

Utilizing Large Language Models to Detect and Decipher Abbreviation in Clinical Notes

Hamed Jafarpour, Cheligeer (ken) Cheligeer, Jun Yan, Yuan Xu, Cathy Eastwood,
Danielle A Southern, Yong Zeng, Hude Quan

Submitted to: JMIR Medical Informatics
on: March 16, 2025

Disclaimer: © The authors. All rights reserved. This is a privileged document currently under peer-review/community review. Authors have provided JMIR Publications with an exclusive license to publish this preprint on its website for review purposes only. While the final peer-reviewed paper may be licensed under a CC BY license on publication, at this stage authors and publisher expressly prohibit redistribution of this draft paper other than for review purposes.

Table of Contents

Original Manuscript..... 5

Supplementary Files..... 32

0..... 32

Preprint
JMIR Publications

Utilizing Large Language Models to Detect and Decipher Abbreviation in Clinical Notes

Hamed Jafarpour¹; Cheligeer (ken) Cheligeer²; Jun Yan¹; Yuan Xu²; Cathy Eastwood²; Danielle A Southern²; Yong Zeng³; Hude Quan²

¹414-3501 Chemin de la Côte-Sainte-Catherine Concordia University Montreal CA

²University of Calgary Calgary CA

³Concordia University Montreal CA

Corresponding Author:

Hude Quan

University of Calgary

TRW3E23, University of Calgary, 2500 University Drive NW

Calgary

CA

Abstract

Background: Preprocessing serves to standardize clinical documentation and create consistency throughout all clinical notes. An essential aspect of this process is text normalization, which mainly entails recognizing abbreviations and translating them into their full forms. A range of methods, including Natural Language Processing (NLP) and Machine Learning (ML) algorithms, are employed to identify and interpret abbreviations found in clinical notes. Recently, fine-tuning Large Language Models (LLMs) and applying prompt engineering techniques have been utilized to carry out various tasks involving clinical notes. Each method presents its own set of benefits and drawbacks.

Objective: To utilize LLMs to detect and decipher abbreviations in clinical notes effectively and efficiently.

Methods: A framework is proposed for the detection and deciphering of abbreviations within clinical text, using LLMs. This framework consists of four phases. First, the task and sub-tasks are identified. Second, the relevant properties of task and sub-tasks within the designated domain are clearly defined. Third, these properties are utilized to generate optimized examples. The fourth stage involves the application of these properties in LLMs, which can serve either as engineering prompts enhanced by examples or be employed in the creation of a labeled dataset for fine tuning. Finally, two case studies are conducted; the first involves prompt engineering enhanced by optimized examples is employed as input for ChatGPT utilizing the GPT-4 language model, while the second experiment entails the fine-tuning of both GPT-2 and T5 models using the previously established labeled dataset.

Results: The findings emphasize a distinct advantage of example-based prompts, whether examples are selected randomly or through optimized selection, in comparison to basic prompts. The results indicate that basic prompts consistently yield lower accuracy. Furthermore, prompts enhanced by the optimized selection of 5-shots examples demonstrate significantly superior performance relative to those utilizing randomly selected examples. In contrast to the example-based prompt approach, the result indicates that the T5 models, which are fine-tuned with the labeled dataset using the proposed framework, exhibit enhanced performance when compared to the example-based prompt.

Conclusions: This research underscores the potential of LLMs for the proficient detection and interpretation of abbreviations within clinical notes. This capability is particularly evident when optimized examples are utilized in the preparation of the labeled dataset for the fine-tuning of T5 models, resulting in reduced costs and enhanced performance. Further investigation is necessary to develop an LLM model specifically designed for the detection and interpretation of abbreviations in clinical notes.

(JMIR Preprints 16/03/2025:74042)

DOI: <https://doi.org/10.2196/preprints.74042>

Preprint Settings

1) Would you like to publish your submitted manuscript as preprint?

✓ **Please make my preprint PDF available to anyone at any time (recommended).**

Please make my preprint PDF available only to logged-in users; I understand that my title and abstract will remain visible to all users.

Only make the preprint title and abstract visible.

No, I do not wish to publish my submitted manuscript as a preprint.

2) If accepted for publication in a JMIR journal, would you like the PDF to be visible to the public?

✓ **Yes, please make my accepted manuscript PDF available to anyone at any time (Recommended).**

Yes, but please make my accepted manuscript PDF available only to logged-in users; I understand that the title and abstract will remain visible to all users.

Yes, but only make the title and abstract visible (see Important note, above). I understand that if I later pay to participate in <http://www.jmir.org/>

No. Please do not make my accepted manuscript PDF available to anyone.

Original Manuscript

Utilizing Large Language Models to Detect and Decipher Abbreviation in Clinical Notes

Hamed Jafarpour¹, Cheliger(Ken) Cheliger², Jun Yan³, Yuan Xu⁴, Cathy Eastwood⁵, Danielle A Southern⁶, Yong Zeng⁷, and Hude Quan^{8,*}

¹Ph.D. Candidate of Concordia Institute for Information Systems Engineering, Gina Cody School of Engineering and Computer Science, Concordia University, Montreal, Canada.

²Data Scientist, Cumming School of Medicine, University of Calgary, Calgary, Canada

³Associate Professor of Concordia Institute for Information Systems Engineering, Gina Cody School of Engineering and Computer Science, Concordia University, Montreal, Canada.

⁴Assistant Professor of Department of Oncology, Community Health Science and Surgery, Cumming School of Medicine, University of Calgary, Calgary, Canada.

⁵Assistant Professor of Department of Community Health Sciences, Cumming School of Medicine, University of Calgary, Calgary, Canada.

⁶Senior Researcher, Cumming School of Medicine, Centre for Health Informatics, University of Calgary, Calgary, Canada.

⁷Professor of Concordia Institute for Information Systems Engineering, Gina Cody School of Engineering and Computer Science, Concordia University, Montreal, Canada.

⁸Professor of Department of Community Health Sciences, Cumming School of Medicine, University of Calgary, Calgary, Canada.

*TRW3E23, University of Calgary, 2500 University Drive NW, Calgary, Canada, T2N 1N4., E-mail: hquan@ucalgary.ca, Telephone: +1-(403)-210-8617

Abstract

Background: Preprocessing serves to standardize clinical documentation and create consistency throughout all clinical notes. An essential aspect of this process is text normalization, which mainly entails recognizing abbreviations and translating them into their full forms. A range of methods, including Natural Language Processing (NLP) and Machine Learning (ML) algorithms, are employed to identify and interpret abbreviations found in clinical notes. Recently, fine-tuning Large Language Models (LLMs) and applying prompt engineering techniques have been utilized to carry out various tasks involving clinical notes. Each method presents its own set of benefits and drawbacks.

Objective: To utilize LLMs to detect and decipher abbreviations in clinical notes effectively and efficiently.

Methods: A framework is proposed for the detection and deciphering of abbreviations within

clinical text, using LLMs. This framework consists of four phases. First, the task and sub-tasks are identified. Second, the relevant properties of task and sub-tasks within the designated domain are clearly defined. Third, these properties are utilized to generate optimized examples. The fourth stage involves the application of these properties in LLMs, which can serve either as engineering prompts enhanced by examples or be employed in the creation of a labeled dataset for fine tuning. Finally, two case studies are conducted; the first involves prompt engineering enhanced by optimized examples is employed as input for ChatGPT utilizing the GPT-4 language model, while the second experiment entails the fine-tuning of both GPT-2 and T5 models using the previously established labeled dataset.

Results: The findings emphasize a distinct advantage of example-based prompts, whether examples are selected randomly or through optimized selection, in comparison to basic prompts. The results indicate that basic prompts consistently yield lower accuracy. Furthermore, prompts enhanced by the optimized selection of 5-shots examples demonstrate significantly superior performance relative to those utilizing randomly selected examples. In contrast to the example-based prompt approach, the result indicates that the T5 models, which are fine-tuned with the labeled dataset using the proposed framework, exhibit enhanced performance when compared to the example-based prompt.

Conclusions: This research underscores the potential of LLMs for the proficient detection and interpretation of abbreviations within clinical notes. This capability is particularly evident when optimized examples are utilized in the preparation of the labeled dataset for the fine-tuning of T5 models, resulting in reduced costs and enhanced performance. Further investigation is necessary to develop an LLM model specifically designed for the detection and interpretation of abbreviations in clinical notes.

Keywords: Large Language Model; Prompt Engineering; Fine-tuning; Abbreviation; Clinical note.

Introduction

Overview

In recent years, Electronic Medical Records (EMRs) have shown significant potential for advancing clinical research [1–4]. While they contain structured data like prescriptions, test results, and diagnosis codes, a substantial portion of clinical notes remains in unstructured narrative text format. This unstructured data can make up as much as 80% of the Electronic Health Record (EHR) dataset. However, the time-consuming and labor-intensive nature of processing and extracting information from this data often leads to it being unused or abandoned and a significant challenge in clinical research applications [5, 6].

Narrative EMR data, or narrative EHR data, encompass clinical notes that comprehensively detail various aspects of a patient's record. This data contains a rich source of information that can be leveraged through the utilization of Natural Language Processing (NLP) [7–10], Machine Learning (ML) techniques [11–15], and large language models (LLMs) [16–18]. Such text exhibits distinctive characteristics, including grammatical errors, shortened words, frequent abbreviations, acronyms, specialized characters with specific meanings, and occasional misspelled words [19]. Consequently, one of the pivotal tasks for utilizing clinical text is preprocessing. Preprocessing aims to normalize clinical text and establish uniformity across all clinical notes.

Text normalization represents a vital preprocessing task that strives to achieve uniformity

across all terms within the clinical text, converting it into a standardized and noise-free format. This task involves word tokenization and normalization. Word tokenization entails the identification of individual words (or phrases), abbreviations, and special characters, while word normalization focuses on interpreting clinical abbreviations and special characters, and establishing uniformity among similar terms within the clinical text. In the context of clinical texts, an abbreviation is an acronym or a shortened form of a word or phrase used for efficiency and brevity, often representing common medical terms or concepts. For instance, the sentence “Pt is 65 yo m w/ -fx referred to pt for lbp” is transformed through the processes of text normalization to read as “Patient is 65 years old male with no fracture referred to physical therapy for low back pain.”

LLMs are artificial intelligence (AI) systems trained on vast text datasets, leveraging neural networks and deep learning techniques [20]. LLMs aim to represent complex word relationships and have shown promising applications in fields like medicine [21, 22]. In order to effectively utilize a LLM as a facilitative tool, users must submit their requests based on specific prompts or engage in fine-tuning or training the LLM to cater to their particular domain of interest.

Several researchers engage in the fine-tuning of LLMs to enhance their performance within specific domains (e.g., [23–26]). Fine-tuning is defined as the procedure of taking a trained LLM and subsequently tuning it on a targeted dataset, thereby adapting it to a particular task or domain [27]. This process entails the modification of the model’s weights utilizing task-specific data, frequently employing supervised learning methods. The fine-tuning process is instrumental in enhancing performance for specialized applications while leveraging the extensive knowledge acquired during the initial pretraining phase.

This paper presents a framework designed to delineate the properties of the specified task, which, in this study, pertains to the detection and decipherment of abbreviations within clinical notes. The framework illustrates these properties through a series of examples pertaining to the detection and decipherment of abbreviations. Subsequently, these examples, which encompass all relevant properties of the task, are incorporated into LLM, as an engineered prompt that is enhanced by optimized samples or labeled dataset for fine-tuning LLMs. This framework effectively narrows the gap between user requirements and LLMs by mapping the properties of task and sub-tasks into the LLMs either as engineered prompt or fine-tuning LLMs. This method streamlines the execution of tasks utilizing LLMs, while also reducing the necessity for labor-intensive, time-consuming, and costly efforts.

In alignment with the objectives of this paper, the research questions are delineated as follows:

1. In what ways are LLMs employed for the detection and decipherment of abbreviations within clinical texts?
2. How are the characteristics of the task defined?
3. In what manner does example-based prompt facilitate the detection and decipherment of abbreviations in clinical texts?
4. How do fine-tuned LLMs effectively detect and decipher abbreviations in clinical texts?

The research hypothesis is as follows: “Employing LLMs detect and decipher abbreviations within clinical notes in an effective and efficient manner.”

Background

Preprocessing is used to standardize clinical text and establish uniformity across all clinical

notes. One key task in this process is text normalization, which primarily involves identifying abbreviations and deciphering them by converting them into their complete forms. Various approaches, including NLP and ML algorithms, are used to detect and decipher abbreviations in clinical notes. Fine-tuning LLMs and prompt engineering techniques have been recently applied to perform various tasks using clinical notes. Each approach has its own advantages and disadvantages.

NLP approaches. Numerous clinical NLP systems, including tools like MedLEE [28, 29], MetaMap [30, 31], KnowledgeMap [32], MedTagger [33], and MedEx [34] have specific modules dedicated to abbreviation detection and expansion. The majority rely on NLP approaches, which are rule-based. Indeed, valuable repositories of biomedical vocabulary such as the Unified Medical Language System (UMLS) [35], Systematized Nomenclature of Medicine Clinical Terminology (SNOMED-CT) [36], and RxNorm [37] have a repository for abbreviations and their expanded forms. However, navigating these repositories can be complex and may require specific knowledge of their structure and tools to effectively extract abbreviation data. Additionally, because these repositories rely on user expertise to detect abbreviations in context, they may not always provide straightforward solutions.

ML approaches. With the increasing popularity of ML algorithms, numerous researchers have sought to address challenges using these techniques. Rajkomar et al. employed an ML approach to decode clinical abbreviations, utilizing a diverse set of data sources for training, including public web data, synthetic snippets, CASI [38], MIMIC-III [39], and i2b2-2014 [40], thus preserving data privacy [41]. In developing supervised ML algorithms, the preparation of labeled data is fundamental. Many researchers have leveraged supervised ML algorithms to detection and interpret of abbreviations and acronyms concepts [42–45]. Transformer is a deep learning architecture that leverages the parallel multi-head attention mechanism [46]. Recently, some researchers have applied the BERT transformer model, pre-trained using unsupervised learning, to disambiguate clinical abbreviations [47, 48]. While transformers have revolutionized NLP and achieved state-of-the-art performance on various tasks, they also come with challenges and limitations. In addition to computational complexity and a large memory footprint, fine-tuning requirements and data constraints remain prominent concerns. Significant volumes of data are often needed for fine-tuning to optimize transformers to specific tasks. In turn, their performance may suffer under limited data for fine-tuning.

Prompt engineering. Prior literature indicates that previous studies have focused on using question-answering techniques with untailored prompts. These approaches involve cloze prompts where a portion of the text is masked or removed, and the model is tasked with filling in the missing information [49, 50]. Researchers have typically employed this cloze prompt format and manually crafted templates for querying ChatGPT. While the creation of manual templates is an intuitive strategy that can

address various tasks to some extent, it presents inherent challenges as follows:

1. Creating and experimenting with these prompts requires artistic skills, time, and experience, particularly for complex tasks like semantic parsing [51].
2. Even experienced prompt designers discover the most optimal prompts manually. [52].
3. Prompt augmentation, which incorporates examples as part of the prompt, remains challenging, with results varying from near state-of-the-art accuracy to near-random guesses [53]. In

response to this challenge, [54] and [55] employ sentence embedding to choose examples closely related to the input in the embedding space. Mishra et al. assessed the generalizability of pre-trained language models for new tasks by providing positive and negative examples illustrating both desired outcomes and pitfalls to avoid [56]. However, this critical question remains unresolved: How can we effectively determine which examples should be included in the prompt or as labeled data in a dataset to achieve optimal results?

Training and fine-tuning LLMs: Training LLMs from scratch to effectively perform text normalization in clinical notes necessitates considerable computational resources [57]. Fine-tuning LLMs presents several challenges, including substantial computational requirements, issues related to data quality, and complexities associated with model overfitting and generalization. Secondly, the success of fine-tuning is contingent upon the availability of high-quality, well-annotated datasets, which may be limited in specialized fields. Lastly, conducting fine-tuning on minimal datasets can result in overfitting, thereby diminishing the model's efficacy when exposed to previously unseen data. Although various techniques, including dropout, weight regularization, and data augmentation, can enhance generalization, these approaches also require additional computational resources.

Gap identification Challenges associated with ML and the LLMs approach often arise from a scarcity of effective labeled data, particularly in tasks that involve limited datasets. Furthermore, the prompt engineering methods require extensive experience in designing effective prompts to achieve optimal results. This study proposes a framework that first identifies the properties of a given task and sub-tasks and subsequently provides examples that represent those properties. These optimized examples will either be incorporated into LLMs to generate an optimized example-based prompt or utilized to create a dataset for fine-tuning LLMs.

Methods

Proposed fine-tuning framework

We present a prompt engineering framework enhanced through optimized examples. Through the development of this framework, we leverage question-and-answer approaches, to clarify the user's need in the prompt [58]. The proposed model is intended for situations where the input comprises both a prompt and a block of text as a target text for ChatGPT to perform actions on that text. Figure 1 demonstrates that the model consists of seven key components: task definition, properties identification, example selection, and utilizing LLMs. Each component will be described in detail further.

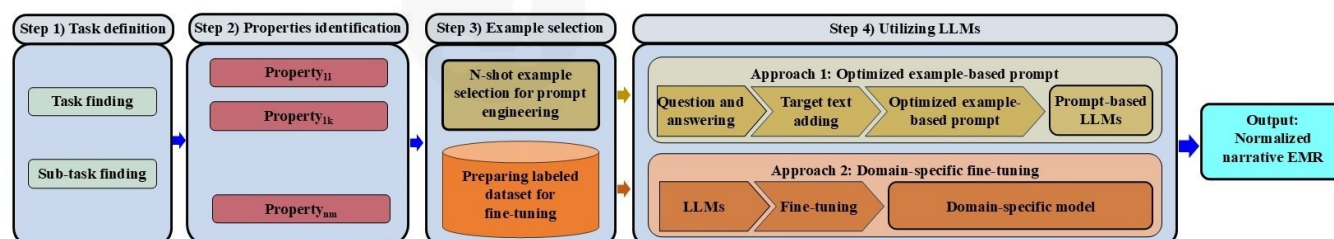


Figure 1. The proposed model for generating learned multi-prompt.

Step 1: Task definition

The first step is task specification, which refers to the user's need. A set of text-related tasks is identified, which includes tasks such as text normalization and data extraction:

{Task= text normalization, data extraction}

In general, each task encompasses two or more sub-tasks, and all the sub-tasks associated with a specific task for a particular task X are detailed as follows:

$X_{Task} = \{\text{sub-task1, sub-task2, sub-task3,}\}$

Example: $X_{TN} = \{\text{finding abbreviations(Abv), deciphering abbreviations}\}$

Step 2: Properties identification

In general, each sub-task possesses one or more properties. A property is defined as a predetermined attribute or principle that outlines the characteristics of a specific aspect of the task. These properties must comprehensively address all conditions pertinent to the task at hand. In the context of this study, the task involves the detection and conversion of abbreviations to their complete forms within clinical notes. The properties associated with the task should encompass all relevant attributes for both the primary task and its associated sub-tasks.

It is important to note that these properties may either intersect or remain independent of one another. In this step, all properties that encompass the entirety of the tasks and sub-tasks are delineated.

- Property 1: The abbreviation can be written using all uppercase letters (e.g., “HTN”: “hypertension”).
- Property 2: The abbreviation can be written using all lowercase letters (e.g., “htn”: “hypertension”).
- Property 3: The abbreviation can be written with the first letter capitalized and the rest in lowercase (e.g., “Pt”: “patient”).
- Property 4: The abbreviation can be a combination of lowercase/uppercase letters and special characters (e.g., “h/o”: “history of”, “-f/c”: “no fever and/or chills”).
- Property 5: The abbreviation can be a combination of lowercase/uppercase letters and numbers (e.g., “85f”: “85-year-old female”, “T4”: “Thyroxine”).
- Property 6: The abbreviation can be a combination of letters, special characters, and numbers (e.g., “BP ↑ 140/90”: “blood pressure increased, with a systolic pressure of 140 mmHg and a diastolic pressure of 90 mmHg”).
- Property 7: The abbreviation can have the same form but different meanings depending on the context (e.g., “CA”: “Canada”, “CA”: “California”, or “CA”: “Cancer”).

Properties 1 and 7 intersect, meaning their combinations can be applied simultaneously. However, some properties are mutually exclusive (e.g. properties 1 and 2). Regardless of this overlap or exclusivity, the most important consideration is that the prompt designer fully understands and defines all relevant properties related to the task and sub-tasks. This comprehensive understanding aids in the identification of pertinent examples.

Step 3: Example selection

It is imperative to select relevant examples that align with the properties outlined in the previous step. The quantity of examples should be adequate, provided that they encompass all specified properties. Below are all examples that effectively illustrate these properties for abbreviation detection and deciphering.

- Example 1: Example which covers properties 1, 3, 4, and 5: Pt is 85f with h/o HTN, asthma p/w respiratory failure.

Expected output is:

‘Pt’:‘patient’, ‘85f’:‘85 years old’, ‘h/o’:‘history of’, ‘HTN’:‘hypertension’, ‘p/w’:‘presented with’

- Example 2: Example which covers properties 2, 3, and 7: Pt with copd, living in CA, has history of CA.

Expected output is:

‘Pt’:‘patient’, ‘copd’:‘chronic obstructive pulmonary disease’, ‘CA’:‘Canada/California’, ‘CA’:‘cancer’

Step 4: Utilizing LLMs

The examples extracted in Step 3 are utilized for the generation of optimized example-based prompts or for the creation of datasets intended for fine-tuning purposes.

- Optimized example-based prompt: The optimized example-based prompt, an engineered prompt, is constructed by concatenating the responses to all relevant questions (i.e., questions 1 to 9) formulated in previous steps. Table 1 presents all the questions along with their template responses that contribute to the creation of an engineered prompt. Any text highlighted in blue must be closely associated with a specific task and purpose, and it should be appropriately masked.
- Domain-specific fine-tuning: This approach utilizes the properties identified in Step 2 to prepare a domain-specific labeled dataset. Subsequently, the fine-tuning of the LLM models is conducted using the prepared dataset.

Table 1. Questions and responses to make the engineered prompt.

Questions	Responses
Q1: What is the nature of the task?	R1: The task is the [Mask Task] in the below text.
2: What is the main subject of the text?	R2: The below text is [Mask Text].
Q3: What concept is presented in the text?	R3: The text [Mask Text Concept].
Q4: What sub-tasks are involved?	R4: The sub-tasks are [Mask Sub Tasks].
Q5: What is the expected format for the responses?	R5: The response must be in [Mask Response].

Q6: What are the properties for each subtask?	R6: All examples related to the properties for subtasks are identified as follows:
Q7: What are the examples that cover all the properties?	R7: The examples are [Mask Example Selection].
Q8: What response shape should be adhered to the target text?	R8: Please provide the response in [Mask Response] [Mask Response Shape].
Q9: What is the context of the target text?	R9: The [Mask Text] that must be used is here: [Mask Target Text].

Masking the model for normalizing clinical notes is available in Appendix 1.

Results

Dataset

This paper utilizes the Clinical Abbreviation Sense Inventory (CASI), a publicly available collection of abbreviations and acronyms, each associated with possible meanings [38]. However, the publicly available dataset was insufficient for our specific purpose, so the author expanded the CASI dataset by incorporating additional abbreviations, acronyms, and shortened words. Random clinical notes, including social history, history of present illness, past medical history, past surgical history, discharge diagnosis, and hospital course, were selected from the CASI dataset.

To enhance the dataset, the author developed and applied a comprehensive guideline (See Appendix 2.). This guideline enabled the identification and inclusion of additional abbreviations, acronyms, shortened words, and their expanded forms that were not originally part of the CASI dataset. These new entries were added to the expanded CASI dataset as a new column.

The datasets are publicly available in the “Supplementary File” attached to this paper (See Appendix 3.). Metadata regarding the dataset can be found in Table 2.

The table reveals notable variations in sentence and word counts across different clinical note categories. Overall, the dataset includes 1,292 notes with 6,925 sentences and 94,389 words.

Table 2. Metadata for the dataset.

Clinical note	Note count	Sentence count	Word count	Abbreviation count
Social history	209	1,202	14,838	192
History of present illness	240	981	18,289	214

Past medical history	244	1,614	18,009	191
Past surgical history	163	1,025	11,150	148
Discharge diagnosis	177	918	12,385	212
Hospital course	259	1,185	19,718	234
Total number	1,292	6,925	94,389	1,191

Configurations for LLMs

1. **Configurations for example-based prompt approach:** The experiments were conducted to evaluate the prompts using the GPT-4 language model. For these evaluations, the temperature parameter was set to zero in order to minimize the randomness in the responses generated by the model.
2. **Configurations for fine-tuning approach:**
 - **LLMs variants:** In this experimental investigation, two distinct transformer models are utilized: GPT-2 [59], encompassing small, large, and extra-large variants, and T5 [60], which similarly includes small, large, and extra-large versions. The terms “extra-large”, “large”, and “small”, as employed in the extant literature, correspond to the parameter sizes of each model. Parameters denote the adjustable weights within the neural network that undergo optimization during the training process to accurately forecast subsequent words within a given sequence. This study leverages these two models for the interpretation of clinical text. The configurations and characteristics of each transformer model are elaborated upon in the following sections.
 - a. **GPT-2 (small):** Decoder transformer model with 124 million parameters, 12 layers, 12 attention heads, embedding size of 768, vocabulary of 50,257 tokens, and maximum token length of 512 (by default this value is 1024).
 - b. **GPT-2 (large):** Decoder transformer model with 774 million parameters, 36 layers, 20 attention heads, embedding size of 1280, vocabulary of 50,257 tokens, and maximum token length of 1024.
 - c. **GPT-2 (extra-large, 1.5B):** Decoder transformer model with 1.5 billion parameters, 48 layers, 25 attention heads, embedding size of 1600, vocabulary of 50,257 tokens, and maximum token length of 1024.
 - d. **T5 (small):** Encoder-decoder transformer model with 60 million parameters, hidden size of 512, 6 layers, 8 attention heads, feed-forward size of 2048, vocabulary of 32,128 tokens, and maximum token length of 512.
 - e. **T5 (large):** Encoder-decoder transformer model with 770 million parameters, hidden size of 1024, 24 layers, 16 attention heads, feed-

forward size of 4096, vocabulary of 32,128 tokens, and maximum token length of 512.

- f. **T5 (extra-large, 3B):** Encoder-decoder transformer model with 3 billion parameters, hidden size of 1536, 24 layers, 32 attention heads, feed-forward size of 6144, vocabulary of 32,128 tokens, and maximum token length of 512.
- **Parameters settings:** The transformer models, configured with default settings, have been applied for fine-tuning to detect the abbreviations and discern them from the clinical text. The batch size has been established at 8, accompanied by a learning rate of 1×10^{-6} . This configuration was consistently employed throughout the training process. A total of six models were developed, comprising three models utilizing GPT-2 and three models utilizing T5. During the training process, both the training and validation losses were closely monitored as Figure 2 illustrates.
- **Train, validation, and test set:** The dataset is divided into three distinct subsets: the train set, the validation set, and the test set. The test set comprises 70% of the total data, while both the validation set and the test set each constitute 15% of the overall dataset. During the training process, a 10-fold cross validation is employed. The monitoring of training and validation losses is conducted for each fold and at each epoch throughout the training phase. The Gaussian Error Linear Unit (GELU) activation function, employed in all LLMs.

The cross-entropy loss function applicable to text-to-text transformer [60] is formally defined as follows:

$$L = \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T y_{it} \log(\hat{y}_{it}) \quad (1)$$

In Equation 1, N represents the total number of sequences in the batch, T denotes the max sequence length after padding. Indeed, y_{it} is the one-hot encoded ground truth token at position t in sequence i . $\hat{y}_{it}$ signifies the model-predicted probability for the correct token at position t , obtained via softmax. Furthermore, the padding token IDs within the target sequence are excluded from consideration by the loss function. In both PyTorch and TensorFlow, this can be achieved by substituting these IDs with -100, which serves as the ignore index for the cross-entropy loss [61]. Finally, the sum runs over all target tokens, and the loss is averaged over the batch. This loss function is designed to penalize incorrect predictions by calculating the negative logarithm of the probability. Consequently, it encourages the model to yield probabilities that closely align with the established ground truth.

- **Early stopping criteria:** For all models, the early stopping criteria for training are established based on two specific conditions. The training process will be terminated if either of the following occurs: 1) the average validation loss across

all folds in a single epoch falls below 0.05. This threshold is chosen based on Prechelt's findings in [62], which indicate that an average loss below approximately 0.04 results in increased costs and diminished generalizability for the model; 2) the validation loss does not exhibit an improvement of at least 0.01 over three consecutive epochs, which suggests that the patience parameter is set to 3.

Evaluation metrics

Clinical abbreviation disambiguation can be conceptualized as a form of translation, wherein abbreviations are systematically clarified [41]. Assessing text similarity is imperative in NLP tasks, including translation and text generation. Two prevalent metrics utilized for the evaluation of these tasks are ROUGE and BLEU [59,63]. Therefore, this study will employ the following evaluation metrics:

- **ROUGE (Recall-Oriented Understudy for Gisting Evaluation):** ROUGE is primarily used for evaluating text generation models by comparing generated text with reference summaries [64]. It focuses on N-gram overlapping. Common variants include ROUGE-N and ROUGE-L. ROUGE-N measures n-gram overlap (e.g., ROUGE-1 for unigrams, ROUGE-2 for bigrams) and ROUGE-L uses the Longest Common Subsequence (LCS) to measure similarity. ROUGE is recall-focused, meaning it rewards systems that generate text similar to reference in terms of content overlap.
- **BLEU (Bilingual Evaluation Understudy):** BLEU is an evaluation metric mainly used for machine translation and measures precision-based n-gram overlap [65]. Unlike ROUGE, which prioritizes recall, BLEU emphasizes precision, ensuring that the generated text does not include excessive or irrelevant content. BLEU uses n-gram precision (e.g., BLEU-1 for unigrams, BLEU-2 for 2-grams (up to 4-grams)). Indeed, it includes a brevity penalty to prevent short outputs from getting high scores.

In contrast to the ROUGE metric that the LCS can be calculated automatically, the user must define the number of N-grams in the BLEU metric. In literature, the BLEU metric is applied for the translation task and the ROUGE metric for the summarization task [59]. Since detecting and deciphering the abbreviations is a mix of summarization and translation, both metrics are applied in this study.

Case Study 1: Results of example-based prompting

The proposed framework facilitates the generation of prompts, which are subsequently utilized as engineered prompts for GPT-4. An evaluation of the proposed framework is presented in Table 3.

Table 3. Evolution across clinical notes categories using different prompting methods and different metrics ^a.

Prompt method	Evaluation
---------------	------------

	B-1	B-2	R-1	R-2	R-l
Basic prompt	0.024	0.006	0.048	0.008	0.045
1-shot example optimized selection (proposed method)	0.367	0.304	0.450	0.322	0.383
3-shots examples optimized selection (proposed method)	0.354	0.296	0.461	0.329	0.387
5-shots examples optimized selection (proposed method)	0.413	0.347	0.549	0.390	0.463
3-shots example random selection	0.219	0.158	0.280	0.171	0.244
5-shots example random selection	0.360	0.295	0.480	0.331	0.400

^a In the evaluation column, the designations “B-1”, “B-2”, “R-1”, “R-2”, and “R-l” correspond to BLEU-1, BLEU-2, ROUGE-1, ROUGE-2, and ROUGE-L, respectively. All scores presented are F1 scores.

Basic prompt: The basic prompt consistently demonstrates lower accuracy when compared to the application of example-based prompts, including both optimized example selection and random example selection, across all evaluation metrics. This observation underscores the premise that the incorporation of examples within the prompts substantially improves the model’s performance.

Optimized example selection (proposed method): Among the various strategies evaluated, the 5-shots optimized example selection method demonstrates consistently superior accuracy across all assessment metrics. This finding indicates that increasing the number of examples to five within the optimized example selection framework yields the most effective performance.

Random example selection: When employing random example selection, there exists a notable advantage between the 3-shots and 5-shots approaches in this method. However, a comparison between the randomly selected 3-shots and 5-shots samples, as well as the optimized 3-shots and 5-shots samples, indicates that the optimized sample selection yields superior performance.

Case Study 2: Results of fine-tuning model

The models have been fine-tuned using a single instance of GH200 (96 GB), which includes 64 CPU cores, 463.9 GB of RAM, and a 4.4 TB SSD. Figure 2 illustrates the fine-tuning process involving GPT-2 and T5, with distinctions made between training and validation losses for each epoch. Detailed information regarding training, testing, losses, and associated computational costs is provided in Error: Reference source not found and Table 5.

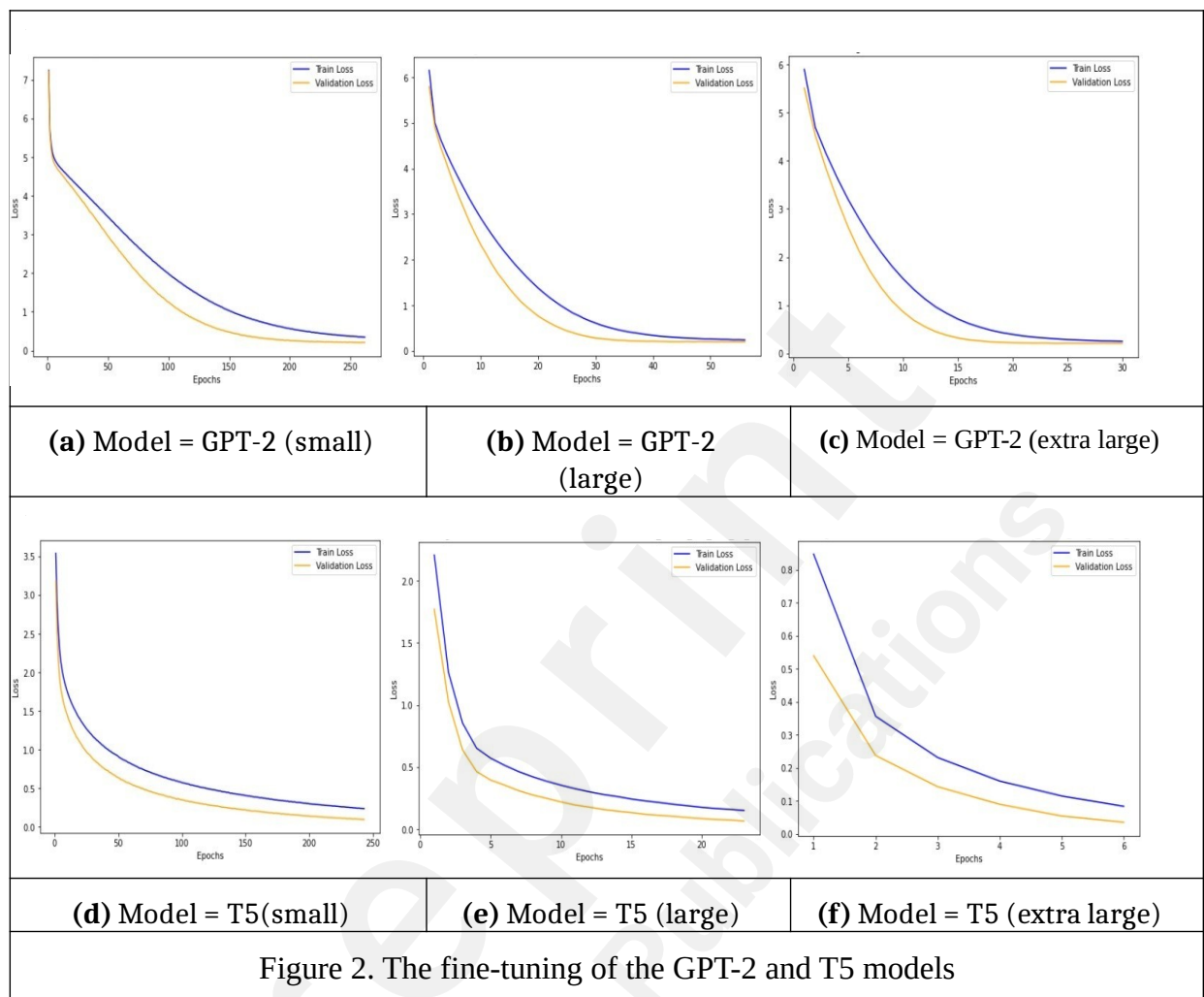


Table 4. Cross-entropy loss metric for various variants of GPT-2 and T5 models.

	Train	Validation	Test	GPU/hr
Models				
GPT-2 (small)	0.342	0.208	11.334	9.578
GPT-2 (large)	0.240	0.198	10.171	10.833
GPT-2 (extra large)	0.249	0.206	9.767	11.216
T5 (small)	0.237	0.097	0.400	3.866
T5 (large)	0.150	0.066	0.254	4.263
T5 (extra large)	0.083	0.035	0.163	4.341

Table 5. Performance of various variants of GPT-2 and T5 models ^a

	Train					Test				
Models	B-1	B-1	R-1	R-1	R-l	B-1	B-1	R-1	R-1	R-l
GPT-2 (small)	0.614	0.576	0.674	0.604	0.674	0.039	0.019	0.054	0.019	0.054
GPT-2 (large)	0.719	0.673	0.754	0.682	0.754	0.040	0.022	0.056	0.024	0.056
GPT-2 (extra large)	0.664	0.616	0.714	0.638	0.714	0.034	0.018	0.046	0.019	0.045
T5 (small)	0.911	0.856	0.920	0.842	0.920	0.789	0.721	0.803	0.702	0.808
T5 (large)	0.926	0.878	0.934	0.896	0.934	0.835	0.766	0.855	0.747	0.854
T5 (extra large)	0.736	0.707	0.946	0.883	0.946	0.709	0.666	0.878	0.785	0.879

^a In the train and test columns, the designations “B-1”, “B-2”, “R-1”, “R-2”, and “R-l” correspond to BLEU-1, BLEU-2, ROUGE-1, ROUGE-2, and ROUGE-L, respectively. All scores presented are F1 scores.

The evaluation results presented in Error: Reference source not found and Table 5 unequivocally indicate that all GPT-2 models exhibit pronounced overfitting. This phenomenon is evidenced by the substantial discrepancy observed between training and test across all variants of GPT-2. For example, GPT-2 (small) attains elevated BLEU and ROUGE scores during training (e.g., BLEU-1 = 0.614, ROUGE-L = 0.674); however, these scores experience a significant decline when assessed on the test set (BLEU-1 = 0.039, ROUGE-L = 0.054). A similar pattern is evident in both GPT-2 (large) and GPT-2 (Ex-large), wherein test performance remains consistently low, despite the enhancements noted in training metrics. The exceedingly high test loss values reported for the GPT-2 models (e.g., 11.334 for GPT-2 (small) and 9.767 for GPT-2 (Ex-large)) further corroborate the issue of overfitting, as these models demonstrate an inability to generalize effectively to previously unseen data.

In contrast, the T5 models demonstrate significantly superior generalization capabilities. T5 (small) is able to maintain relatively high test scores (e.g., BLEU-1 = 0.789, ROUGE-L = 0.808), which closely align with its training performance. This trend persists across T5 (large) and T5 (Ex-large), where test performance remains robust, and the gap between training and test loss is minimal. Notably, T5 (Ex-large) registers the lowest test loss (0.163), underscoring its stability and robustness.

Furthermore, the computational efficiency of T5 models is noteworthy. While the GPT-2 models necessitate over 9 GPU hours for completion, T5 models achieve superior performance with substantially reduced computational costs (e.g., T5 (small) at 3.866 hours and T5 (Ex-large) at 4.341 hours). This data suggests that T5 not only outperforms GPT-2 in terms of effectiveness but also demonstrates greater efficiency for the task at hand.

In summary, these findings strongly advocate for the preference of fine-tuning T5 over GPT-2 for detecting and deciphering the abbreviations in clinical notes that require robust generalization capabilities.

Test the model with the internal dataset

The error analysis for this study is conducted in two distinct parts as outlined below:

1. **Error analysis for the proposed framework:** The error analysis demonstrates that the basic prompt exhibits sub-optimal performance across all metrics, as indicated by low BLEU and ROUGE scores. This deficiency can be attributed to the absence of contextual examples to inform the model's responses. In contrast, the implementation of optimized example selection methods—specifically the 1-shot, 3-shots, and notably the 5-shots approaches—significantly enhances performance, with the 5-shots method yielding the highest results across all evaluated metrics. This underscores the critical role of providing multiple relevant examples. Although random example selection also leads to improved performance compared to the basic prompt, it fails to achieve the effectiveness seen with optimized examples, likely due to the presence of irrelevant or poorly aligned examples. Overall, the findings highlight that the careful selection of task-specific examples is essential for maximizing model performance in the generation of clinical notes.
2. **Error analysis for fine-tuning LLMs:** This error analysis underscores a significant concern regarding overfitting in GPT -2 models while highlighting the impressive generalization capabilities of T5 models.
 - **GPT-2:** A significant source of error observed in the GPT-2 models is their inadequate generalization capabilities, as evidenced by the considerable decline in test scores relative to training scores. This trend indicates that GPT -2 models may tend to memorize training data rather than acquiring generalizable patterns. The elevated test loss values (e.g., 11.334 for GPT-2 (small)) further confirm that these models encounter challenges when processing unseen data, resulting in notable performance degradation. Several factors may contribute to this issue: 1) Insufficient pretraining on diverse sequences: GPT -2 models may excessively depend on specific token dependencies encountered during training. 2) Autoregressive nature: As a unidirectional model, GPT -2 faces difficulties in effectively capturing bidirectional dependencies, which can hinder performance on complex text generation tasks. 3) Limited data efficiency: In contrast to T5, which benefits from a denoising objective, GPT -2 models may necessitate more extensive fine-tuning to achieve improved generalization.
 - **T5:** Although T5 models demonstrate strong generalization capabilities, they still exhibit a decline in performance when evaluated on test data as compared to training data. For example, the T5 (Ex-large) model achieves notable training metrics (BLEU-1 = 0.736, rouge-L = 0.946), yet the corresponding test scores are lower (BLEU-1 = 0.709, rouge-L = 0.879).

This suggests that while the model does not exhibit extreme overfitting akin to GPT-2, some degree of overfitting persists. Potential contributing factors include: 1) Exposure bias: Exposure Bias refers to the disparity between training and inference phases in sequence generation models. Specifically, the model is trained using ground-truth previous tokens (through teacher forcing), but during inference, it relies on its own predicted tokens. This reliance can lead to a cascade of errors if initial predictions are inaccurate. Since T5 is an encoder-decoder transformer, its encoder employs a bidirectional approach; however, its decoder operates autoregressively, generating each token based on previously generated tokens. This mechanism can result in cumulative errors during the generation of longer sequences. 2) Data distribution shift: The training dataset may encapsulate patterns that do not entirely generalize to the test set, particularly evident in smaller datasets, which can lead to minor but discernible performance declines. 3) Over-reliance on training patterns: Despite the bidirectional nature of its encoder, T5 may develop biases toward frequently observed patterns within the training data, thereby compromising its adaptability to previously unseen variations.

Test the T5 models with external MIMIC-III dataset

To assess the generalizability of the three T5 models, a dataset comprising 184 history of present illness notes was randomly selected from the MIMIC-III database. This dataset is entirely unseen to the models, as they have not been trained on this particular data. This dataset includes 2,396 sentences, 42,522 tokens, and 865 distinct abbreviations. Two evaluation approaches were employed: first, the history of present illness notes were directly input into the model and subsequently evaluated (shown in Table 6 as MIMIC-III_{text}); second, sentence segmentation was applied to the history of present illness notes before inputting each sentence into the model for evaluation (shown in Table 6 as MIMIC-III_{sentence}). The results of this analysis are presented in Table 6.

Table 6. Evaluation of the T5 models with external MIMIC-III dataset ^a.

Dataset	Models	Evaluation				
		B-1	B-1	R-1	R-1	R-1
MIMIC-III _{text}	T5 (small)	0.317	0.194	0.357	0.166	0.348
MIMIC-III _{text}	T5 (large)	0.383	0.269	0.437	0.235	0.430

MIMIC-III _{text}	T5 (extra large)	0.388	0.273	0.439	0.238	0.431
MIMIC-III _{sentence}	T5 (small)	0.619	0.516	0.644	0.512	0.642
MIMIC-III _{sentence}	T5 (large)	0.620	0.526	0.659	0.522	0.658
MIMIC-III _{sentence}	T5 (extra large)	0.654	0.559	0.686	0.552	0.685

^a In the train and test columns, the designations “B-1”, “B-2”, “R-1”, “R-2”, and “R-L” correspond to BLEU-1, BLEU-2, ROUGE-1, ROUGE-2, and ROUGE-L, respectively. All scores presented are F1 scores.

The findings presented in Table 6 indicate that the generalizability of the models is enhanced by varying the size of the input sequence. A notable performance improvement is observed when employing MIMIC-III_{sentence} as opposed to MIMIC-III_{text} across all model sizes, including T5 small, large, and extra-large. This observation suggests that shorter input sequences, specifically sentences, facilitate more effective attention mechanisms. The limitation of the context window in transformer models mitigates information dilution and enhances focus on pertinent content. The consistent trend across different model sizes reinforces the notion that shorter contexts optimize performance by enabling more precise token-to-token dependencies within the attention mechanism.

Discussion

This study proposes a framework for identifying the properties associated with each task, followed by the selection of examples that exhibit these properties. Subsequently, these chosen examples are utilized by LLMs through two distinct approaches: an optimized example-based prompt, which is an engineered prompt, and the fine-tuning of LLMs. The findings of this study highlight certain advantages and align with specific drawbacks that will be discussed in the following sections.

Findings

1. **Basic prompt:** The comparative analysis between the basic prompt and example-based prompts reveals important insights into the effectiveness of prompting strategies with and without providing examples for clinical notes. The results demonstrate a clear advantage of example-based prompts over the basic prompt, as the basic prompt consistently yields lower performance across all other categories. This finding underscores the limitations of using the basic prompt and highlights the value of incorporating examples to enhance model performance.
2. **The example-based random selection:** The evaluation of random example selection demonstrates consistent improvements when transitioning from a 3-shots to a 5-shots approach. While the 3-shots random example selection yields superior performance compared to the basic prompt, it remains inferior to the 3-shots optimized example prompt. This relatively low performance can be attributed to the inherent randomness of the examples provided, as they do not introduce new knowledge for enhancing ChatGPT’s learning capacity. Notably, the 5-shots random example selection achieves performance levels that are comparable to those of the 3-shots optimized example selection.

It should be noteworthy that previous research has highlighted several challenges associated with incorporating examples as part of the prompt [53]. In response to these challenges, several approaches have been proposed. For instance, [54] and [55] leverage sentence embeddings to identify examples closely aligned with the input within the embedding space. Meanwhile, [56] explores the generalization capabilities of pre-trained language models by including both positive and negative examples. Despite these advancements, a critical question remains unresolved: How can we effectively determine which examples should be included in the prompt to achieve optimal results? The optimized example selection approach provided in this study increases the performance rather than random example selection, but the performance is still low.

3. **The example-based optimized selection:** The 5-shots optimized examples selection method is the most effective among all example-based prompts. The superior performance observed with this method suggests that carefully chosen examples significantly improve the model's performance across all the example-based methods. This optimized approach allows the model to understand better and generalize from the examples provided, leading to increasing performance. However, this model still performs less than fine-tuning LLMs with the T5 model.
4. **Domain-specific fine-tuning with GPT-2:** The GPT models have decoder-only architectures that use causal attention, meaning they only generate tokens based on past context [66]. This structure is unsuitable for applications like detecting abbreviations and deciphering, categorized as machine translation tasks [41, 59]. Therefore, all the models that are fine-tuned with GPT-2 models in this study are over-fitted, meaning that they have appropriate performance in the train set and low performance in the test set, which is unseen data. In this case, if the GPT models want to be chosen for translation tasks, they should be fine-tuned as instructional fine-tuning [57], which is prompt-based tuning or based on question answering, which is the same method that the current GPT models work.
5. **Domain-specific fine-tuning with T5:** The T5 model is characterized by its encoder-decoder architecture, functioning as a sequence-to-sequence model, which implies that both inputs and outputs are sequences of text. In contrast to GPT, which employs a decoder-only causal language modeling approach, T5 demonstrates superior effectiveness for tasks that necessitate comprehensive input understanding, including translation, summarization, and question answering [60]. T5 is pretrained utilizing a span corruption denoising objective; during this process, random spans of text are masked, and the model is trained to predict these spans. Conversely, GPT generates predictions for the next token sequentially without an explicit encoder. This fundamental distinction renders T5 more suitable for the specific task of abbreviation detection and its subsequent deciphering. The results in Error: Reference source not found and Table 5 reveal

a trade-off between performance metrics and loss values for T5 (large) and T5 (Ex-large). T5 (large) demonstrates superior BLEU and ROUGE scores on the test set, indicating better lexical and structural alignment with target outputs. However, T5 (Ex-large) achieves significantly lower training, validation, and test loss, suggesting a more stable learning process with reduced overfitting. While T5 (large) excels in producing fluent and coherent sequences, the lower loss of T5 (Ex-large) implies stronger generalization. These findings highlight the importance of selecting models based on specific task requirements, balancing generation quality and optimization stability. Additionally, the findings presented in Table 6 indicate that, except when the model is either fine-tuned or only minimally tuned, shorter input sequences, particularly sentences, lead to more effective attention mechanisms. The constraint of the context window in transformer models reduces information dilution and improves focus on relevant content. The consistent pattern observed across various model sizes supports the idea that shorter contexts enhance performance by allowing for more accurate token-to-token dependencies within the attention mechanism.

6. Comparison between example-based prompt and fine-tuning approaches:

The findings of this study indicate that the fine-tuning approach outperforms the example-based prompt method. However, Table 7 provides a detailed examination of the respective advantages and disadvantages associated with each approach.

Table 7. Comparison between example-based prompt and fine-tuning approach

Type	Advantages	Disadvantages
Example-based prompt	• There is no requirement for expertise in large language models or fine-tuning techniques. • Users need only to comprehend the specific task at hand. • The method is readily accessible to all.	• The study necessitated a substantial number of input tokens, totaling 35,179,649, which included additional examples. The output tokens were recorded at 2,475,600 ^a. • There were significant costs associated with the model's usage. • The performance in detecting and deciphering abbreviations was found to be suboptimal.
Fine-tuning	• Exceptional performance. • Reduced costs in comparison to prompt-based methodologies. In this study, for all six	• The performance of models is significantly influenced by domain-specific data collection. • There exists a risk of overfitting when training is extended for

	models, 44 GPU/hr is employed ^b . Facilitates reusability across various systems.	prolonged periods. Hyperparameter tuning is a requisite, which presents considerable challenges.
--	---	---

^a Based on the OpenAI, the unit price for GPT-4 for each input token is 0.00003 USD and for each output token is 0.00006 USD on March 5, 2025.

^b GPU model: GH200 (96 GB), which includes 64 CPU cores, 463.9 GB of RAM, and a 4.4 TB SSD. Based on Lambdalab price, which is used in this study, the price for this GPU is 1.49/hr USD.

In conclusion, Table 7 illustrates that the fine-tuning approach presents a lower cost and superior performance compared to the example-based prompt method. However, it is essential for users to possess technical knowledge of LLMs to effectively develop this methodology.

Limitations

This study has several limitations that should be considered.

- **Example dependency for example-based prompt:** The effectiveness of the example-based prompt is influenced by the quantity of examples utilized. This dependence on a larger number of examples to encompass all relevant properties may limit the capacity to ascertain the optimal number of examples necessary for achieving peak performance. Furthermore, this requirement is likely to increase costs, as the volume of input tokens will expand accordingly.
- **Dataset size and diversity:** Although the model has been evaluated across various clinical notes, it is essential to acknowledge that the dataset utilized was constrained. Further assessment with a more extensive and diverse dataset, encompassing a range of clinical note types, is required to ensure the model's consistent efficacy across different categories of clinical texts.
- **Abbreviation limitation:** This study detects abbreviations in various formats and deciphers them; however, individual special characters that can be interpreted as an abbreviation or are used as mathematical symbols, such as "+", can be interpreted as a mathematical symbol or can be an existing symptom for the patient, are excluded from this analysis. Therefore, the individual special characters are not included in this study.
- **Generalization across other LLMs:** Due to financial budget constraints, the framework was evaluated solely using GPT and T5. Future research endeavors should investigate its applicability and performance with additional large language models, such as LLaMA 2, LLaMA 3, and BART, in order to assess its generalization capabilities across a wider array of models.

Conclusions

In this study, a framework is proposed for detecting and deciphering abbreviations within clinical text, employing LLMs. This framework is structured into four distinct phases: task definition, properties identification, example selection, and the application of LLMs

through either a fine-tuning approach or an optimized example-based prompting method. The results demonstrate that the fine-tuning approach for LLMs yields superior performance at a lower cost compared to the optimized example-based prompt. This finding indicates that fine-tuning LLMs effectively and efficiently facilitates the detection and deciphering of abbreviations in clinical notes. Further research is warranted to fine-tune additional LLMs with larger and more diverse datasets of clinical notes, aiming to develop a generalized model for this specific task.

Acknowledgements

This paper is funded by the Canadian Institutes of Health Research (CIHR), Canada's federal funding agency for health research.

Conflicts of Interest

The authors declare no competing interests.

Abbreviations

AI: artificial intelligence

BLEU: bilingual evaluation understudy

CASI: clinical abbreviation sense inventory

EHR: electronic health record

EMR: electronic medical records

GPT: generative pre-trained transformer

LLM: large language model

LLaMA: large language model meta AI

ML: machine learning

MIMIC-III: medical information mart for intensive care version III

NLP: natural language processing

ROUGE: recall-oriented understudy for gisting evaluation

Appendix 1

In this section, the objective is to implement masking in the proposed model for preprocessing the clinical note. To commence the masking process, the initial questions presented in Table 1 should be addressed. Responding to each of these questions, a comprehensive engineered prompt will be constructed. Therefore, in this case the task is "text normalization" and the sub tasks are "finding abbreviations and their associated complete form". The response format is "JSON" and the example selection method is "one-shot or few-shot examples". During this process, the prompt is constructed as follows.

(R1) The task is the text normalization of narrative electronic medical record data in the below text. (R2) The below text is a history of present illness section in a discharge summary note, which is narrative electronic medical record data. (R3) providing a comprehensive account of the current illness, which is full of shortened words, abbreviations, acronyms, and special characters. (R4) The sub-tasks are finding abbreviations and their associated complete form. (R5) The response must be in JSON format. (R6) All examples related to the properties of all subtasks are identified as follows:

(R7) The examples are:

Example (1)

Example (2)

Example (3)

(R8) Please provide the response in JSON format only with the mentioned key and value. No subkeys are allowed. (R9) The history of present illness text that must be used is here: Target text.

Appendix 2.

This paper is accompanied by two datasets. The first dataset contains 1,292 clinical notes that have been randomly selected from the CASI dataset. The CASI dataset comprises abbreviations that share identical structures but pertain to different concepts. The attached dataset enhances the CASI dataset by incorporating additional abbreviations and their meanings as found within the clinical notes. The second dataset includes 184 history of present illness notes, also randomly selected, from the MIMIC-III dataset. In both datasets, the complete forms of each abbreviation are provided, in accordance with the guidelines outlined in Appendix 3.

Appendix 3

The guidelines for identifying abbreviations and converting them into their full forms are included in the attached document.

References

1. C. Cheliger, G. Wu, S. Lee, J. Pan, D. A. Southern, E. A. Martin, N. Sapiro, C. A. Eastwood, H. Quan, and Y. Xu, "Bert-based neural network for inpatient fall detection from electronic medical records: retrospective cohort study," *JMIR medical informatics*, vol. 12, p. e48995, 2024.
2. M. R. Cowie, J. I. Blomster, L. H. Curtis, S. Duclaux, I. Ford, F. Fritz, S. Goldman, S. Janmohamed, J. Kreuzer, M. Leenay, *et al.*, "Electronic health records to facilitate clinical research," *Clinical Research in Cardiology*, vol. 106, pp. 1–9, 2017.
3. S. R. Raman, L. H. Curtis, R. Temple, T. Andersson, J. Ezekowitz, I. Ford, S. James, K. Marsolo, P. Mirhaji, M. Rocca, *et al.*, "Leveraging electronic health records for clinical research," *American heart journal*, vol. 202, pp. 13–19, 2018.
4. J. Liu, D. Capurro, A. Nguyen, and K. Verspoor, "'note bloat' impacts deep learning-based nlp models for clinical prediction tasks," *Journal of biomedical informatics*, vol. 133, p. 104149, 2022.
5. C. Xiao, E. Choi, and J. Sun, "Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review," *Journal of the American Medical Informatics Association*, vol. 25, no. 10, pp. 1419–1428, 2018.

6. Y. Zhu, A. Mahale, K. Peters, L. Mathew, F. Giuste, B. Anderson, and M. D. Wang, "Using natural language processing on free-text clinical notes to identify patients with long-term covid effects," in *Proceedings of the 13th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*, pp. 1–9, 2022.
7. J. L. Schwartz, E. Tseng, N. M. Maruthur, and M. Rouhizadeh, "Identification of prediabetes discussions in unstructured clinical documentation: validation of a natural language processing algorithm," *JMIR Medical Informatics*, vol. 10, no. 2, p. e29803, 2022.
8. S. Raza and B. Schwartz, "Constructing a disease database and using natural language processing to capture and standardize free text clinical information," *Scientific Reports*, vol. 13, no. 1, p. 8591, 2023.
9. M. Humbert-Droz, Z. Izadi, G. Schmajuk, M. Gianfrancesco, M. C. Baker, J. Yazdany, and S. Tamang, "Development of a natural language processing system for extracting rheumatoid arthritis outcomes from clinical notes using the national rheumatology informatics system for effectiveness registry," *Arthritis Care & Research*, vol. 75, no. 3, pp. 608–615, 2023.
10. K. Xie, R. S. Gallagher, E. C. Conrad, C. O. Garrick, S. N. Baldassano, J. M. Bernabei, P. D. Galer, N. J. Ghosn, A. S. Greenblatt, T. Jennings, *et al.*, "Extracting seizure frequency from epilepsy clinic notes: a machine reading approach to natural language processing," *Journal of the American Medical Informatics Association*, vol. 29, no. 5, pp. 873–881, 2022.
11. T. Mehra, T. Wekhof, D. I. Keller, *et al.*, "Additional value from free-text diagnoses in electronic health records: Hybrid dictionary and machine learning classification study," *JMIR Medical Informatics*, vol. 12, no. 1, p. e49007, 2024.
12. A. Jaiswal, A. Katz, M. Nesca, and E. Milios, "Identifying risk factors associated with lower back pain in electronic medical record free text: deep learning approach using clinical note annotations," *JMIR Medical Informatics*, vol. 11, p. e45105, 2023.
13. E. H. Houssein, R. E. Mohamed, and A. A. Ali, "Heart disease risk factors detection from electronic health records using advanced nlp and deep learning techniques," *Scientific Reports*, vol. 13, no. 1, p. 7173, 2023.
14. A. G. Russo, A. Ciarlo, S. Ponticorvo, F. Di Salle, G. Tedeschi, and F. Esposito, "Explaining neural activity in human listeners with deep learning via natural language processing of narrative text," *Scientific Reports*, vol. 12, no. 1, p. 17838, 2022.
15. A. Borjali, M. Magnéli, D. Shin, H. Malchau, O. K. Muratoglu, and K. M. Varadarajan, "Natural language processing with deep learning for medical adverse event detection from free-text medical narratives: A case study of detecting total hip replacement dislocation," *Computers in biology and medicine*, vol. 129, p. 104140, 2021.
16. X. Yang, A. Chen, N. PourNejatian, H. C. Shin, K. E. Smith, C. Parisien, C. Compas, C. Martin, A. B. Costa, M. G. Flores, *et al.*, "A large language model for electronic health records," *NPJ Digital Medicine*, vol. 5, no. 1, p. 194, 2022.
17. D. Van Veen, C. Van Uden, L. Blankemeier, J.-B. Delbrouck, A. Aali, C. Bluethgen, A. Pareek, M. Polacin, W. Collins, N. Ahuja, *et al.*, "Clinical text summarization: Adapting large language models can outperform human experts," *arXiv preprint arXiv:2309.07430*, 2023.
18. K. Singhal, S. Azizi, T. Tu, S. S. Mahdavi, J. Wei, H. W. Chung, N. Scales, A. Tanwani, H. Cole-Lewis, S. Pfohl, *et al.*, "Large language models encode clinical knowledge," *arXiv preprint arXiv:2212.13138*, 2022.
19. S. M. Meystre, G. K. Savova, K. C. Kipper-Schuler, and J. F. Hurdle, "Extracting information from textual documents in the electronic health record: a review of recent research," *Yearbook of medical informatics*, vol. 17, no. 01, pp. 128–144, 2008.
20. A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean, "A guide to deep learning in healthcare," *Nature medicine*, vol. 25, no. 1, pp.

24–29, 2019.

21. R. Aggarwal, V. Sounderajah, G. Martin, D. S. Ting, A. Karthikesalingam, D. King, H. Ashrafian, and A. Darzi, “Diagnostic accuracy of deep learning in medical imaging: a systematic review and meta-analysis,” *NPJ digital medicine*, vol. 4, no. 1, p. 65, 2021.
22. A. J. Thirunavukarasu, D. S. J. Ting, K. Elangovan, L. Gutierrez, T. F. Tan, and D. S. W. Ting, “Large language models in medicine,” *Nature medicine*, pp. 1–11, 2023.
23. D. Anisuzzaman, J. G. Malins, P. A. Friedman, and Z. I. Attia, “Fine-tuning llms for specialized use cases,” *Mayo Clinic Proceedings: Digital Health*, 2024.
24. X. Zhang, N. Talukdar, S. Vemulapalli, S. Ahn, J. Wang, H. Meng, S. M. B. Murtaza, D. Leshchiner, A. A. Dave, D. F. Joseph, *et al.*, “Comparison of prompt engineering and fine-tuning strategies in large language models in the classification of clinical notes,” *AMIA Summits on Translational Science Proceedings*, vol. 2024, p. 478, 2024.
25. Q. Shi, K. Luzuriaga, J. J. Allison, A. Oztekin, J. M. Faro, J. L. Lee, N. Hafer, M. McManus, A. H. Zai, *et al.*, “Transforming informed consent generation using large language models: Mixed methods study,” *JMIR Medical Informatics*, vol. 13, no. 1, p. e68139, 2025.
26. W. S. Jang, S. Sultana, Z. Yao, H. Tran, Z. Yang, S. Kwon, and H. Yu, “Enhancing llms for identifying and prioritizing important medical jargons from electronic health record notes utilizing data augmentation,” *arXiv preprint arXiv:2502.16022*, 2025.
27. J. Howard and S. Ruder, “Universal language model fine-tuning for text classification,” *arXiv preprint arXiv:1801.06146*, 2018.
28. C. Friedman, “A broad-coverage natural language processing system,” in *Proceedings of the AMIA Symposium*, p. 270, American Medical Informatics Association, 2000.
29. C. Friedman, P. O. Alderson, J. H. Austin, J. J. Cimino, and S. B. Johnson, “A general natural-language text processor for clinical radiology,” *Journal of the American Medical Informatics Association*, vol. 1, no. 2, pp. 161–174, 1994.
30. A. R. Aronson, “Effective mapping of biomedical text to the umls metathesaurus: the metamap program,” in *Proceedings of the AMIA Symposium*, p. 17, American Medical Informatics Association, 2001.
31. A. R. Aronson and F.-M. Lang, “An overview of metamap: historical perspective and recent advances,” *Journal of the American Medical Informatics Association*, vol. 17, no. 3, pp. 229–236, 2010.
32. J. C. Denny, P. R. Irani, F. H. Wehbe, J. D. Smithers, and A. Spickard III, “The knowledgemap project: development of a concept-based medical school curriculum database,” in *AMIA Annual Symposium Proceedings*, vol. 2003, p. 195, American Medical Informatics Association, 2003.
33. K. B. Wagholikar, M. Torii, S. R. Jonnalagadda, and H. Liu, “Pooling annotated corpora for clinical concept extraction,” *Journal of Biomedical Semantics*, vol. 4, no. 1, pp. 1–10, 2013.
34. M. Jiang, Y. Wu, A. Shah, P. Priyanka, J. C. Denny, and H. Xu, “Extracting and standardizing medication information in clinical text—the medex-uima system,” *AMIA Summits on Translational Science Proceedings*, vol. 2014, p. 37, 2014.
35. O. Bodenreider, “The unified medical language system (umls): integrating biomedical terminology,” *Nucleic acids research*, vol. 32, no. suppl_1, pp. D267–D270, 2004.
36. “Snomed-ct.” <https://confluence.ihtsdotools.org/display/DOCGLOSS/SNOMED+CT>. Accessed: 2023-09-26.
37. “Rxnorm.” <https://www.nlm.nih.gov/research/umls/rxnorm/index.html>. Accessed: 2023-09-26.
38. “Clinical abbreviation sense inventory.” <https://conservancy.umn.edu/handle/11299/137703>. Accessed: 2023-09-17.

39. "Mimic-iii documentation." <https://mimic.mit.edu/docs/iii/>. Accessed: 2023-09-17.
40. V. Kumar, A. Stubbs, S. Shaw, and Ö. Uzuner, "Creation of a new longitudinal corpus of clinical narratives," *Journal of biomedical informatics*, vol. 58, pp. S6–S10, 2015.
41. A. Rajkomar, E. Loreaux, Y. Liu, J. Kemp, B. Li, M.-J. Chen, Y. Zhang, A. Mohiuddin, and J. Gottweis, "Deciphering clinical abbreviations with a privacy protecting machine learning system," *Nature Communications*, vol. 13, no. 1, p. 7456, 2022.
42. I. Li, M. Yasunaga, M. Y. Nuzumlalı, C. Caraballo, S. Mahajan, H. Krumholz, and D. Radev, "A neural topic-attention model for medical term abbreviation disambiguation," *arXiv preprint arXiv:1910.14076*, 2019.
43. S. Moon, S. Pakhomov, and G. B. Melton, "Automated disambiguation of acronyms and abbreviations in clinical texts: window and training size considerations," in *AMIA annual symposium proceedings*, vol. 2012, p. 1310, American Medical Informatics Association, 2012.
44. Y. Wu, J. C. Denny, S. Trent Rosenbloom, R. A. Miller, D. A. Giuse, L. Wang, C. Blanquicett, E. Soysal, J. Xu, and H. Xu, "A long journey to short abbreviations: developing an open-source framework for clinical abbreviation recognition and disambiguation (card)," *Journal of the American Medical Informatics Association*, vol. 24, no. e1, pp. e79–e86, 2017.
45. V. Joopudi, B. Dandala, and M. Devarakonda, "A convolutional route to abbreviation disambiguation in clinical text," *Journal of biomedical informatics*, vol. 86, pp. 71–78, 2018.
46. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
47. A. Jaber and P. Martínez, "Disambiguating clinical abbreviations using a one-fits-all classifier based on deep learning techniques," *Methods of Information in Medicine*, vol. 61, pp. e28–e34, 2022.
48. M. Cevik, S. Mohammad Jafari, M. Myers, and S. Yildirim, "Sequence labeling for disambiguating medical abbreviations," *Journal of Healthcare Informatics Research*, pp. 1–26, 2023.
49. F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel, "Language models as knowledge bases?," *arXiv preprint arXiv:1909.01066*, 2019.
50. L. Cui, Y. Wu, J. Liu, S. Yang, and Y. Zhang, "Template-based named entity recognition using bart," *arXiv preprint arXiv:2106.01760*, 2021.
51. R. Shin, C. H. Lin, S. Thomson, C. Chen, S. Roy, E. A. Platanios, A. Pauls, D. Klein, J. Eisner, and B. Van Durme, "Constrained language models yield few-shot semantic parsers," *arXiv preprint arXiv:2104.08768*, 2021.
52. Z. Jiang, F. F. Xu, J. Araki, and G. Neubig, "How can we know what language models know?," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 423–438, 2020.
53. Y. Lu, M. Bartolo, A. Moore, S. Riedel, and P. Stenetorp, "Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity," *arXiv preprint arXiv:2104.08786*, 2021.
54. T. Gao, A. Fisch, and D. Chen, "Making pre-trained language models better few-shot learners," *arXiv preprint arXiv:2012.15723*, 2020.
55. J. Liu, D. Shen, Y. Zhang, B. Dolan, L. Carin, and W. Chen, "What makes good in-context examples for gpt-3?," *arXiv preprint arXiv:2101.06804*, 2021.
56. S. Mishra, D. Khashabi, C. Baral, and H. Hajishirzi, "Natural instructions: Benchmarking generalization to new tasks from natural language instructions," *arXiv preprint arXiv:2104.08773*, pp. 839–849, 2021.
57. S. Raschka, *Build a Large Language Model (From Scratch)*. Simon and Schuster, 2024.
58. Y. Zeng, "Environment-based design (ebd): A methodology for transdisciplinary design," *Journal of Integrated Design and Process Science*, vol. 19, no. 1, pp. 5–24, 2015.

59. A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
60. C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.
61. “T5.” https://huggingface.co/docs/transformers/en/model_doc/t5. Accessed: 2025-03-05.
62. L. Prechelt, “Early stopping-but when?,” in *Neural Networks: Tricks of the trade*, pp. 55–69, Springer, 2002.
63. S. Gehrmann, Y. Deng, and A. M. Rush, “Bottom-up abstractive summarization,” *arXiv preprint arXiv:1808.10792*, 2018.
64. C.-Y. Lin, “Rouge: A package for automatic evaluation of summaries,” in *Text summarization branches out*, pp. 74–81, 2004.
65. K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a method for automatic evaluation of machine translation,” in *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.
66. W. Zhang *et al.*, “A residual lstm and seq2seq neural network based on gpt for rice q&a,” *Agriculture*, vol. 12, no. 6, p. 813, 2022.

Supplementary Files

Dataset MIMIC_III.

URL: <http://asset.jmir.pub/assets/a76ed3ed00fd041a2eeacab1dd74c19d.xlsx>