Automated clinical coding using off-the-shelf large language models

Joseph S. Boyle[1,2], Antanas Kascenas[1,3], Pat Lok[1,4], Maria Liakata[2,5,6], Alison Q. O'Neil[1,3]

[1]Canon Medical Research Europe, [2]Queen Mary University of London,
[3]University of Edinburgh, [4]Anglia Ruskin University,
[5]The Alan Turing Institute, [6]University of Warwick

joseph.boyle@mre.medical.canon

Abstract

The task of assigning diagnostic ICD codes to patient hospital admissions is typically performed by expert human coders. Efforts towards automated ICD coding are dominated by supervised deep learning models. However, difficulties in learning to predict the large number of rare codes remain a barrier to adoption in clinical practice. In this work, we leverage off-the-shelf pre-trained generative large language models (LLMs) to develop a practical solution that is suitable for zero-shot and few-shot code assignment, with no need for further task-specific training. Unsupervised pre-training alone does not guarantee precise knowledge of the ICD ontology and specialist clinical coding task, therefore we frame the task as information extraction, providing a description of each coded concept and asking the model to retrieve related mentions. For efficiency, rather than iterating over all codes, we leverage the hierarchical nature of the ICD ontology to sparsely search for relevant codes. We validate our method using Llama-2, GPT-3.5 and GPT-4 on the CodiEsp dataset of ICD-coded clinical case documents. Our tree-search method achieves state-of-the-art performance on rarer classes, achieving the best macro-F1 of 0.225, whilst achieving slightly lower micro-F1 of 0.157, compared to 0.216 and 0.219 respectively from PLM-ICD. To the best of our knowledge, this is the first method for automated clinical coding requiring no task-specific learning.

1 Introduction

International Classification of Disease (ICD) coding is the task of assigning ICD codes to episodes of patient care e.g. hospital stays. Assigned codes may be used for billing, audit, resource management, epidemiological study, measurement of treatment effectiveness, and other purposes. Coding is usually a manual process, performed by specialists via inspection of a patient's medical documentation. This is time-consuming and error-prone; with one study estimating the costs related to medical coding to be billions of dollars per year in the US alone [6]. Automation of clinical coding has been pursued since the late 1990s [3], with deep learning techniques currently dominating [4].

A challenge for automation is the large number of codes that must be predicted. The ICD-10-CM ontology contains 96,000 distinct codes, of which 73,000 are assignable [13]. The remaining 23,000 codes are higher-level parent codes corresponding to broader categories of codes. Many assignable codes are rare, and there may be few or no examples in any given training set. Moreover, there is considerable conceptual overlap between ICD codes. This poses difficulties for supervised learning, which relies on large volumes of labelled training data to learn distinct representations for closely related concepts. Compared to other text classification tasks, ICD coding is a challenging

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

problem; quantitatively, the state-of-the-art achieves a micro-F1 of 0.585 and macro-F1 of 0.211 on the MIMIC-IV ICD coding benchmark [4].

In this paper, we explore an alternative approach using off-the-shelf generative LLMs which have been trained using self-supervised learning on trillions of tokens [2, 12]. Recent models have proven powerful for medical tasks such as question answering, summarisation and clinical information retrieval [1], demonstrating good performance on clinical text and notably achieving an 86.5% (pass) score on the MedQA dataset of questions from the US Medical License Examination [10]. We consider how to leverage generative LLMs to perform the task of ICD coding even in the absence of labelled training examples i.e. with no task-specific training. Leveraging the hierarchical nature of the ICD ontology, we propose a novel LLM-based solution that formulates the task as a dynamic sequence of searches for clinical entities, with the sequence corresponding to paths followed through the tree to all relevant assignable "leaf" codes. In this framework, the LLM is prompted to assess the relevance of each branch of the tree based on its text description. Our contributions are as follows:

1. We create the first method for ICD coding requiring no task-specific training or fine-tuning.
2. We demonstrate that LLMs have some out-of-the-box ICD coding abilities.
3. We propose a method that avoids reliance on model knowledge of the target coding ontology via the injection of information into the LLM prompt and the application of a novel search strategy conceptually similar to a multi-label decision tree. We show empirically that this tree-search strategy improves model performance on rare codes.

2 Related Work

Models building on pre-trained encoder transformers such as BERT currently lead performance benchmarks for ICD coding [4, 5]. However, prediction for codes with few or no training examples remains difficult. Song et al. proposed a novel GAN-based feature generation approach for zero-shot labeling by exploiting the ontological structure of ICD to extrapolate features learned from seen sibling and parent codes [11]. Lu et al. proposed aggregating multiple label graphs (the original code ontology, the semantic similarity graph computed from their descriptions, and a label co-occurrence frequency matrix) as a per-label pre-processing step [7]. Recently, Yang et al. conducted a study into few-shot ICD coding by reformulating the classification problem as masked language modelling one (a 'cloze task') such that the model learns to replace the masked tokens in the input prompt with binary predictions [14]. Each of these approaches require labelled data for model training. In contrast, we seek to develop zero-shot learning methods for this task.

3 Data

The open access MIMIC dataset is the most commonly used benchmark for this task [6], however the conditions of use prohibit sharing of data with third parties (such as OpenAI). Therefore, we chose to use **CodiEsp**, a publicly available dataset which formed the basis of the *eHealth CLEF 2020 Multilingual Information Extraction Shared Task*, a competition for automated clinical coding [8]. In this competition, 1000 span-level expert-annotated case notes were released in Spanish, alongside machine-translated English versions. We evaluate on the competition test set but abstract the span-level labels into document-level labels as would be available in real-world clinically coded data. This comprises 250 case note documents from 250 unique patients, covering 1767 distinct ICD-10 codes (2.4% of the ICD-10-CM codeset).

Inspection of the translated documents revealed errors such as failure to translate Spanish terms for drugs. Since translation errors could hamper diagnostic coding performance, we re-translated the documents using GPT-3.5, which reduced errors and yielded modest performance improvements for all models during experimentation. We henceforth refer to this as the 'CodiEsp-English' dataset. Lastly, we remove the small number of ground truth labels not assignable in the ICD-10-CM.

4 Method

We describe our methods and their implementation below.

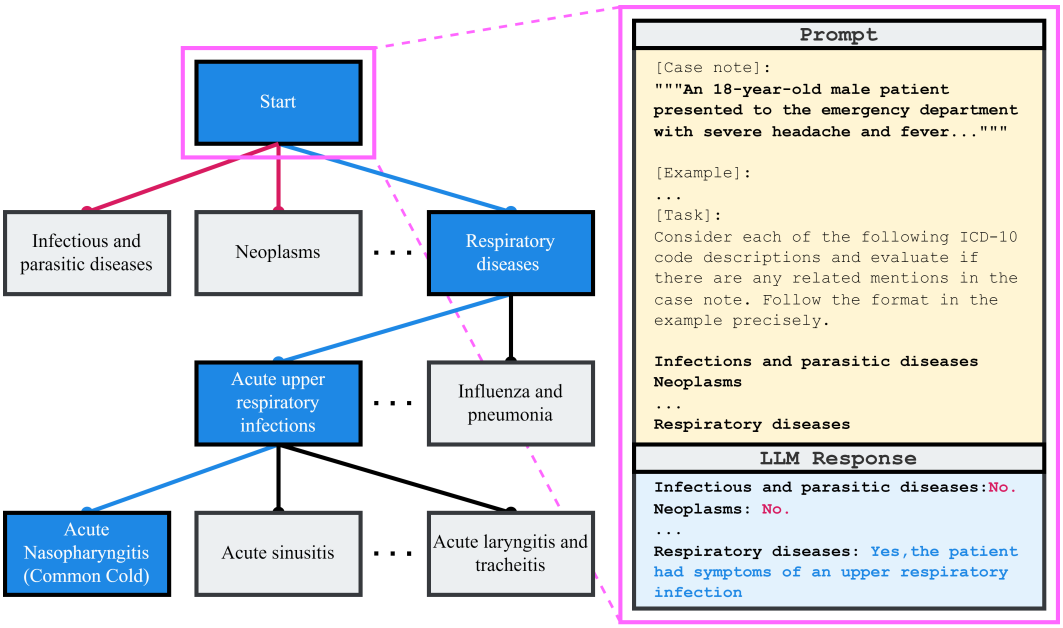

Figure 1: Illustration of our LLM-guided tree-search method, as applied to assign an example code **"Acute nasopharyngitis (Common Cold)"** to CodiEsp document `S0212-71992006000100006-1`, using GPT-4. Left: ICD-10 diagnostic tree, showing the codes traversed to reach the correct assignable "leaf" code. Right: Abridged prompt containing the case note; we colour the LLM model's response with **blue** and **pink** indicating the positively and negatively predicted codes, respectively.

Information retrieval task framing: We hypothesised that LLMs might be capable ICD coders 'out-of-the-box'. Whilst the prompt "`You are a clinical coder, consider the case note and assign the appropriate ICD codes`" elicits the correct behaviour from GPT-4, the responses contained inaccuracies such as mismatched codes and descriptions (see full prompt in Appendix A.1). For instance, the model assigned the code: "C63.2 - Malignant neoplasm of **left testis**' where the true description for the code "C63.2" is 'Malignant neoplasm of **scrotum**".

We therefore reframe the problem as information retrieval and prompt the LLM to retrieve 'mentions' of candidate codes from the case note. Taking inspiration from Yang et al. [14], we include the descriptions in the prompt, as shown in Figure 1. If a relevant mention is found, the code is predicted.

LLM guided tree-search: Determining which ICD code descriptions to employ in the information retrieval prompt is non-trivial as the ICD includes thousands of codes. The ICD diagnostic ontology is a tree, with 'is a' (hyponymic) semantics relating child and parent codes. For instance, the child code *Acute Nasopharyngitis* is an *Upper Respiratory Infection* (parent code). We exploit the tree-like structure of the ontology to efficiently search for relevant ICD codes, using the LLM to decide at each decision point which paths to explore. Starting at the root of the tree, the model selects relevant chapters to explore, as shown in the prompt-response pair in Figure 1. This is performed recursively until the set of candidate paths to explore is exhausted, at which point the set of relevant labels (leaf codes) is returned. For instance the search path shown in Figure 1 would add the label 'Acute Nasopharyngitis' to the set of predicted labels. The exact algorithm is described in Appendix A.3.

For a relational model of the ICD diagnostic tree, we used the open-source `simple_icd_10_cm` library[1]. The LLMs used to guide the tree-search are Llama-2 (70B-chat), GPT-3.5, and GPT-4 [2, 12]. Llama-2 was accessed via the hosted DeepInfra service [2], and GPT-3.5/4 via OpenAI's API. The `06-13` revisions of both GPT-3.5 and GPT-4 were used. To get the most deterministic model outputs possible, the `temperature` parameter was set to it's minimum value; 0 and 0.001 for GPT and Llama respectively (smaller values did not work for Llama).

[1] `https://github.com/StefanoTrv/simple_icd_10_CM`
[2] `https://deepinfra.com/`

Output parsing: Resolving the LLM generated text into per-code predictions is performed by processing the text as a set of lines. These lines are greedily matched, starting with the longest code description in the current prompt. This is relevant as in some instances there are code descriptions which are substrings of another code descriptions. For instance, the description of A48.1 is 'Legionnaires' disease', a substring of A48.2, 'Nonpneumonic Legionnaires' disease'. To avoid erroneously matching model outputs we first string match for a line corresponding to the longer description.

Baselines: We report baseline results from the Pretrained-Language-Model framework (PLM-ICD) [5] which is the state-of-the-art (SOTA) model for the task of ICD coding [4]. It combines BERT as a text encoder model with per-label attention and per-label binary classification heads. Due to the limited size of the CodiEsp training dataset (500 case notes), we use the PLM-ICD provided model weights learnt on the MIMIC-IV dataset [4] [3]. This represents a realistic transfer learning scenario.

Our second 'clinical coder' baseline is the previously described out-of-the-box approach of prompting the model to act as a clinical coder, without providing further information about ICD. We identify the returned ICD codes in two ways: a) by matching the alpha-numeric codes themselves and b) by matching their natural language descriptions.

5 Results

Main Results: Performance metrics are shown in Table 1. Our evaluation metrics are micro (instances weighted equally) and macro (classes weighted equally) precision, recall and F1 scores. We consider macro-F1 to be the most representative metric for performance on rare / zero-shot codes, as it weights all classes equally. The PLM-ICD baseline performed well, demonstrating best micro-F1 score. Our out-of-the-box single prompt 'You are a clinical coder...' method demonstrated similar performance on micro-metrics but poorer performance on macro-metrics compared to our proposed tree-search method, which achieved the highest macro-F1 when GPT-4 was used.

	Micro			Macro		
Model	**Rec.**	**Prec.**	**F1**	**Rec.**	**Prec.**	**F1**
PLM-ICD	0.213	0.225	**0.219**	0.244	0.237	0.216
Clinical coder (match codes):						
Llama-2	0.011	0.033	0.016	0.007	0.011	0.006
GPT-3.5	0.163	0.155	0.159	0.149	0.161	0.136
GPT-4	0.242	0.161	0.193	0.219	0.214	0.195
Clinical coder (match descriptions):						
Llama-2	0.037	**0.282**	0.065	0.034	0.061	0.040
GPT-3.5	0.147	0.168	0.157	0.135	0.155	0.128
GPT-4	0.217	0.166	0.188	0.187	0.190	0.169
Tree-search:						
Llama-2	0.173	0.039	0.064	0.197	0.152	0.144
GPT-3.5	0.206	0.159	0.179	0.220	**0.241**	0.208
GPT-4	**0.331**	0.087	0.138	**0.381**	0.190	**0.225**

Table 1: Results for the supervised PLM-ICD model in a transfer-learning context; the single prompt 'you are a clinical coder', and the LLM tree-search algorithm on the CodiEsp English dataset.

Level-wise Analysis In Table 2 we show the performance of our GPT-4 tree-search method at each level of the ICD ontology. Numbers are cumulative i.e. performance at a lower level of the tree includes prediction failures at higher levels of the tree, allowing us to observe the level-wise degradation of performance. However, since assignable leaf codes (class labels) can occur from the 'category' level onwards, meaning some codes are "dropped" before the extension levels, it is not possible to directly relate the results from results tables 1 and 2. It can be seen that both micro- and macro-recall drop to approximately 50% already by the second "Block" level, with approximately 40% reaching the assignable levels (Subcategory, Extension I, extension II).

[3] `https://github.com/JoakimEdin/medical-coding-reproducibility`

	Micro			Macro		
	Rec.	Prec.	F1	Rec.	Prec.	F1
Chapter	0.835	0.677	0.748	0.778	0.668	0.693
Block	0.501	0.321	0.392	0.529	0.452	0.431
Category	0.401	0.201	0.268	0.448	0.392	0.364
Subcategory	0.343	0.110	0.167	0.418	0.370	0.348
Extension I	0.347	0.064	0.108	0.362	0.319	0.306
Extension II	0.192	0.028	0.049	0.216	0.212	0.199

Table 2: Cumulative performance metrics, computed on the GPT-4 tree-search CodiEsp English test set predictions.

Prediction of Mutually Exclusive Codes For the tree-search strategy, a common failure mode was predicting several similar leaf codes, leading to poor precision. For instance, GPT-4 tree-search predicted codes 'B27.89' and 'B27.80' for document `S0212-71992006000100006-1`: that is, mononucleosis with *and* without complications, when these labels are clearly mutually exclusive.

6 Discussion

Reported performance in the original CodiEsp competition was comparatively high, with a dictionary-based method winning the competition with a micro-F1 score of 0.687. However, in real clinical data there are no span-level labels to learn from. Indeed, as we have seen for many classes there are not even samples to learn from, hence our choice of a zero-shot approach. Whilst the CodiEsp models may perform well on the small fraction of labels represented in the dataset, they are not generalisable to the remaining labels, whereas our tree-search method is truly general in this sense.

The best strategy for prompting models is not well understood. We could not find a common prompt that both Llama-2 and GPT-3.5/4 would adhere to, which forced us to employ separate prompt templates for each model. Due to its relative newness, we have less experience developing prompts for Llama-2 than for GPT models, which may somewhat account for the poor performance of Llama-2 in our experiments.

The advantage of our method (sparse exploration of the ICD tree) is also its potential weakness, in the sense that false negative errors can compound at each level. For instance, for a 'subcategory' leaf code to be predicted, each of its parent, grandparent and great-grandparent codes must also be predicted; if any one of those is not predicted then the code is unreachable. This is illustrated in Table 2 which shows that ancestors of many true codes are not correctly predicted, thus the codes cannot be predicted, even if the model deems them "relevant" when directly provided with their descriptions.

There are a few promising avenues for future research. Our current approach has the advantage of being generic; we use the exact same prompt for every node and can operate with any tree of concepts for which each concept has an associated text description. However, to achieve better accuracy for ICD coding, we might introduce consideration of the ICD coding rules to allow/disallow legitimate code assignment combinations (i.e. mutually exclusive codes), or try to improve level-wise accuracy by tailoring the prompt for different tree levels (e.g. use a different approach for "chapter" prompts compared to "subcategory" prompts).

7 Conclusion

In this paper we have presented a promising approach for performing ICD coding with generative LLMs without the need for task-specific training. We note that our first LLM method, the out-of-the-box 'You are a clinical coder...' prompt, is heavily contingent on the model's pre-trained knowledge of the ICD-10 ontology and the specialist clinical coding task. Our tree-search method does not have this requirement, and thus could handle little/never seen rare codes, leading to better macro-averaged performance. This allows handling of new codes such as the `U07.1 - COVID-19` code introduced in February 2020, or whole ontology revisions such as ICD-11 (introduced in January 2022 [9]). We conclude that our LLM tree-search method and generative LLMs more broadly show exciting potential for zero-shot clinical coding.

References

[1] M. Agrawal, S. Hegselmann, H. Lang, Y. Kim, and D. 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, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics.

[2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[3] L. R. S. De Lima, A. H. F. Laender, and B. A. Ribeiro-Neto. A hierarchical approach to the automatic categorization of medical documents. In *Proceedings of the seventh international conference on Information and knowledge management*, pages 132–139, Bethesda Maryland USA, Nov. 1998. ACM.

[4] J. Edin, A. Junge, J. D. Havtorn, L. Borgholt, M. Maistro, T. Ruotsalo, and L. Maaløe. Automated Medical Coding on MIMIC-III and MIMIC-IV: A Critical Review and Replicability Study. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, pages 2572–2582, New York, NY, USA, July 2023. Association for Computing Machinery.

[5] C.-W. Huang, S.-C. Tsai, and Y.-N. Chen. PLM-ICD: Automatic ICD Coding with Pretrained Language Models. In *Proceedings of the 4th Clinical Natural Language Processing Workshop*, pages 10–20, Seattle, WA, July 2022. Association for Computational Linguistics.

[6] R. Kaur, J. A. Ginige, and O. Obst. AI-based ICD coding and classification approaches using discharge summaries: A systematic literature review. *Expert Systems with Applications*, 213:118997, Mar. 2023.

[7] J. Lu, L. Du, M. Liu, and J. Dipnall. Multi-label Few/Zero-shot Learning with Knowledge Aggregated from Multiple Label Graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2935–2943, Online, Nov. 2020. Association for Computational Linguistics.

[8] A. Miranda-Escalada, A. Gonzalez-Agirre, J. Armengol-Estape, and M. Krallinger. Overview of automatic clinical coding: annotations, guidelines, and solutions for non-English clinical cases at CodiEsp track of CLEF eHealth 2020. *CLEF*, Sept. 2020.

[9] P. Pezzella. The ICD-11 is now officially in effect. *World Psychiatry*, 21(2):331–332, June 2022.

[10] K. Singhal, T. Tu, J. Gottweis, R. Sayres, E. Wulczyn, L. Hou, K. Clark, S. Pfohl, H. Cole-Lewis, D. Neal, M. Schaekermann, A. Wang, M. Amin, S. Lachgar, P. Mansfield, S. Prakash, B. Green, E. Dominowska, B. A. y. Arcas, N. Tomasev, Y. Liu, R. Wong, C. Semturs, S. S. Mahdavi, J. Barral, D. Webster, G. S. Corrado, Y. Matias, S. Azizi, A. Karthikesalingam, and V. Natarajan. Towards Expert-Level Medical Question Answering with Large Language Models, May 2023. arXiv:2305.09617 [cs].

[11] C. Song, S. Zhang, N. Sadoughi, P. Xie, and E. Xing. Generalized Zero-Shot Text Classification for ICD Coding. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 4018–4024, Yokohama, Japan, July 2020. International Joint Conferences on Artificial Intelligence Organization.

[12] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open Foundation and Fine-Tuned Chat Models, July 2023. arXiv:2307.09288 [cs].

[13] World Health Organization. ICD-10 : international statistical classification of diseases and related health problems : tenth revision. Technical report, World Health Organization, 2004. ISBN: 9789241546492.

[14] Z. Yang, S. Wang, B. P. S. Rawat, A. Mitra, and H. Yu. Knowledge Injected Prompt Based Fine-tuning for Multi-label Few-shot ICD Coding. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1767–1781, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics.

A Appendix

A.1 Clinical coder prompt

Below we show the prompt used for the out-of-the-box 'You are a clinical coder...' method.

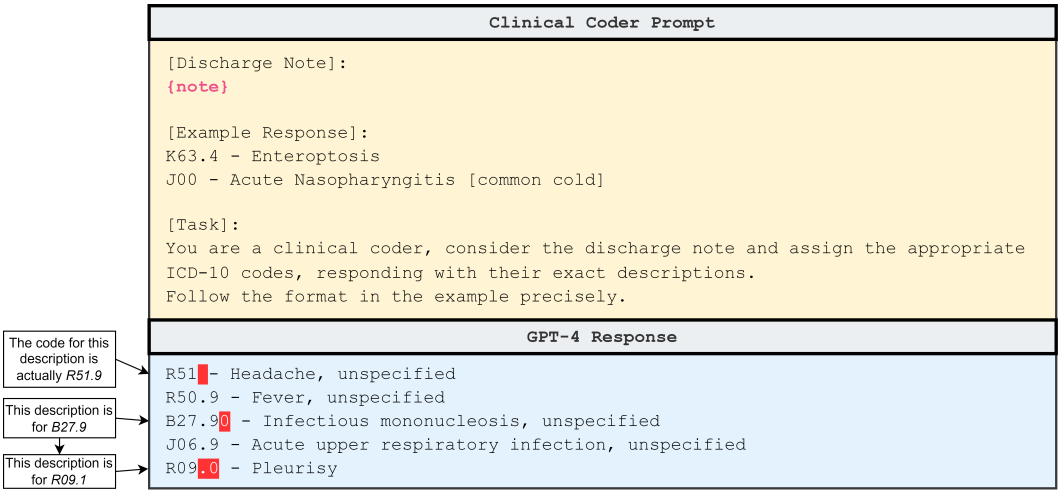

Figure 2: The 'You are a clinical coder...' prompt used for both GPT and Llama models and a sample GPT-4 response for document `S1130-05582008000500007-1`. The discrepancies between the true ICD codes and code descriptions generated by GPT-4 are highlighted in red.

A.2 LLM guided Tree-Search prompts

Below we show the two prompts used in our tree-search method for the GPT and Llama models respectively.

GPT-3.5/4 Tree Search prompt

```
[Case note]:
"""
{note}
"""

[Example]:
<example prompt>
Gastro-esophageal reflux disease
Enteroptosis

<response>
Gastro-esophageal reflux diease: Yes, Patient was prescribed omeprazole.
Enteroptosis: No.

[Task]:
Consider each of the following ICD-10 code descriptions and evaluate if there are
any related mentions in the case note.
Follow the format in the example precisely.

{code descriptions}
```

Llama-2 Tree Search Prompt

```
[Case note]:
{note}

[Example]:
<code descriptions>
* Gastro-esophageal reflux disease
* Enteroptosis
* Acute Nasopharyngitis [Common Cold]
</code descriptions>

<response>
* Gastro-esophageal reflux diease: Yes, Patient was prescribed omeprazole.
* Enteroptosis: No.
* Acute Nasopharyngitis [Common Cold]: No.
</response>

[Task]:
Follow the format in the example response exactly, including the entire
description before your (Yes|No) judgement, followed by a newline.
Consider each of the following ICD-10 code descriptions and evlaluate if there
are any related mentions in the Case note.

{code descriptions}
```

Figure 3: Tree-search prompt templates for GPT and Llama models.

We found that Llama-2 exhibited poor adherence to the desired output format when prompts developed for GPT were used. Differences between the prompts reflect experimental improvements in achieving model adherence to this output format. For example, Llama-2 preferred to respond with bullet points, so we adapted our prompt to encourage this, as it improved the parse-ability of the model response.

A.3 Tree-search algorithm

Below we describe the steps of the tree-search algorithm.

Algorithm 1 Assign a set of ICD codes for a case note using LLM-guided tree-search.

Requisites:

- **tree**: an ICD ontology, with methods for accessing the natural language code descriptions, selecting the children of a code and checking if a given code is assignable (i.e a leaf code).
- **prompt_template**: An LLM prompt which embeds the string variables 'case_note' and 'code_descriptions'.
- **llm_api_request**: a function accepting a case note and a set of code descriptions to be inserted into the prompt. Returns a text completion from the LLM.
- **match_code_descriptions**: a function which takes an LLM text completion and the set of candidate code descriptions and returns the codes which the model predicted as being relevant.

```
function SEARCH_TREE(case_note)
    assigned_codes ← ()                                   ▷ Predicted leaf codes
    candidate_codes ← tree.root.children                  ▷ The set of ICD-10 chapters

    while true do
        code_descriptions ← tree.get_descriptions(candidate_codes)
        llm_response ← llm_api_request(case_note, code_descriptions)
        predicted_codes ← match_code_descriptions(llm_response, code_descriptions)

        for code in predicted_codes do
            if code in tree.assignable_codes then
                assigned_codes.append(code)
            else
                parent_codes.append(code)
            end if
        end for

        if parent_codes.length > 0 then
            parent_code ← parent_codes.pop(0)
            candidate_codes ← tree.get_child_codes(parentCode)
        else
            break                                         ▷ There are no further codes to explore.
        end if
    end while
    return assigned_codes
end function
```

Under the simplifying assumption that generating a response to each prompt takes constant time, the time complexity for the LLM-guided tree-search algorithm is the same as that of a multi-label decision tree query: $\mathcal{O}(k \cdot \log(d))$, where k refers to the number of predicted labels and and d refers to the depth of the tree at which each of these labels is found.

Whilst any single path from the root of the tree to a leaf is on average ≈ 4 steps, if many paths are explored, the number of prompts can be large (> 100). We therefore enforce a simple prompt limit (not described in Algorithm A.3) in place of the `while true` loop to avoid excessive steps. We set this hyper-parameter to a value of 50, as early experiments suggested that additional steps beyond this yield little benefit.