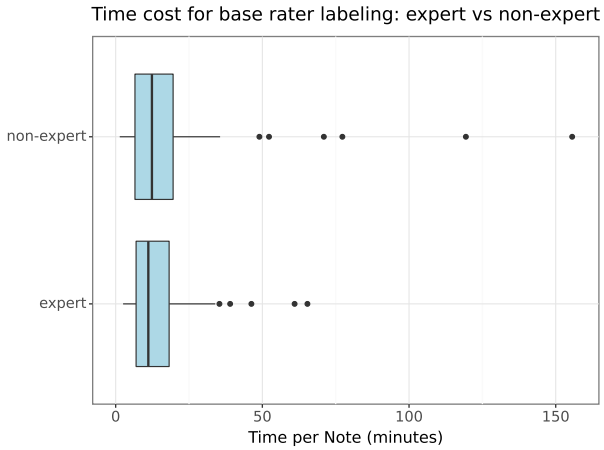

Figure 16: Base labeling time cost: expert raters vs non-expert raters. Measurements are in minutes per note.

Appendix F. Example Prompts

Note: The following examples are synthetic and inspired by, but not directly taken from, the *i2b2* dataset. Only a single question-answer example is provided in each prompt for the sake of brevity.

F.1. IOB-Token Prompt Example

```
description: "You are a helpful medical knowledge assistant for Beginning-Inside-
    Outside (BIO) tagging of medications' information. Your task is to create
    Medical Named Entity Recognition (NER) annotations using the Beginning-Inside-
    Outside (BIO) format. Label the given sentence (question) using BIO tags from
    these classes ONLY: MEDICATION (medication name), DOSE, FREQUENCY, DURATION,
    REASON, MODE. For tokens not belonging to these classes, tag as O-tag. For each
     token you MUST start a new line. Punctuations should be skipped in tokens.
    Each line must include: (1) word or token, (2) BIO tag, and (3) entity number
    for B-tags and I-tags ONLY. Use '<None>' for O-tags. Maintain the correct
    format. Example: 'token', BIO-tag, 'entity_1'."

examples: {
  question: "'CURRENT MEDICATIONS: Ibuprofen as needed, and diclofenac for one
      month as needed, for abdominal discomfort.'"
  answer:
    "'CURRENT', O, '<None>'\n"
    "'MEDICATIONS', O, '<None>'\n"
    "'Ibuprofen', B-MEDICATION, 'entity_1'\n"
    "'as', B-FREQUENCY, 'entity_1'\n"
    "'needed', I-FREQUENCY, 'entity_1'\n"
    "'and', O, '<None>'\n"
    "'diclofenac', B-MEDICATION, 'entity_2'\n"
    "'for', B-DURATION, 'entity_2'\n"
    "'one', I-DURATION, 'entity_2'\n"
    "'month', I-DURATION, 'entity_2'\n"
    "'as', I-DURATION, 'entity_2'\n"
    "'needed', I-DURATION, 'entity_2'\n"
    "'for', O, '<None>'\n"
    "'abdominal', B-REASON, 'entity_1|entity_2'\n"
    "'discomfort', I-REASON, 'entity_1|entity_2'\n"
}
```

F.2. Direct-Chunk Prompt Example

```
description:
  "You are a medical assistant with expertise in document processing.\n"
  "\n"
  "- Your task is to tag groups of entities related to medications. (Some groups
      may have overlap as two medications can be used for the same reason.)\n"
  "- Each medication is a SPECIFIC reference to the drug's name used by the
      patient. This should not contain other data already within a group.\n"
  "- The mode of administration, such as 'oral' or 'intravenous', should be tagged
       SEPARATELY from the medication's name. \n"
  "- Text should contain the EXACT characters in the raw input, including
      punctuation, and typos.\n"
  "- Use YAML as the output format."

examples: {
  question: "'CURRENT MEDICATIONS: Ibuprofen as needed, and diclofenac for one
      month as needed, for abdominal discomfort.'"
  answer:
    "```yaml\n"
    "entities:\n"
    "  - group: 1\n"
    "    MEDICATION: \n"
    "      text: Ibuprofen\n"
    "      start_pos: 20\n"
    "      end_pos: 29\n"
    "    DOSE: \n"
    "      text: ''\n"
    "      start_pos: \n"
    "      end_pos: \n"
    "    FREQUENCY: \n"
    "      text: as needed\n"
    "      start_pos: 30\n"
    "      end_pos: 39\n"
    "    DURATION: \n"
    "      text: ''\n"
    "      start_pos: \n"
    "      end_pos: \n"
    "    REASON: \n"
    "      text: abdominal discomfort\n"
    "      start_pos: 81\n"
    "      end_pos: 100\n"
    "    MODE: \n"
    "      text: ''\n"
    "      start_pos: \n"
    "      end_pos: \n"
    "  - group: 2\n"
    "    MEDICATION: \n"
    "      text: diclofenac\n"
    "      start_pos: 45\n"
    "      end_pos: 55\n"
    "    DOSE: \n"
    "      text: ''\n"
    "      start_pos: \n"
    "      end_pos: \n"
```

```
    "    FREQUENCY: \n"
    "      text: as needed\n"
    "      start_pos: 68\n"
    "      end_pos: 77\n"
    "    DURATION: \n"
    "      text: for one month\n"
    "      start_pos: 56\n"
    "      end_pos: 68\n"
    "    REASON: \n"
    "      text: abdominal discomfort\n"
    "      start_pos: 81\n"
    "      end_pos: 100\n"
    "    MODE: \n"
    "      text: ''\n"
    "      start_pos: \n"
    "      end_pos:\n"
    "```\n"
}
```