Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

MEDX-FE: A Robust LLM-Based Framework for Medical Feature Extraction with Matching Gate Hallucination Filtering

Manal Abumelha[1,2], Abdullah AL-Malaise AL-Ghamdi[1], Ayman G. Fayoumi[1] and Mahmoud Ragab[3]

[1] *Information Systems Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia*

[2] *Information Systems Department, Applied College at Khamis Mushait, King Khalid University, Abha 62521, Saudi Arabia*

[3] *Information Technology Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia*

mabumelha@kku.edu.sa, aalmalaise@kau.edu.sa, afayoumi@kau.edu.sa, mragab@kau.edu.sa

ARTICLE INFO

Received: 22 Dec 2024

Revised: 20 Feb 2025

Accepted: 28 Feb 2025

ABSTRACT

Licensed medical examination assessment is a resource-intensive task that requires significant expert involvement and time. Automating this process is both critical and challenging due to the need to ensure reliable evaluation while minimizing manual effort. In this work, we introduce MEDX-FE (Medical Exam Feature Extractor), a framework that leverages the semantic understanding capabilities of large language models (LLMs) to extract clinical features from patient notes written by medical students. Central to our framework is the Matching Gate, a novel three-stage validation module designed to mitigate LLM hallucinations and ensure that only text-grounded features are retained. MEDX-FE combines instruction fine-tuning with few-shot in-context learning to produce accurate extractions even under limited supervision. Experiments on the United States Medical Licensing Examination (USMLE) Step 2 Clinical Skills (CS) dataset show that MEDX-FE achieves a micro F1 score of 0.96 using the full training set and maintains strong performance (0.946) even when trained on as few dataset (10 examples per case). These results highlight the potential of integrating LLMs with lightweight validation modules to enable scalable and trustworthy assessment in medical education.

Keywords: USMLE, Medical Exam, Feature Extraction, Large Language Model, Mistral.

INTRODUCTION

Medical education institutes around the world conduct a series of standardized assessments to license their students to assure their ability to practice their knowledge based on high standards (Amaral and Norcini 2023). The United States conducts a three-step examination to license their physicians. These exams are owned by the Federation of State Medical Boards (FSMB) and the National Board of Medical Examiners (NBME) and are known as the United States Medical Licensing Examination (USMLE)[1]. One of these three-step exams is the Step-2 Clinical Skills exam, which assesses medical students' ability to recall clinical knowledge integrated with communication skills to provide

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

a comprehensive written medical note documenting patient history, symptoms, and expected diagnosis. In this exam, a set of patients, who are actually actors, present with specific symptoms to be described, then medical students need to examine the actors by taking history and highlighting the leading symptoms to justify the diagnosis.

Conducting the Step-2 Clinical Skills exam requires extensive and expensive preparation by medical education institutes. These preparations start by preparing the case scenarios and their associated symptoms, practicing with patient actors, and allocating time for physicians to manually assess the written patient notes (more than 100 physicians required to assess 30,000 written patient notes every year) (Sarker et al. 2019), in addition to other physical costs of conducting the exam itself. During the COVID-19 pandemic, which enforced some distance and communication restrictions leading to exam suspension, and with all cost considerations, the CS exam was discontinued in January 2021 (Tsichlis, Del Re, and Carmody 2021).

Many attempts have been conducted to reduce the cost by automating the assessment of written patient notes (Latifi et al. 2016; Yim et al. 2019; Sarker et al. 2019). However, due to the sensitivity of the exam information, this led to limited data sharing, which decreased the scale of community contribution to finding the best automated solution.

To address this shortage in data availability, in February 2022, the National Board of Medical Examiners (NBME) launched a Kaggle competition[2] with Step-2 CS exam data to push the boundary of automated assessment with recent techniques inspired by community contributions around the world.

The earlier solutions were based on n-gram and fuzzy logic (Latifi et al. 2016; Yim et al. 2019; Sarker et al. 2019), which demonstrated limited capability in capturing contextual meaning. Subsequently, solutions based on BERT architectures emerged.

One of these solutions was provided by Zhou et al. (J. Zhou et al. 2022), which used weakly supervised training of a BERT-based model relying on self-collected data along with USMLE data for medical entity extraction and matching. Another solution was provided by Xu et al. (Xu et al. 2023), which used DeBERTa as a BERT-based model along with pseudo labelling and masked language modelling. Lastly, Yaneva et al. (V. Yaneva et al. 2024) combined winner solutions from the Kaggle competition along with traditional solution provided by Sarker et al. (Sarker et al. 2019). Majority of earlier solutions relied on lightweight LLMs known as BERT-based pretrained models which adapt bidirectional encoder-decoder architecture. However, these approaches are still limited in generalizing and capturing unseen data.

Recognizing the advancement in the field of Large Language Models (Benítez et al. 2024) (autoregressive models) and their significant performance in many fields, especially Natural Language Processing, we utilize LLM with recent techniques to solve the patient note automation problem efficiently. We extend the capabilities of the Mistral NeMo model to develop our models (Dubey et al. 2024). The details of our contributions as follows:

- Development of MEDX-FE (Medical Exam Feature Extractor) Framework: We have developed MEDX-FE, a large language model trained on the USMLE Step-2 CS dataset using different training approaches using different dataset splits.
- Expansion of Annotated Data: Leveraging the capability of Mistral NeMo, we have increased the annotated data by training the model on annotated examples; the expanded dataset will be used to re-train the model.
- Demonstration the Efficiency of LLM Feature Extraction: Our research provides empirical evidence of Large Language Models' capability to perform efficient feature extraction by deeply understanding the meaning behind the context, which is essential for automated scoring.
- Implementation and Evaluation of Recent Training Approaches: We have employed recent approaches in training our LLM and applied different training settings to quantify the impact of each setting.
- Development of Matching Gat Three-Step Validation: We propose a novel Matching Gate module to validate extracted features, ensure the reliability of matched features, enhancing the overall accuracy and transparency of the automated assessment.
- Comprehensive Error Analysis: An in-depth error analysis was conducted for further potential enhancement, providing a roadmap for future improvement.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Furthermore, we provide a complete pipeline to automate the assessment, potentially adapted for similar examinations requiring the extraction of specific clinical features using the power of LLM combined with Matching Gate for valid results.

Our research proves that using large language model for feature extraction shows remarkable results, even with limited training dataset. The rest of the paper organized as follows: In Section (background), we will highlight the background of each aspect used in previous related research, followed by the task description in Section (task description). Then, we provide a detailed description of the MEDX-FE framework along with the Matching Gate validation and the output evaluation in Section (MEDX-FE (Medical Exam Feature Extractor) Framework). Lastly, we present the results and conclusion of this study in Sections (result and conclusion).

BACKGROUND

Automating the assessment of medical examination offers a transformative advancement in the medical education system. Utilizing cutting-edge technologies can produce unbiased, reliable, and objective assessment of medical knowledge (Margolis and Clauser 2020). This automation is multidisciplinary, combining three fields: computer science, linguistics, and medicine, to develop a system with language understanding capability to handle complex medical text. This background section represents the historical evolution of Automated Essay Scoring (AES), also known as Automated Writing Evaluation (AWE), and its integration with clinical context, followed by highlight the revolutionaries that influence the use of artificial intelligence and natural language processing in automate assessment field.

[2]*https://www.kaggle.com/nbme-score-clinical-patient-notes/*

Automated assessment initially intended to measure the correctness of given text according to specific evaluation metrics and based on appropriate features (C. Chen and Cheng 2008). The automation of text-based assessment attracted researcher attention since the 1960s, when the first automated essay scoring system was invented even before the emergence of desktop computers. This system, named Project Essay Grade (PEG), used linear regression and was utilized in mainframe computers (Page 1966). The early systems of automation relied on rule-based approaches to automate scoring, which required extensive manual work to cover all possible linguistics and analyses of sentences and words. Then came bag of words and n-grams where the sentences represented as sequences using statistical methods such as find the co-occurrence of words and their probability based on position in the sequence. In this way, more context could be captured and analysed (Leacock and Chodorow 2003). One of the traditional methods included the use of Term Frequency-Inverse Document Frequency (TF-IDF) to calculate the importance of words in documents. It was used in automated scoring by aggregating pre-graded documents and calculating the importance of each word by its occurrence (Miller 2003).

Machine Learning and Word Embeddings

With the advancement in artificial intelligence and machine learning, models can be trained on text with features annotated by humans. Most automated systems rely on machine learning either as regression or classification tasks. Regression tasks are used to score the text; Cummins et al. (Cummins, Zhang, and Briscoe 2016) trained models with multiple features such as word unigrams and bigrams, parts of speech, overall essay length, grammatical relationships, maximum word length, and sentence length. The model was then adapted to predict numerical scores for each essay. On the other hand, classification tasks are used to classify answers. Sakaguchi et al. (Sakaguchi, Heilman, and Madnani 2015) developed a support vector model to process responses using n-grams and Word2Vec to assess sentence similarity through cosine similarity.

Since the emergence of word embeddings, they have revolutionized fields that need to capture more semantics from text. Word embeddings represent words in a dense, continuous vector space, which is rich in context and computationally efficient. The vector space uses the principle of distributional semantics, which posits that words appearing in similar contexts tend to have similar meanings (Dong and Zhang 2016).

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Deep Learning and Transformer Models

Since the emergence of deep learning neural networks, the field of automated assessment, as well as other fields, has improved dramatically due to Deep Neural Networks' ability to mimic human brain function by applying deep computational layers with backpropagation mechanisms that allow the neural network to learn from errors without human intervention (Uto 2021). Recurrent neural networks were invented with the ability to keep memory for longer sentences by looping output back into input. Taghipour et al. (Taghipour and Ng 2016) used RNN to train models to learn the relationship between essays and their scores without any annotated features by using comprehensive word embedding and hierarchical structure. Then convolutional neural networks emerged with the ability to use fewer parameters and filter features based on local patterns. Dong et al. (Dong and Zhang 2016) used hierarchical CNN where the lower model processes sentence structure while the upper layer processes essays as sets of sentences, allowing the model to weigh different parts of the essay differently.

Recently, attention mechanisms applied to neural networks allow models to capture longer sentences and deep descriptive semantics because, the attention consists of encoder and decoder heads able to capture the characteristics of the most important words in sentences through multiple attention heads working in parallel to capture different text dependencies. These models, known as transformer models, which are able to train efficiently on large data and capture more context due to their structure. Since the emergence of transformers, many models have used this architecture and trained on very large corpora to capture knowledge that solve specific tasks efficiently (Vaswani, Shazeer, and Parmar 2017). Sung et al. (Sung, Dhamecha, and Mukhi 2019) used the BERT model and achieved state-of-the-art results with few examples.

Large Language Models

After the success of transformer-based models, this field became active, which led to the emergence of Large Language Models (LLMs) where the parameter size ranges in billions rather than in millions as in BERT-based models. With increased numbers of parameters and training data, these LLMs' capability to capture context from very large texts approaches human-like performance. As with any field using LLMs, the medical field has gained high attention from researchers, with many LLMs trained on biomedical data to perform various tasks serving the medical domain (Thirunavukarasu et al. 2023). The LLM tasks in medicine can be divided into two main categories (H. Zhou et al. 2023):

- **Discriminative Tasks:** These tasks don't require generating new text but require reasoning and understanding to classify, categorize, or extract data from text. The input required involves electronic health records, medical questions, medical scientific papers, and medical documents. The expected output can be information retrieval, relation and entity extraction, named entity recognition, question answering, and labelling.
- **Generative Tasks:** These tasks require models to generate new text, such as summarization, generation, and simplification. The input involves medical reports, hospitalization records, medical education texts, and patient education materials. The expected output includes summarized medical documents or reports, summarized educational content, simplified educational textbooks, simplified hospitalization records, and generated educational content either for medical students or for patients to understand their health conditions.

Our task belongs to Discriminative Tasks, where features need to be extracted from free-text patient notes based on different semantic representations, making text understanding mandatory prior to extraction. Since the emergence of LLMs and as an active research area, many methods and approaches have been conducted to maximize LLM performance. These approaches can be categorized into four steps (H. Zhou et al. 2023; Liu et al. 2024):

- **Pre-Training:** This step involves training base large language models from scratch using huge general or domain-specific datasets to train neural network, the objectives of training can be varying such as next token prediction, next sentence prediction, or masked language modelling. The more data used to train model the more context and semantics can be captured by model. Due to the high cost and complexity related with

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

training base model it is done by big companies involve team of research. Many pretrained open-source models are publicly available, while others are commercially available require use of their paid APIs.

- **Continual Pretraining:** A common practice to avoid the cost associated with new training. We can use knowledge saved in model weight and add the new knowledge by autoregressive continual pretraining using data related to the task with next token prediction objective in this way the model can keep both previous knowledge and knowledge from task data.
- **Fine-tuning:** To use the model knowledge and adapt it to perform specific task, we can Fine-tuning the model by training it to solve specific task using specific task dataset. This fine-tuning change the model learned weight to adapt to new task it can be done in three different ways: training use label data known as Supervised Fine-Tuning (SFT); training use instruction-response pair known as Instruction Fine-Tuning (IFT); and to avoid loss of earlier saved knowledge in parameter the model can train using only subset of model parameters this training known as Parameter-Efficient Fine-Tuning (PEFT). Xiao et al. (Xiao et al. 2024) proved that fine-tuned LLMs in automated essay scoring outperform few-shot and zero-shot approaches.
- **Prompting:** This is another effective way to utilize LLMs for different tasks without consuming computational power while utilizing knowledge inherited in LLM weights. This approach doesn't change the weights of the model rather than use the concept of In-Context Learning (ICL), here the prompt of task instructions provided to the model to fulfil the task based on these instructions these is known as zero-shot learning, or few examples of task can be provided along with instructions which is known as few-shot learning. Stah et al. (Stahl et al. 2024) explored the effect of various prompts on LLM results in automated essay scoring.

In this section, we discussed the evolution phases of automated assessments starting from early statistical techniques to modern Large Language Models.

Figure 1 shows the development milestones that have affected the field of Automated Essay Scoring (AES). Starting with pioneering work in automated assessment in the 1960s and progressing to the sophisticated application of Large Language Models in the 2020s that mimic human-like understanding, each milestone represents a significant advancement toward refining the precision and capabilities of automated assessments.

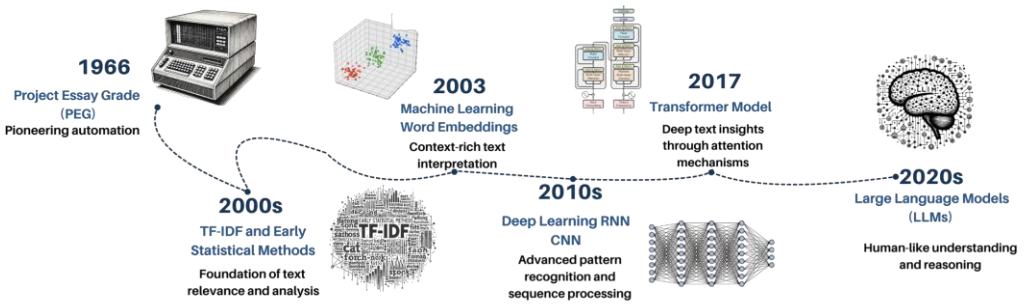

Figure 2 Historical evolution affecting Automated Essay Scoring (AES) Development.

Related Work

The assessment of medical written exams, such as the USMLE Step 2 CS, is a very specific domain task, and there are few works done in this field. Starting with a similar system developed in 2016 (Latifi et al. 2016), machine learning was used to automate the scoring of the Medical Council of Canada Qualifying Examination. They trained a machine learning classification model using decision-tree maximum likelihood estimation, which split the features and chose the split with specific features appearing in it.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Yim et al. (Yim et al. 2019) used simple feature extraction using n-gram followed by a BERT-based system for embedding the features and comparing rubrics in patient notes. In 2019, the Intelligent Clinical Text Evaluator (INCITE) was developed; this system relied on semi-supervised techniques that basically used traditional word statistics such as Levenshtein ratio and lexicon-based matching (Sarker et al. 2019).

Suen et al. Suen et. al. (Suen et al. 2023) used a BERT-based model with contrastive representation learning training to correlate similar text after embedding. This approach fits with extracting the correct answers of short answers by feeding the model with wrong and right examples. Zhou et. al. (J. Zhou et al. 2022) utilized the BERT-based model with weakly supervised and multi-level transfer learning where the patient note searches for exact features; if not found, it uses BioWordVec embedding similarity. The BioWordVec was designed for embedding medical words regardless of their position in the sentence, so it is considered static embedding.

The latest published work (V. Yaneva et al. 2024) merge notable Kaggle competition winner solutions in addition to the INCITE sequence module developed by Sarker et al. (Sarker et al. 2019), which is discussed in details:

- **INCITE system:** Sarker et al. developed the Intelligent Clinical Text Evaluator (INCITE) system to automate patient note assessment. It passe the patient notes through a series of modules to extract required features from text. These modules work sequentially; if any feature is extracted in one module, it will not be passed to the remaining modules, so these modules work as complementary to each other. These modules start with lexicon-based exact matching, where the lexicon of features is used to match the exact features appearing in text. Then, to capture more variant text including misspelled words, fuzzy matching using Levenshtein ratio measures the similarity between two strings. The Levenshtein ratio find similarity between two words by (insert, delete, or substitute) single character required to transform one word to the other. They also apply fixed and dynamic thresholds of Levenshtein ratio as high thresholds can capture shorter features and low thresholds can capture longer features. This method provides high performance with less computation, but it is impractical to build huge lexicons for more patient notes, as huge lexicons will increase the search space. This method is limited in its ability to capture semantics, especially for long sentences and negative phrases.
- **Kaggle competition approaches:** 1,471 teams participated in the Kaggle competition, contributing 28,049 code submissions. Many good insights and advanced techniques were used by top-ranked shared solutions. Most winners relied on transformer-based models which use attention mechanisms that can process longer sentences. These models are deep neural networks trained on massive text by masking some tokens to be predicted by the model; in this way, the model can capture the context and meaning from text. Bidirectional Encoder Representations from Transformers (BERT) follows the same transformer architecture but in a bidirectional way, so the word will be predicted based on right and left words. Since the emergence of BERT, many models have been developed based on BERT architecture, known as BERT-based models. One of these models is DeBERTa, which separates word content and position calculation to capture meaning and placement, which ultimately leads to deeper understanding of text. Most Kaggle teams used DeBERTa as their backbone model due to its performance. In addition to using DeBERTa, these are the highlighted techniques used by top winners (V. Yaneva et al. 2024):
    - *MLM training*: Utilize unannotated data to make the model more context-aware of the domain data. This is done through Domain Adaptive pretraining and task adaptive pretraining. These two approaches are unsupervised training using data belonging to the medical domain and/or task itself by masking some words to be predicted by the model. This training is known as masked language modelling (MLM) training. The primary goal of such training is to make the model more aware of domain and/or task semantic representation.
    - *Increase annotation:* In the Kaggle dataset, the annotated patient notes were limited to almost 1,000 compared to unlabelled annotated patient notes of almost 14,000. To take advantage of these unannotated data in the training process, pseudo labelling and meta pseudo labelling were used to create more annotated data. It is semi-supervised training where the model trains on annotated data to produce unannotated data to be used for further training.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

- *Multi-task learning:* Train the model to do two tasks jointly. First task, predict the feature. Second task assigning weight to the beginning and end of feature to be detected accurately.
- *Other techniques:* Various technique used to prevent overfitting such as dropout and cross-validation. Many teams ensemble different trained models then use knowledge distillation to provide smaller final models.

All these advanced solutions were produced before the realm of large language models (autoregressive models), so it is worth trying to solve the task with the power of large language models that can better generalize compared to BERT-based models (encoder-decoder model) (Yang et al. 2024).

TASK DESCRIPTION

USMLE Step 2 clinical skill exam was one of the exams required for licensing medical students in united states. This exam was designed to evaluate patient-focused clinical skills of medical students. In this exam, medical students rotate through different encounters with patients who are actually trained actors, each describing specific symptoms. The medical student has 15 minutes to communicate with the patient and conduct proper physical examination along with taking patient history. Then, in the last 10 minutes, the medical student must document the information in a patient note. Every patient note is rated by a physician rater, and if it is below the passing score, another rater is assigned to re-rate the patient note. The rating or scoring is based on the presence of specific features associated with each case; these features are decided before the exam by a committee of physicians (Sarker et al. 2019; Victoria Yaneva et al. 2022). The task to automate the assessment process is to determine whether these specific features are present or absent in the patient note.

Token-Level vs Binary Evaluation

All works have been done to solve this task, can be categorized as either token-level evaluation (Ganesh and Bansal 2023; B. Long, Tan, and Newman 2023; J. Zhou et al. 2022; Sarker et al. 2019). In token-level evaluation, which followed by the majority of previous works whether using neural networks or traditional NLP and fuzzy logic, the decision of features to be detected depends on the span. The span corresponds to a feature's character start and end position in the patient note. It is a pair or set of pairs representing the position of features, and these spans exist in ground truth to be compared with predicted spans using F1 score. The data is labelled as BIO (Beginning, Inside, Outside), which is widely used in part-of-speech tagging tasks (Ramshaw and Marcus 1999).

The evaluation process categorizes features as follows: if a feature is within both a ground-truth and a predicted span, it's considered a true positive (TP); if it's within a ground-truth but not a predicted span, it's a false negative (FN); and if it's within a predicted span but not a ground truth, it's a false positive (FP). These categorizations are then aggregated to calculate the F1 score. On the other hand, binary evaluation is a straightforward high-level evaluation that only decides if a feature is present or not in the patient note without much reliance on the precise location in the text, which is practically what real physicians do. They set a threshold of confidence equal to 50% to decide if the feature present in patient note.

In our approach, we will follow binary evaluation. Despite the power of LLM's to deep understand the context and recognize features expressed in various ways, they can't be trained to predict the exact span of the text. This limitation makes binary evaluation the optimal choice, allowing for broader generalizability and scalability which aligns with practical evaluation conducted by physicians. In our work, to calculate the F1 score, semantic matching must be done to evaluate the similarity between ground-truth features and predicted features. Semantic similarity and evaluation are covered in detail in section (error analysis).

Problem Notation

To address automated assessment by extracting medical features from patient notes, we systematically represent the key elements of this problem using formal notation. This notation allows us to identify the components of our system as a foundation for discussing the extraction and evaluation process.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Given:

- $C = \{c_1, c_2, c_3, \dots, c_i\}$ as set of cases represent specific medical problem described by patient. Each case associated with:
  - Set of features $F_{ik} = \{f_{1k}, f_{2k}, f_{3k}, \dots, f_{i2}\}$ these features need to be found in each patient note. i represent case number and k is index of features.
  - Set of patient notes $P_{ij} = \{p_{1j}, p_{2j}, p_{3j}, \dots, p_{i2}\}$ written by medical student. i represent case number and j is index of patient note.

So, we have Automated assessment function $A: P^i \times F^i \rightarrow \{0,1\}^{j \times k}$

$$A(p_{ij}, f_{ik}) = \begin{cases} 1 & if\ feature\ f_{ik}\ present\ in\ p_{ij}. \\ 0 & otherwise. \end{cases} , for\ k = 1,2, \dots, K$$

Objective: for each $p_{ij} \in P_i$ associated with case C_i:

- Find all the features associated with this case $f_{ik} = \{f_{1k}, f_{2k}, f_{3j}, \dots, f_{i2}\}$, and evaluate them $A(p_{ij}, f_{ik})$.
- Compile the results into a binary vector of length k.

Dataset

The data were collected from 2017 to 2020 from 35,156 medical students at testing locations in the United States. The dataset consists of 43,985 patient notes, with each note belonging to one of ten clinical cases. The annotated patient notes, where features and spans are specified for model training, comprised approximately 2,840 notes (284 patient notes per case), as illustrated in Figure 3.

Ten expert annotators were divided into five pairs to annotate the features in patient notes. To ensure quality 29 notes for each case were double-annotated, and F1 agreement for those double annotation was calculated to determine the feature spans within each case. The overall F1 agreement score across all cases was 48%. Additionally, the binary feature detection agreement, which determines whether a feature exists or not, was calculated across all cases and reached 97%. More information about the data annotation process is presented in (Victoria Yaneva et al. 2022).

Patient Note

HI (1) 35 yo F (2) (3) heavy uternine bleeding. Last normal period w (4) 6 month ago.
(6) LMP w (3) heavy bleedin (6) months ago. No clots.
Changes tampon every few hours, previously 4/day. Menarche at 12.
Minimal crampin (9) Never been pregnent, tried previously but unsuccesful.
Two adopted children and fe (7) fatigued at home.
Last PAP 6 months ago was normal, never abnormal.
(5) Gained 10-15 lbs past few months, eating out more though.
Hyperpigmented spots on hands and LT neck that she noticed 1-2 years ago.
SH: state social worker; no smoking or drug use; beer or two on weekends;
sexually active with boyfriend of 14 months, us (8) condoms at first b (8) no longer uses them.

Features assosiated with the case		Matched features and span from patient note	
(1)	35-year	['35 yo']	['5 10']
(2)	Female	['F']	['11 12']
(3)	heavy-periods-OR-irregular-periods	['heavy uternine bleeding', 'heavy bleeding']	['18 41', '88 102']
(4)	symptoms-for-6-months	['6 month']	['66 73']
(5)	Weight-Gain	['Gained 10-15 lbs']	['365 381']
(6)	Last-menstrual-period-2-months-ago	['LMP 2 months ago']	['79 82;103 115']
(7)	Fatigue	['fatigued']	['297 305']
(8)	Unprotected-Sex	['condoms no longer uses']	['746 753;767 781']
(9)	Infertility-HX-OR-Infertility-history	['Never been pregnent, tried previously but unsuccesful']	['211 264']

Figure 3 Annotated patient note sample illustrating features mapping with associated case-feature (case 7) and corresponding span positions.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The number of features associated with each case varies, as shown in Figure 4.

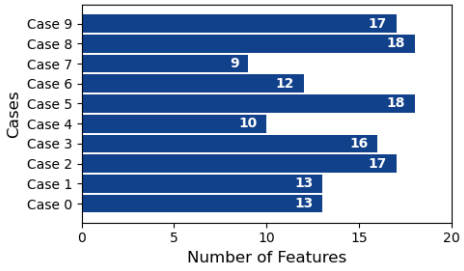

Figure 4 Number of features associated with each case.

To find easy and difficult features captured by medical students, we combined the annotated features for all patient notes that have specific features and detect the most empty and non-empty features, so if a feature is easy to capture by students, such as gender and age, it appears in all patient notes (284 times). Some features can be represented in limited synonyms such as (gender, age, vomiting, fever), while other features can be represented using a wide range of synonyms. We combined features from annotated patient notes, then dropped the duplicated features and kept unique representations of features, as shown in Figure 5 shows the feature with more than 200 synonyms or different representations of features which make this feature more difficult to capture, some feature captured more than one time in same patient note for those who have more than 284 unique representations.

From students' perspective some features captured by most of students such as gender and age, while other features missed by most of students, Figure 6 shows the features with empty annotation greater than 200, empty annotations means it is not written by medical students in patient note.

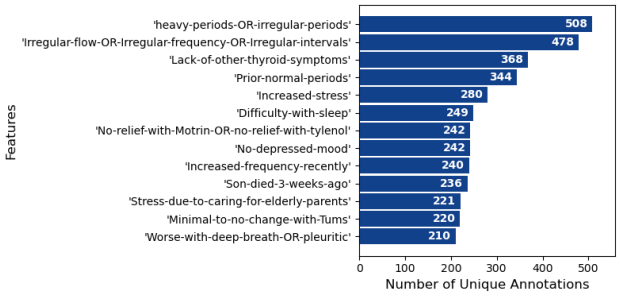

Figure 5 The features and their unique representation in patient note (how many synonyms can represent the feature?)

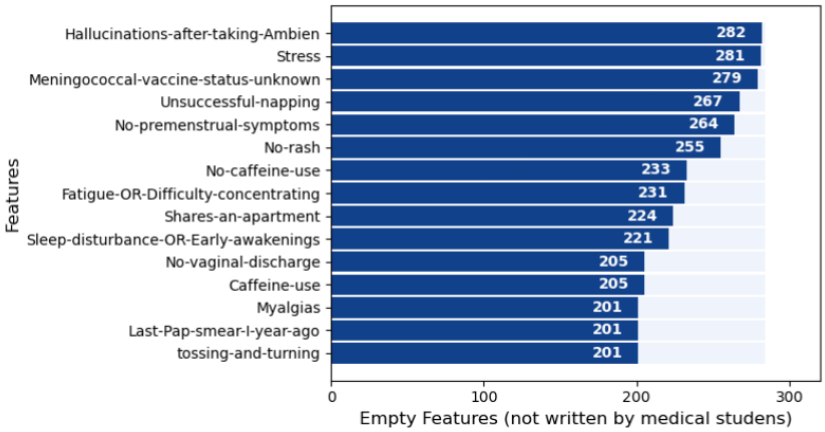

Figure 6 Number of non-exist features (how many students miss this feature?).

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

There are two data splits known as public and private. The total number of annotated patient notes is 2,839, almost 284 notes per case. In the public dataset split, as was the case in the Kaggle competition, the total number of annotated patient notes available publicly was only 1,000 (100 patient notes per case). These 1,000 records were used to develop and test models before final submission. After final submission, the remaining 184 patient notes per case (total 1,839 patient notes) were used to test models to determine the competition leaderboard. In the private split, all 2,839 notes can be used for training and testing. We follow both divisions for fair comparison with previous work in the competition and in this work (Yang et al. 2024).

Dataset preprocess

Raw text data can't directly handle by LLM models, it must be preprocess and transform into a format that can be understood by LLMs. In this paper, we apply minimal text preprocessing for a more generalized solution, utilizing LLMs' ability to handle case sensitivity, abbreviations effectively, and other common preprocess steps. Below are the steps involved in preparing the dataset:

1. **Merge Data:**
   Data is represented using different files: the features and their spans in a train file, and the corresponding full patient note text in a patient note file. The preprocessing involves aggregating all associated features for each patient note into a single array of features.
   Then, these features are merged with the corresponding patient note text. This integration step creates a comprehensive representation of each patient note. After merging and combining, the train file was converted from 40,596 rows—each row representing a single feature—to 2,839 rows, with each row representing an entire patient note and its features. To enhance the context and properly guide the model, we appended reference features associated with each case, along with the case number, into a single file.
2. **Data Templating:**
   For fine-tuning, we need to construct the data in a specific way by including the instruction and response as pair for each data entry. The instruction clarifies how the LLM should approach the data for accurate feature extraction. The response is used to demonstrate the desired output from the LLM; it is a dictionary of key-value pairs where the key is the feature number, and the value is the feature itself. Instructions and responses pair aggregated with remaining contextual information such as patient note text, reference features associated with the case, and the case number. This structure of data allows LLM to easily finetuned and follow the instructions associated with all necessary data to understand the task and feature context.
3. **Tokenize Data:**
   Text data to be used in any neural model must be represented as numerical values. This conversion knows as tokenization, each LLMs has its own tokenization approach. Deploying textual data into model-specific numerical tokens is an essential first step. We have developed a custom tokenization function to iterate through each data entry and construct a unified string representation. Each string entry contains instructions, inputs (patient history and reference features), and expected responses (set of annotated features). The contents of each data entry separated by special markers (e.g., ###patient_history:), to clarify boundaries between different components of the input. At the end of each data entry, end-of-sequence (EOS) token is appended, which help the model separate each data entry.

MEDX-FE (MEDICAL EXAM FEATURE EXTRACTOR) FRAMEWORK

The task of extracting features from a patient note requires a sophisticated level of semantic comprehension. A large language model has the ability to capture complex semantic meanings; utilizing this ability to perform this task inspired us to develop the MEDX-FE (Medical Exam Feature Extractor) framework Figure 7. This framework is a novel approach that leverages the strengths of LLMs for precise feature extraction from medical texts. MEDX-FE consists of two main phases: training and inference. In the training phase, the model's weights are changed to fit this task. The inference phase involves guiding the model to create the required output by providing examples of tasks and then validating the results using Matching Gate Validation steps. This integrated approach combines the

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

semantic understanding of LLMs with a robust validation mechanism, potentially improve medical text analysis. The detailed processes described in the subsequent sections.

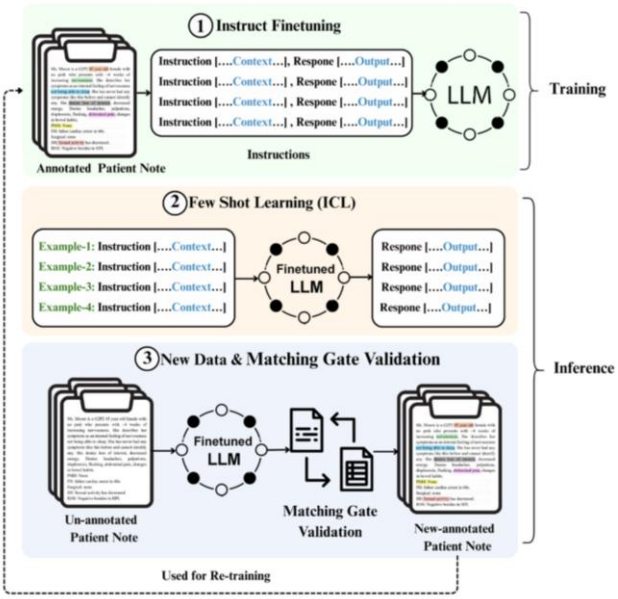

Figure 7 MEDX-FE (Medical Exam Feature Extractor) Framework.

Phase-1: Training

Since the emergence of large language models, this area has become an active field of research, and many efforts have been made to produce open-source models for the public to further research. These LLM models are known as foundation models, and they require high computational power and rigorous data preparation. Many researchers use these foundation models and build their solutions on top of them with additional enhancements (Kamath et al. 2024; Liu et al. 2024; Thirunavukarasu et al. 2023). To utilize the knowledge captured by LLMs and tune it to specific tasks, open-source LLMs can be fine-tuned to learn the knowledge required to complete specific task, by modifying the model weight to capture the knowledge of task in addition to previous knowledge of base model.

During instruction fine-tuning, the model is train with the objective of minimize the cross-entropy loss between the predicted token probability and the input token probability. Formally, the probability of generating the token y_t at time step t is defined as:

$$P_\theta(y_t|y < t, x)$$

where:

- x is the input: (instruction, patient note, and features associated with case).
- $y < t$ represent the previously generated sequence,
- θ is Mistral model trainable parameters.

The training objective is to reduce cross entropy loss of target output sequence of length

$$\mathcal{L}_{CE} = -\sum_{t=1}^{T} \log P_\theta\,(y_t|y < t, x)$$

Minimizing this loss encourage the model to learn and provide the better output align with input. The ability of LLMs to handle specific tasks better than others depends on many factors including, the initial training objectives, the

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

quality and variety of data used for training, and the architecture of the model itself. The Large Language Model Selection section highlights the procedure followed to select the best LLM to perform the feature extraction task.

Large Language Model Selection

Determining the most suitable LLM for our task requires running a small experiment using 200 data samples (20 per case) and following the MEDX-FE (Medical Exam Feature Extractor) framework. To maintain simplicity, the training phase is excluded from the framework. This experiment provides insight to see which LLM has the potential knowledge to provide the best feature extraction. The evaluation metric used is the micro F1 score. We use general and medical open-source large language models with a parameter size that less than 20 billion; the results are presented in the table below:

Table 1 Comparing the performance of general and medical language models using few-shot learning

Category	**Model**	**F1 Score**	**Time/ hour**
Llama General Models (Dubey et al. 2024)	Llama-3.1 8B	0.783	1:29:24
	Llama-3.1 8B Instruct	0.886	**0:57:39**
	Llama-3 8B	0.787	2:20:47
	Llama-3 8B Instruct	0.872	1:24:44
Mistral General Models (Mistral 2024)	Mistral NeMo 12B	0.818	2:33:18
	Mistral NeMo Instruct 12B	**0.932**	1:14:08
	Mistral-7b-instruct-v0.3	0.805	1:04:13
Medical Models	Meditron-3 8B (Z. Chen et al. 2023)	0.764	3:38:11
	BioMistral 7B (Labrak et al. 2024)	0.744	1:40:59
	Medical Llama3 8B (Vsevolodovna 2024)	0.825	2:50:01
	BioMedical Llama-3 8B (ContactDoctor 2024)	0.826	1:10:53

We focus in our experiment on two important factors. First is the computational efficiency, represented as the time required to process 200 samples. Second is task-specific performance, represented as F1 score. We chose the two most state-of-the-art open-source models (Hou and Lian 2024): Llama and Mistral. We also tried 4 medical domain LLMs (Meditron, BioMistral, Medical Llama3, and BioMedical Llama), which are pretrained on medical data.

General foundation models come with different versions, either in parameter size or training objectives. In our experiment, we evaluated two versions of Llama model, Llama 3 and Llama 3.1, with parameter sizes of 8 billion and 7 billion respectively. We avoided 70 billion parameter models due to their higher computational demand.

Many open-source LLMs introduce an "instruct" version of their base model. The instruct version is derived from the base version by finetuning on instruction datasets. Instruction finetuning approach shows improvement in performance and increases the ability of large language models to follow instructions. Chung et al. (Chung et al. 2024) prove that using multitask instructions data increases the model's generalization to handle different tasks. This is the case with Llama and Mistral, as both provide instruct versions of their base models to better follow human instructions, showing better performance on different benchmarks compared to base models, as corroborated by our experimental findings (Touvron et al. 2023; Jiang et al. 2023).

Our experiments show that Mistral Nemo 12B achieves superior performance, with an F1 score of 84%, and the ability to process two patient notes per minute, which is comparable to Llama-3.1 8B instruct that can process three and a half patient notes per minute with an F1 score of 79%. Considering the choice between these two models, we prioritized performance over time because the time difference was almost 17 minutes with performance improved by 5%.

Interestingly, medical LLMs which are further trained on biomedical data show lower performance. Transfer learning of medical domain to these general LLMs by adapting continual pretraining of general LLMs over biomedical data

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

can show improvement in performance over tasks related to the same medical domain data, such as question-answering (Z. Chen et al. 2023; Labrak et al. 2024). Due to the complexity and nature of medical text, transfer learning can have negative impact due to domain shift and catastrophic forgetting. Catastrophic forgetting is a common problem in many LLMs where the oldest learned weights from initial training are lost after further training on other complex tasks; considerable research efforts have been conducted towards mitigating this effect (Haizhou et al. 2024; Tobias 2024).

Mistral NeMo Instruct

In the late of 2023, Mistral team release Mistral 7B large language model that outperformed existing LLMs. The Mistral model added the inference cost in 2-D scaling law, which already considered: model capabilities cost and training cost, so it turns to be 3-D scaling law. Two main contributions that make the model cost-effective in inference are Grouped-Query Attention (GQA) and Sliding Window Attention (SWA).

GQA reduces computational cost by batching multiple queries as groups before processing. This batching allows fewer attention calculations, which indeed reduces the memory and computational cost and allows for processing larger batches of data at once. SWA localizes attention of tokens by considering a fixed-size window rather than the entire sequence, so each token computes attention only within its local window. SWA not only reduces the cost but also increases the model's ability to handle longer sequences.

Through the implementation of these two techniques, the complexity of the model is reduced from $O(n^2)$ to $O\left(\frac{nw}{g}\right)$, where input query is replaced with patch of queries g, and entire sequence is replaced with a window of sequence w. These reductions allow efficient inference and enhanced performance with reduced cost.

In mid of 2024, the Mistral AI team released Mistral NeMo 12B. The notable improvement in this model compared to previous version is larger context window (128k token). Mistral NeMo used quantization-aware training (QAT), which reduce the precision data during training and preserve the model performance.

For tokenization, Mistral NeMo introduces "Tekken," which is based on Tiktoken. Tiktoken, trained on over 100 languages and used in OpenAI models. It is 30% more efficient for text compression. Mistral collaborates with NVIDIA by using NeMo (Neural Modules) Framework, which is a platform designed to create, customize, and deploy new generative AI models, with advanced features such as Parameter-Efficient Fine-Tuning and Model Alignment (Gerald et al. 2024; Mistral 2024).

Instruct Finetuning

Early language models such as BERT-based models must be trained on labelled data of specific tasks to perform those tasks. This is known as supervised fine-tuning (T. Zhang et al. 2020). Then came the era where models could follow instructions written by humans as natural language (Avia and Omer 2020; Mishra et al. 2021).

Following this era, the approach used the base model with further finetuning on instruction data, known as instruction tuning or instruct finetuning, where the large language model is trained on pairs of instructions and responses. This approach showed better alignment with user requirements (Yuntao et al. 2022; O. Long et al. 2022).

Many LLMs are trained to perform multitasking by using instruction tuning. This approach improves model generalization and the ability to follow instructions (Chunting et al. 2023; Chung et al. 2022). Significant efforts have been made to produce a variety of instruction datasets to enable LLMs to generalize to new tasks. Many open-source LLMs introduce instruct versions of their base models by finetuning on these instruction data, with examples such as P3, Flan 2021, and LIMA (Shengyu et al. 2023).

Each LLM has its own instruction template to follow when implementing instruct finetuning. This template depends on how the LLM was originally instruct finetuned and the nature of the task. The template differs from one LLM to another. As we will use Mistral NeMo, we will follow their template to prepare data before it is tokenized and used for training. This template consists of instructions, input, and output. For our task, instructions are a set of guidelines

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

that direct the model how to extract features from text. The input consists of two parts: patient notes and features associated with the case. The output is the annotated features by experts.

Parameter Efficient Finetuning

Despite the evolution of large language models to solve many tasks, the nature of LLMs and their billion-size parameters make finetuning these large models with their full parameter size an obstacle in terms of time and computational cost. Deploying and using these huge models requires high-cost computation, so research has turned attention to deploying smaller models with comparable performance.

Parameter-efficient finetuning (PEFT) is a set of approaches used to reduce the cost of LLM training while maintaining high performance (Eric 2020; Clifton et al. 2023). This reduction in cost is associated with reduction in parameters and can be done from different perspectives including addition-based, specification-based, and reparameterization-based approaches. The addition-based approach introduces additional components or layers to improve performance or reduce size (Eric 2020). The specification-based approach finetunes models on a selectively limited set of parameters while keeping the rest unchanged, allowing targeted adjustments for new tasks (Ben Zaken, Shauli, and Yoav 2021). In reparameterization-based approaches, the parameters transfer into more parameter-efficient forms, such as using low-dimensional subspace to represent the parameters (Tianci, Ziqi, and Heng 2023).

One reparameterization-based PEFT approach is Low Rank Adaptation, or LoRA. LoRA freezes the pretrained model weights and uses low dimension decomposed extra weights to train the model. The smaller the decomposition ranks, the fewer parameters used for training (Hu et al. 2021).

Model quantization is also a PEFT approach. It is a compression technique that maps high-precision values of weights and activations of LLMs into low-precision format, for example, converting data representation from FP32 precision to 16-bit, 8-bit, or 4-bit. Quantization can reduce the memory and computation required but may sacrifice some accuracy. Quantization can be applied in different phases of developing LLMs, either during training, known as quantization-aware training (QAT), or after training, known as post-training quantization (PTQ) (Amir et al. 2021).

Quantized low rank adapter (QLoRA) combines the LoRA approach with model quantization by freezing the pretrained model weights and using low dimension decomposed and quantized extra weights to train the model. QLoRA allows LLMs to be trained and deployed on a single GPU. In our model, we use QLoRA for Mistral NeMo, which is provided by the Unsloth bitsandbytes library[3].

Phase-2: Inference

In this phase, the learned weights of the model will not change. The inference guides the model performance to meet the requirements of this task. The inference here can be divided into two parts. The first one (few-shot in-context learning) relates to guiding the model with few examples, and the second (Matching Gate validation) is a module to prevent hallucinations caused by LLMs.

- **Few-shot in-context learning (ICL)**
  To utilize the knowledge held by LLMs without changing the model weights, a few examples of the task can be shown as in-context learning (ICL) (Brown et al. 2020). In this way, the LLM model can be instructed using natural language prompts and a set of examples (few-shot) or even a single example (one-shot) pair as instructions to perform the task, with in-context examples as responses.
  Combining ICL with instruct finetuning can improve performance (T. Chen et al. 2024). In our task, we use few-shot in-context learning by providing the model with 4 examples from each case. This approach allows the model to see different examples of the required task before execution, helping it to better adapt to the output. In our model, we combine the power of instruct finetuning with ICL to achieve better performance for this specific task.

[3] https://huggingface.co/unsloth

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

- **Matching Gate Validation**

Despite the power of LLMs and their ability to solve many tasks in different domains, the problem of hallucinations persists. LLM hallucinations occur when unfaithful or false information is generated (Ji et al. 2022).

LLM hallucinations can take various forms, ranging from minor factual incorrectness, which is less harmful, to moderate distortions such as blending related facts, to major fabrications where the model creates unrealistic or irrational content. LLM hallucinations can fall under one of these four categories: contextual disconnection where the output mismatches the context of input, factual inaccuracies when the model produces misleading or false information, semantic distortion where the model misrepresents the underlying meaning of the input, and content hallucination where the model generates absent features from the input (Sahoo et al. 2024).

These hallucinations in medical field applications can lead to serious consequences that must be addressed. In our model, applying LLMs to extract features from patient notes can result in two types of issues: first, content hallucination where the model extracts non-existent features from patient notes; second, terminology variation, which is not explicitly considered a hallucination but rather a case where the semantics match but are not exactly represented in text (for example, the output "women" versus the written "female").

Understanding the root cause of hallucinations is beyond the scope of this research, but the issue can be related to the data, training, or inference phase. This field has been actively explored by many researchers, with significant effort devoted to mitigating these limitations (Tonmoy et al. 2024). In our case, the complex nature of medical terminologies, along with different synonyms, context-dependent meanings of features, and abbreviations represented in the text, makes it challenging for LLMs. Another limitation is overfitting, which is common when the dataset is very small, as in our case.

For our model, we tried different techniques to prevent hallucinations. Starting from training the model, we tried different instructions to emphasize the accuracy of features extracted. We also used the same instructions for few-shot in-context learning after training. These two modifications increased performance by almost 10%. After prediction, we match the predicted features with the existing features in patient notes. The key challenge is matching semantic features when they are represented differently. Semantic matching is core to our task and is fundamental to many natural language processing tasks such as information retrieval and question-answering systems.

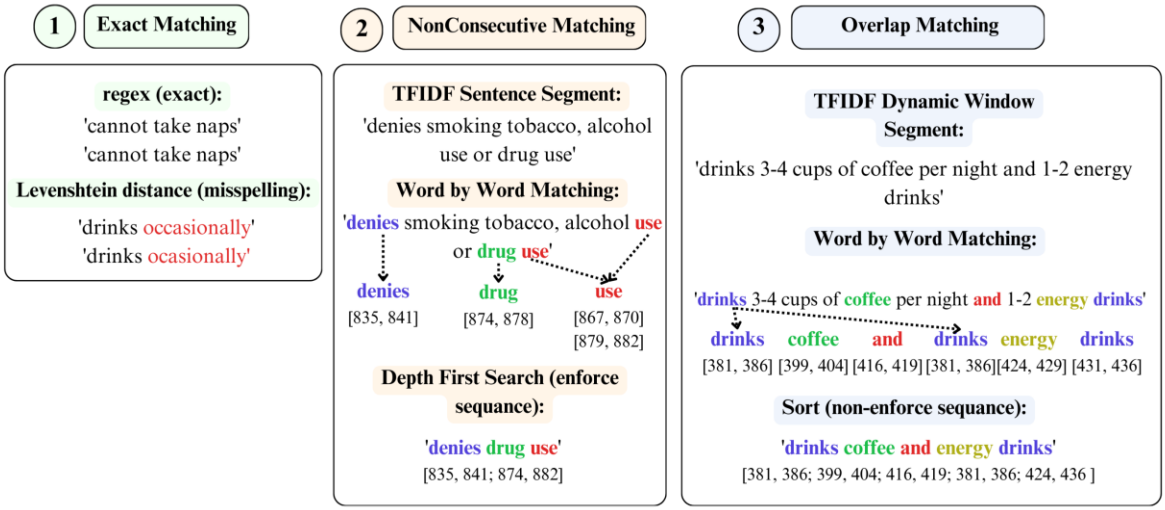

Figure 8 Matching Gate: three validation steps with examples.

In our approach, we rely on the power of LLMs to extract features, then apply a combination of matching approaches which rely on a hybrid of statistical term-weighting and fuzzy string-matching methods. This

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

comprehensive combination was built gradually as a result of iterative experiments to cover most matching possibilities that appear in feature extraction. This Matching Gate module consists of 3 main sub modules as shown in Figure 8, The three steps illustrated below:

- *Exact and Approximate String Matching:* For simplicity, as most of the predicted features exist exactly in the patient note, we use regular expressions (regex) built into Python modules to match the predicted feature with patient note text, then locate the span accordingly. Exact match using regex span detection:

$$Match_{exact}(f, T) = True\ if\ f \in T$$

As is common practice with LLMs, the predicted feature is free of misspelled words, which is not the case in real patient notes. To avoid this problem, we use the Levenshtein distance algorithm, which calculates the required edits in text (delete, insert, substitute) to transform it to other text, with the threshold set to 76%. Approximate match using Levenshtein similarity:

$$Sim_{lev}(f, s_i) = 1 - \frac{EditDistance\ (f, s_i)}{\max(|f|, |s_i|)} \geq \tau_1$$

where $s_i \in T$is a segment of the note and $\tau_1 = 0.76$ is a threshold empirically determined. Sometimes it is required to match nonconsecutive words separated by non-predicted words; in this case, exact matching and Levenshtein distance can't find the match. For this reason, we need to separate the patient note text into segments as described in steps 2 and 3.

- *Non-Consecutive Matching:* For non-consecutive words matching, we need to narrow the patient note text to easily capture the non-consecutive words in a single segment. In this approach, we apply sentence-level segmentation by dividing the patient note text into segments based on full stops and some recurrent patterns noticed in the patient notes, including new lines and some predefined sections (FH:, Med:, ROS:). Then, we need to find the most similar segment to the predicted feature. We use Term Frequency–Inverse Document Frequency (TF-IDF), which was originally invented for document classification, document ranking, information retrieval, and keyword extraction. It simply multiplies Term Frequency (TF) (how frequently a term appears in a corpus) by Inverse Document Frequency (IDF) (how important a term is in a corpus). Based on different experiments, TF-IDF shows the most effective way to bring the most similar sentence, which is why it is used for the third step as well. Let V_f and $V_s$be the TF-IDF vectors of the predicted feature and sentence segment respectively:

$$sim_{TFIDF}(f, s) = \frac{V_f \,.\, V_s}{\|V_f\| \,.\, \|V_s\|}$$

Then, we match every word in the predicted feature with every word in the patient note using Levenshtein distance with 70% as the threshold. A dictionary of matched words, similarity scores, and corresponding spans is passed to depth-first search to filter out the matched words by prioritizing those with the highest similarity scores and shortest paths between spans, as illustrated in Figure 8. This approach is comprehensive and enforces the order of spans, so in cases of overlapping spans, it will not pass this step. Sometimes it is challenging to specify the sentences as a single segment, as some sentences are very long or very short, which requires another segmentation approach as illustrated in step-3.

- *Overlap Matching:* The third step divides the sentence using dynamic window sizes based on the length of the predicted feature, where the window size is larger (key length + 2) if the key is large (greater than 10); otherwise, the window is equal to the length of the key multiplied by 2. This approach narrows the segment by searching the entire patient note using dynamic window sizes. The segments are compared with the predicted feature using TFIDF, and the most similar sentence is selected. We follow the same procedure for word-by-word matching between segments and predicted

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

features along with their spans. The final words are sorted but without enforcing span sorting, which allows overlapping. This step solves two shortcomings from the previous step: first, the length of short sentence segments, and second, overlapping matches due to non-enforced span ordering.

- **Evaluation**:
For evaluation, we follow the previous work where micro F1 score is used along with precision and recall for measuring the final performance score of the model (Ganesh and Bansal 2023; B. Long, Tan, and Newman 2023; J. Zhou et al. 2022; Sarker et al. 2019; V. Yaneva et al. 2024). F1 score is the harmonic mean between precision and recall. Precision represents the model's ability to accurately predict the correct features (how many correct features are predicted?). Recall represents the model's ability to completely predict all features (Does the model capture all key features?). To calculate F1, precision, and recall, we need to compare the predicted features with ground truth features which are annotated by experts. We must count True Positives (TP), False Positives (FP), and False Negatives (FN) as shown in the equations below:

$$Precision = \frac{True\ Positives\ (TP)}{True\ Positives\ (TP)\ +\ False\ Positives\ (FP)}$$

$$Recall = \frac{True\ Positives\ (TP)}{True\ Positives\ (TP)\ +\ False\ Negatives\ (FN)}$$

$$F1Score = \frac{Precision\ \times\ Recall}{Precision\ +\ Recall}$$

The process of matching predicted features with ground truth features consists of two primary stages:

  - First, three-step validation Matching Gate, which prevents the hallucination caused by LLM and finds the predicted features in the text using a hybrid of statistical term-weighting and fuzzy string matching methods techniques illustrated in the Matching Gate module section.
  - Second, once Matching Gate validates and matches the existence of predicted features in patient notes, the predicted features are semantically compared with ground truth using a sophisticated combination of matching techniques. The best and most accurate semantic similarity can be calculated using embedding. The embedding converts text into vectors in high-dimensional space, with each model having its own embedding technique. The semantic similarity goes through three sequential similarity steps:
    - *Using base LLM embedding*: In this case, using the embedding of the same model already loaded in memory (Mistral), which can produce fast comparison. The similarity is calculated using cosine similarity, which calculates the angle between two embeddings. These angles are represented using direction and size of vectors in high-dimensional space (Bojanowski et al. 2017).
      After observation, we set a high threshold to accept the similarity, especially with models trained with less data such as 6-shot instruct fine-tuning. Matching under 94% shows more obvious mistakes in calculating similarity, so 94% is set to be the threshold of Mistral LLM cosine embedding matching. The remaining features proceed to step 2.
    - We use an efficient and lightweight model designed to generate high-quality sentence embeddings (Wang et al. 2020). It uses contrastive training on sets of sentence pairs, enabling it to produce embeddings that capture not just keyword overlap but deeper conceptual relations in text. The similarity is calculated using cosine similarity as well, and the threshold is set to accept similarity at 50%. All features below the threshold are passed to step 3.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

After finishing two important steps: validating the existence of features using Matching Gate, then comparing the similarity between the predicted features and ground truth using the three mentioned steps. The next step is to pass the results into a classifier to count TP, FP, and FN as shown in (feature match classifier) algorithm 1:

Algorithm 1 Feature Match Classifier

Input:
PN - Patient Note: A text representation of patient medical records or notes.
PF - Predicted Feature: The feature predicted by the model.
GTF - Ground Truth Feature: The actual feature confirmed by medical experts.
AF - Associated Feature: Basic features associated with each case in general.

Output:
A string indicating the classification of the predicted feature:
"TP" - True Positive: Correctly identified is the predicted feature is present.
"TN" - True Negative: Correctly identified that the predicted feature is not present.
"FP" - False Positive: Predicted feature is incorrectly identified as present.
"FN" - False Negative: Predicted feature is incorrectly identified as not present.

```
procedure MATCH_PREDICTED_FEATURE(PN, PF, GTF, AF)
    if not PF and not GTF then
        return "TN"
    end if
    if PF and GTF then
        span ←MATCHING_GATE(PN, PF)
        sim ←CALCULATE_SIMILARITY(PF, GTF)
        if span and sim >0.5 then
            return "TP"
        else
            return "FN"
        end if
    end if
    if PF and not GTF then
        span ←MATCHING_GATE(PN, PF)
        sim ←CALCULATE_SIMILARITY(PF, AF)
        if span and sim >0.5 then
            return "TP"                          ▷ Annotation mistake
        else
            return "FP"
        end if
    end if
end procedure
```

In this classifier the predicted feature is considered as true positive if the feature exist and the ground truth feature exist as well, then two conditions must be satisfied the Matching Gate span detected and similarity greater than or equal to 50%, otherwise it will be considered as false negative.
For comprehensive evaluation to overcome missing annotation from expert, as the ability of LLM to capture feature consider very high as we will see in error analysis, we use the associated feature with each case as annotated feature if there is no annotated feature and the model predicts feature. The associated feature and the predicted feature go through same process as ground truth features annotated by experts.

RESULT

The results of the experiment are highlighted in Table 2. For fair comparison, we used the same data division. The total annotated patient notes are approximately 2,840 patient notes belong to 10 cases only 284 annotated patient notes per case (total 2840). There are two types of data: public and private. In the public dataset used in the Kaggle competition, only 1,000 annotated patient notes are available to the public; the remaining 1,840 are used for testing after users submit their notebooks. In the private dataset setting, the whole annotated data of 2,840 patient notes is available. We use public and private division for fair comparison with Yaneva et al.'s work (V. Yaneva et al. 2024). Table 2 summarizes the results of our models compared to Yaneva et al.'s results. Yaneva et al. conducted different experiments: INCITE, which uses traditional effective NLP approaches already used by Sarker (Sarker et al. 2019); DB = DeBERTa, which is used as a base model and trained on data; MLM = DeBERTa + Masked Language Modeling (MLM), where MLM is used as a pretraining step in which the model masks 15% of words in patient notes to better capture the context (in this case, non-annotated data is used to pretrain the model); and MTL = DeBERTa + Multi-task Learning, where the model trains on two tasks jointly—the primary task handles the token boundary (whether

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

it is inside or outside the span), and auxiliary tasks determine the boundaries of features. This approach showed the best results in their work. We ran different training approaches to see the capability and effect of each approach as follows:

- **Few-shot learning:** Here, we didn't train the model and no weights of the model changed, only utilizing the knowledge of the model itself with guidance by providing few examples of what the response should look like. We provided approximately 6 examples per case. Even though the result is not bad, with an F1 score of almost 82% in private and public datasets with zero training, this indicates that the model can already handle the task effectively.
- **Few-shot instruct tuning (6 examples):** We instruct-finetuned the model using only 6 examples from each case, which means a total of 60 examples. After finetuning, the model was also given few-shot learning examples to provide more guidance after training. Even with this small training set, the result improved to almost 93%, which is competitive considering the effort required for this task.
- **Few-shot instruct tuning (10 examples):** We also instruct-finetuned the model using only 10 examples from each case, which means a total of 100 examples, the same as public data in the Kaggle competition. After finetuning, the model was also given few-shot learning examples to provide more guidance after training. This result highlighted the power of LLM and how it can deeply understand semantics properly. The F1 score result exceeds the benchmark of previous work at almost 94%.
- **Full data instruct tuning:** Here, we followed the same setting as previous work but included standard data training division, which is 80% for training and 20% for testing. The result surpassed all previous results, though it was not significantly higher than the 10-example few-shot approach.
- **Automate annotated data**: As per the effort from Kaggle users, we tried to create more annotated data using a model finetuned on 2,000 annotated records and 3,000 automated patient notes, but the result was not as expected, possibly due to the quality of automated annotation of patient notes and parsing challenges during training.

Table 2 Performance comparison of different models on public and private datasets showing

		Public			Private		
		P	R	F1	P	R	F1
(V. Yaneva et al. 2024)	INCITE	.962	.818	.883	.961	.828	.888
	DB	.950	.962	.956	.951	**.963**	.957
	DB + MTL	.947	**.961**	.954	.953	**.963**	.958
	DM + MLM	.952	**.961**	.957	.961	.956	.958
Our Models	Few-shot Learning	.870	.789	.828	.866	.758	.824
	Few-shot_instruct tuning 6 samples	.953	.923	.938	.954	.928	.941
	Few-shot_instruct tuning 10 samples	.970	.928	.949	.971	.934	.952
	Full data instruct tuning	**.979**	.952	**.965**	**.974**	.952	**.963**
	New annotated data & original data	.909	.952	.929	.916	.932	.924

These different training approaches demonstrate the capabilities of large language models to capture deep semantic meaning in patient notes. Even with no training, the model was able to achieve an 82% F1 score on both private and public datasets. Moving to minimal training by fine-tuning the model with 6 samples per case, the score jumped significantly to 93%, proving that minimal training can produce notable improvements. Increasing the training samples to 10 per case improved the result to 94%. Providing more examples to train the model by using 80% of data resulted in the best performance at 96%. Lastly, using LLMs to produce more training data led to a drop in performance due to many challenges in producing annotated data where the spans were not consistent, and the parsing of annotated data was quite complex; the quality and accuracy of annotated data affected the performance of

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

the large language model. All these results attest to the fact that LLMs can effectively adapt to tasks where semantic understanding is highly required.

ERROR ANALYSIS

In this section we dive deep into the model prediction errors to provide a comprehensive view of model behaviour and what is the possible causes of these errors. This analysis can help in many directions: it can improve the model performance, distinguish between model limitations and data related issues, and provide insight for future research in same area. Based on our observation we divide the error into five categories as follows:

1. **Prediction Failure:** This is the fundamental error that show the model failure to predict the corresponding feature it is appear in three different ways:
    a. *Missing features:* The required feature not predicted (empty).
    b. *Hallucinated features:* The fact that LLM can hallucinate and predict feature similar to the feature associated with case itself but not actually exist in patient note. This is the reason we develop and use Matching Gate validation so only exist feature pass this validation step.
    c. *Irrelevant features:* The model predict non-correct and non-related feature which is another form of LLM hallucination This error and its types considered as false negative FN in the feature match classifier algorithm.
2. **Boundary Discrepancies:** Decide specific boundary where the feature starts and ends form a challenge even for medical experts, who's assigned to do the data annotation. So, this challenge is confusing for human themselves. The annotators follow specific guideline and conduct regular meeting to discuss ambiguous annotation such as "visual hallucinations" feature, should be captured with visual or hallucinations sufficient? 18% of the data was double checked by annotators and the remaining notes were single rated. The boundary discrepancy appears in model behaviour in two ways:
    a. Extended Boundaries: The boundary of predicted feature include longer phrase than the ground truth annotated feature.
    b. Partial Boundaries: The boundary of predicted feature exclude phrase that exist in the ground truth annotated feature.

    These discrepancies exist in ground truth data agreement the boundary not set to be accurate 100%, but after observation it reflect the required meaning of the features.
3. **Disordered predicted Feature:** This type of error occurs once the model able to predict the required feature but, it fails to maintain the exact order of feature, so when feature pass to Matching Gate 3 steps it is failed to match unordered features. Even this error is less frequent, but it affects the order of the features.
4. **Inference Limitations:** Despite significant efforts invested in inference pipeline and after many experiments to cover all possibilities by applying different techniques there are very little errors
    a. Matching Failure: Matching Gate validation steps occasionally fail to match the exact portion in the text with predicted feature.
    b. Similarity Failure: Once apply feature match classifier, it fails to detect the similarity using either embedding cosine similarity or BERT sentence cosine similarity.

    The feature with similarity or matching failure, once passed to classifier it is classified as false negative prediction.
5. **Annotation Divergence:** This type of error shows the power of LLM in capture deep meaning in the text. Here the model captures similar feature, but it doesn't match the ground truth in these two cases:
    a. Annotation Errors: The predicted feature exists as it is in other patient note belong to the same case, so in this case it is considered as annotation error.
    b. Unrecognized Correct Predictions: The semantic similarity between predict feature and feature associated with case is clear but can't be decide by non-medical expert.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 3 shows representative examples of these errors observed in the prediction of our models. These examples are represented based on the categories and subcategories discussed earlier. The table lists the predicted feature along with real annotation annotated by expert and feature associated with the case itself.

Category	Subcategory	Predicted Feature	Real Annotated Feature	Feature associated with case
Prediction Failure	Hallucinated features	'denies weight loss'	'Weight stable'	'Weight stable'
	Irrelevant features	'during episodes'	'feels nauseous'	'Associated nausea'
Boundary Discrepancies	Extended Boundaries Case 6	'8/10 INTENSITY WITH respiration'	'8/10'	'Sharp OR stabbing OR 7 to 8 out of 10 on pain scale'
	Partial Boundaries Case 1	'8-10 hrs'	'abdominal pain began 8-10 hrs 8 to 10 hours of acute pain'	'8 to 10 hours of acute pain'
Inference Limitations	Match Failure	'aches'	'achy'	'Myalgias'
	Similarity Failure	'SOB', 'SOB'	'SOB'	'Shortness of breath'
Annotation Divergence	Annotation Errors	'denies dyspnea'	-	'No shortness of breath'
	Unrecognized Correct Predictions	'has gotten worse'	-	'Increased frequency recently'

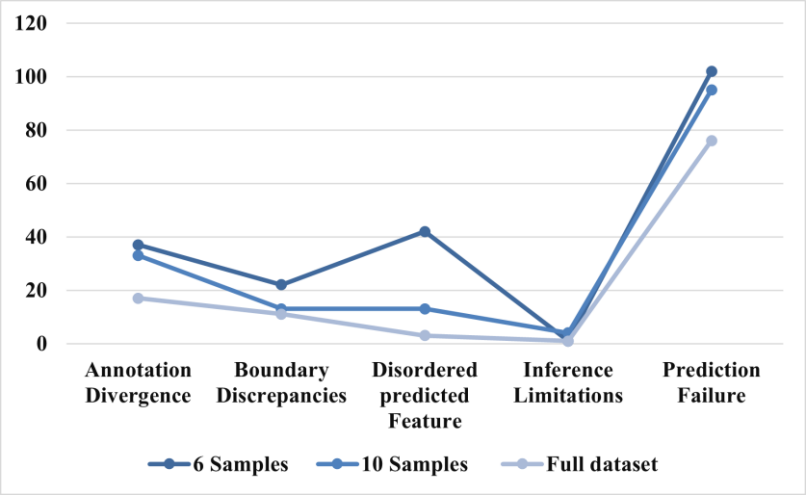

Figure 9 Detailed error analysis distributions on three models: instruct tuning (6 samples), instruct

The error analysis was conducted on three models with different size of training data: few-shot instruct tuning with 6 examples, few-shot instruct tuning with 10 examples, and full data instruct tuning, using public dataset splits. As shown in Figure 9.
, the pattern of errors appears on three distributions show relationships with the size of training data. Prediction Failure form 70% of the errors observed in the model tuned with full data, while other error types occur less

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

frequently. This indicates that more training data means more accurate prediction, when it shifts it tends to complete prediction failure rather than other errors. on the other hand, smaller training dataset tends to produce other than fail prediction error, but even though the model still able to capture the meaning and fulfil the task in satisfied accuracy.

consider broader view of these errors we can categorise it into 3 categories as shown in Figure 10:

1. **Terminology Variations:** The variation of terminologies in any language is a fundamental challenge for many NLP tasks. This challenge is the key to be resolved in this specific task. Representing the meaning using different synonyms and capturing this variation by understanding deep meaning is not a simple task. This type of error appears in the predicted features as annotation divergence and boundary discrepancies. In these errors, as discussed earlier, the model is almost able to predict the meaning, but either the boundary is not accurate, or the annotation is not accurate.
2. **Disordered and Inference Limitations:** This category does not belong to LLM prediction errors but rather to the inference pipeline when we validate predictions against hallucination. We use Matching Gate validation steps and then measure similarity using cosine similarity. In addition to inference failures, we have disordered features if the LLM predicts correct features but not in their proper order; in this case, they cannot be captured as correct features. This category is the least occurring error in prediction results.
3. **Content Hallucination:** This type of error is the highest captured type among all where the model couldn't predict the feature an all or predict non-exist or non-correct feature. This type of error is major drawback of LLM and take significant attention by researchers to resolve this issue.

by observing the Figure 10, we can notice two behaviours:

- Training Data Impact: More data used to train model less prediction failure occurs.
- Error Distribution Pattern: Content hallucination is highest score among other error categories, this error category reflects the model ability to predict. This insight drives us to focus more on improve the model ability to predict in future.

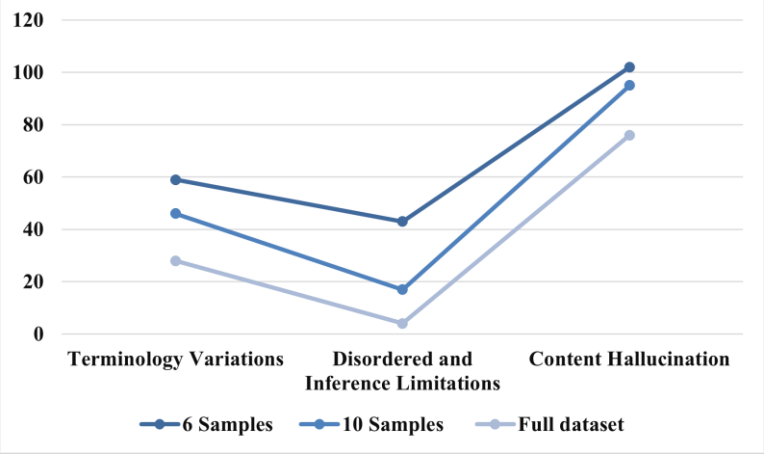

Figure 10 High-level error analysis distributions on three models: instruct tuning (6 samples), instruct

LIMITATION AND ETHICAL IMPLICATIONS

The dataset used in this work is relatively small, and all samples belong to only 10 cases. The goal of building Matching Gate matching steps is to use the fastest result by a hybrid of statistical term-weighting and fuzzy string-matching methods which is time efficient matching. These steps can be replaced by using a lightweight model trained on synonyms for this task; similarly, we aim to build a more generalized solution to serve further cases in the future. We avoid using lexicons or dictionaries to make the work more generalized and prove that LLMs are able to perform such feature extraction with minimal guidance. This task is challenging due to the variety of annotations, and the ability to specify exact start and end of features can be ambiguous and variable. Decisions that affect medical students' careers are very sensitive; expert involvement is required, and this model can be used as an aid to the automated

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

scoring process. This automation is only part of the complete exam evaluation, as aspects such as communication skills must be fulfilled by expert examiners, so a hybrid approach of automated and manual evaluation is required. Using a large language model, which is not an interoperable model as it acts as a black box, may affect the transparency and interpretability of the scoring process, which can affect the stakeholders' understanding and trust in this technology.

CONCLUSION

In this paper we produce MEDX-FE (Medical Exam Feature Extractor) framework, to automate the evaluation of licensed medical exams (USMLE step-2 CS). Our framework revels transformative capability of large language model to automate this process with remarkable performance (F1 score: 0.96), outperform previous work with minimal training data. This accomplishment represents considerable progress in the automation of medical assessment tools.

The most notable founding of our framework is its ability to achieve remarkable results even with limited training data (F1 scores of 0.94 with 10 examples and 0.93 with 6 examples per case). These results demonstrate the power of LLM utilize pretraining knowledge to resolve specialized medical task.

Our Matching Gate 3-steps validation module ensure reliable feature extraction and mitigate LLM hallucination by applying combination of matching techniques (exact, non-consecutive, and overlap matching). Matching Gate not only increases the accuracy but introduces transparency to the assessment process by matching each predicted feature withs its equivalent in the patient note.

We conducted comprehensive error analysis on three of our models. This analysis reveals insights into the behavior of models along with their limitations. The three significant results of this analysis are: As the training data increased the result improved, the LLM ability to capture meaning may exceed the human annotation process, and the content hallucination in addition to non-predicted feature remain the challenging error that need to resolve in the future.

National Board of Medical Examiners realize the importance of providing more efficient and resilient evaluation system, especially after the discontinuation of the USMLE Step 2 CS exam after COVID-19 pandemic. NBME in 2022 lunched a Kaggle computation to push the boundary of this automation process. MEDX-FE prove that LLM- based model can fill this gap. Despite that our framework designed for USMLE Step 2 CS examination, but the development was with aim of genialized and scalable solution that can be used by other medical educational institute conduct similar examinations.

With new era of AI many AI-based solution invented and used in medical education field. Our framework proves that autoregressive modern model like LLM can be used to the task were traditionally handled by bidirectional encoder-decoder model architecture. These traditional models have classification ability by understanding context in both directions, but LLM with its huge size and training parameters have deeper semantic understanding with limited token classification ability that can be resolved by Matching Gate validation steps.

The future directions of this research can be viewed from different directions. Use continual pretraining to train model on medical exam data either in form of MCQ or free text (Cossu et al. 2024). Use hybrid model LLM and BiLSTM or encoder-decoder model perfect to find the boundary. Adapt LLM self-training approach by augment or annotate training data then retrain model (D. Zhang et al. 2024). Lastly train LLM on many Medical exam tasks such as reasoning and solve multiple short answer questions.

REFRENCES

[1] Amaral, Eliana, and John Norcini. 2023. “Quality Assurance in Health Professions Education: Role of Accreditation and Licensure.” *Medical Education* 57 (1): 40–48.
[2] Amir, Gholami, Kim Sehoon, Dong Zhen, Yao Zhewei, W. Mahoney Michael, and Keutzer Kurt. 2021. “A Survey of Quantization Methods for Efficient Neural Network Inference.” *ArXiv [Cs.CV]*. arXiv. http://arxiv.org/abs/2103.13630.
[3] Avia, Efrat, and Levy Omer. 2020. “The Turking Test: Can Language Models Understand Instructions?” *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2010.11982.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

[4] Ben Zaken, Elad, Ravfogel Shauli, and Goldberg Yoav. 2021. "BitFit: Simple Parameter-Efficient Fine-Tuning for Transformer-Based Masked Language-Models." *ArXiv [Cs.LG]*. arXiv. http://arxiv.org/abs/2106.10199.
[5] Benítez, Trista M., Yueyuan Xu, J. Donald Boudreau, Alfred Wei Chieh Kow, Fernando Bello, Le Van Phuoc, Xiaofei Wang, et al. 2024. "Harnessing the Potential of Large Language Models in Medical Education: Promise and Pitfalls." *Journal of the American Medical Informatics Association: JAMIA* 31 (3): 776–83.
[6] Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. "Enriching Word Vectors with Subword Information." *Transactions of the Association for Computational Linguistics* 5 (December): 135–46.
[7] Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, et al. 2020. "Language Models Are Few-Shot Learners." *ArXiv [Cs.CL]*. arXiv. https://splab.sdu.edu.cn/GPT3.pdf.
[8] Chen, C., and Wei Cheng. 2008. "Beyond the Design of Automated Writing Evaluation: Pedagogical Practices and Perceived Learning Effectiveness in EFL Writing Classes." *Language Learning & Technology* 12 (June): 94–112.
[9] Chen, Tao, Enwei Zhang, Yuting Gao, Ke Li, Xing Sun, Yan Zhang, Hui Li, and Rongrong Ji. 2024. "MMICT: Boosting Multi-Modal Fine-Tuning with in-Context Examples." *ACM Transactions on Multimedia Computing Communications and Applications*, August. https://doi.org/10.1145/3688804.
[10] Chen, Zeming, Alejandro Hernández Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, et al. 2023. "MEDITRON-70B: Scaling Medical Pretraining for Large Language Models." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2311.16079.
[11] Chung, Hyung Won, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, et al. 2024. "Scaling Instruction-Finetuned Language Models." *Journal of Machine Learning Research: JMLR* 25 (70): 1–53.
[12] Chung, Hyung Won, Hou Le, Longpre Shayne, Zoph Barret, Tay Yi, Fedus William, Li Yunxuan, et al. 2022. "Scaling Instruction-Finetuned Language Models." *ArXiv [Cs.LG]*. arXiv. http://arxiv.org/abs/2210.11416.
[13] Chunting, Zhou, Liu Pengfei, Xu Puxin, Iyer Srini, Sun Jiao, Mao Yuning, Ma Xuezhe, et al. 2023. "LIMA: Less Is More for Alignment." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2305.11206.
[14] Clifton, Poth, Sterz Hannah, Paul Indraneil, Purkayastha Sukannya, Engländer Leon, Imhof Timo, Vulić Ivan, Ruder Sebastian, Gurevych Iryna, and Pfeiffer Jonas. 2023. "Adapters: A Unified Library for Parameter-Efficient and Modular Transfer Learning." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2311.11077.
[15] ContactDoctor. 2024. "Bio-Medical-MultiModal-Llama-3-8B-V1: A High-Performance Biomedical Multimodal LLM."
[16] Cossu, Andrea, Antonio Carta, Lucia Passaro, Vincenzo Lomonaco, Tinne Tuytelaars, and Davide Bacciu. 2024. "Continual Pre-Training Mitigates Forgetting in Language and Vision." *Neural Networks: The Official Journal of the International Neural Network Society* 179 (106492): 106492.
[17] Cummins, Ronan, Meng Zhang, and Ted Briscoe. 2016. "Constrained Multi-Task Learning for Automated Essay Scoring." In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Stroudsburg, PA, USA: Association for Computational Linguistics. https://doi.org/10.18653/v1/p16-1075.
[18] Dong, Fei, and Yue Zhang. 2016. "Automatic Features for Essay Scoring – an Empirical Study." In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics. https://doi.org/10.18653/v1/d16-1115.
[19] Dubey, Abhimanyu, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, et al. 2024. "The Llama 3 Herd of Models." *ArXiv [Cs.AI]*. arXiv. http://arxiv.org/abs/2407.21783.
[20] Eric, Hulburd. 2020. "Exploring BERT Parameter Efficiency on the Stanford Question Answering Dataset v2.0." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2002.10670.
[21] Ganesh, Jay, and Ajay Bansal. 2023. "Transformer-Based Automatic Mapping of Clinical Notes to Specific Clinical Concepts." In *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*, 558–63. IEEE.
[22] Gerald, Shen, Wang Zhilin, Delalleau Olivier, Zeng Jiaqi, Dong Yi, Egert Daniel, Sun Shengyang, et al. 2024. "NeMo-Aligner: Scalable Toolkit for Efficient Model Alignment." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2405.01481.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

[23] Haizhou, Shi, Xu Zihao, Wang Hengyi, Qin Weiyi, Wang Wenyuan, Wang Yibin, Wang Zifeng, Ebrahimi Sayna, and Wang Hao. 2024. "Continual Learning of Large Language Models: A Comprehensive Survey." *ArXiv [Cs.LG]*. arXiv. http://arxiv.org/abs/2404.16789.
[24] Hou, Guangyu, and Qin Lian. 2024. "Benchmarking of Commercial Large Language Models: ChatGPT, Mistral, and Llama." https://www.researchsquare.com/article/rs-4376810/latest.
[25] Hu, Edward J., Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. "LoRA: Low-Rank Adaptation of Large Language Models." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2106.09685.
[26] Ji, Ziwei, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, et al. 2022. "Survey of Hallucination in Natural Language Generation." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2202.03629.
[27] Jiang, Albert Q., Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, et al. 2023. "Mistral 7B." *ArXiv [Cs.CL]*. arXiv. https://www.amax.com/content/files/2023/12/Mistral-7B-Whitepaper.pdf.
[28] Kamath, Uday, Kevin Keenan, Garrett Somers, and Sarah Sorenson. 2024. "LLM Challenges and Solutions." In *Large Language Models: A Deep Dive*, 219–74. Cham: Springer Nature Switzerland.
[29] Labrak, Yanis, Adrien Bazoge, Emmanuel Morin, Pierre-Antoine Gourraud, Mickael Rouvier, and Richard Dufour. 2024. "BioMistral: A Collection of Open-Source Pretrained Large Language Models for Medical Domains." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2402.10373.
[30] Latifi, Syed, Mark J. Gierl, André-Philippe Boulais, and André F. De Champlain. 2016. "Using Automated Scoring to Evaluate Written Responses in English and French on a High-Stakes Clinical Competency Examination." *Evaluation & the Health Professions* 39 (1): 100–113.
[31] Leacock, Claudia, and Martin Chodorow. 2003. "C-Rater: Automated Scoring of Short-Answer Questions." *Computers and the Humanities* 37 (4): 389–405.
[32] Liu, Lei, Xiaoyan Yang, Junchi Lei, Xiaoyang Liu, Yue Shen, Zhiqiang Zhang, Peng Wei, et al. 2024. "A Survey on Medical Large Language Models: Technology, Application, Trustworthiness, and Future Directions." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2406.03712.
[33] Long, Bowen, Fangya Tan, and Mark Newman. 2023. "Ensemble DeBERTa Models on USMLE Patient Notes Automatic Scoring Using Note-Based and Character-Based Approaches." *Advances in Engineering Technology Research* 6 (1): 107–107.
[34] Long, Ouyang, Wu Jeff, Jiang Xu, Almeida Diogo, L. Wainwright Carroll, Mishkin Pamela, Zhang Chong, et al. 2022. "Training Language Models to Follow Instructions with Human Feedback." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2203.02155.
[35] Margolis, Melissa J., and Brian E. Clauser. 2020. "Automated Scoring in Medical Licensing." In *Handbook of Automated Scoring*, 445–68. Chapman and Hall/CRC.
[36] Miller, Tristan. 2003. "Essay Assessment with Latent Semantic Analysis." *Journal of Educational Computing Research* 29 (4): 495–512.
[37] Mishra, Swaroop, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2021. "Cross-Task Generalization via Natural Language Crowdsourcing Instructions." *Annual Meeting of the Association for Computational Linguistics*, April, 3470–87.
[38] Mistral, A. I. 2024. "Mistral NeMo." July 18, 2024. https://mistral.ai/news/mistral-nemo/.
[39] Ramshaw, L. A., and M. P. Marcus. 1999. "Text Chunking Using Transformation-Based Learning." In *Natural Language Processing Using Very Large Corpora*, edited by Nancy Ide, Jean Véronis, Susan Armstrong, Kenneth Church, Pierre Isabelle, Sandra Manzi, Evelyne Tzoukermann, and David Yarowsky, 11:157–76. Text, Speech and Language Technology. Dordrecht: Springer Netherlands.
[40] Sahoo, Pranab, Prabhash Meharia, Akash Ghosh, Sriparna Saha, Vinija Jain, and Aman Chadha. 2024. "A Comprehensive Survey of Hallucination in Large Language, Image, Video and Audio Foundation Models." *ArXiv [Cs.LG]*. arXiv. http://arxiv.org/abs/2405.09589.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

[41] Sakaguchi, Keisuke, Michael Heilman, and Nitin Madnani. 2015. "Effective Feature Integration for Automated Short Answer Scoring." In . Stroudsburg, PA, USA: Association for Computational Linguistics. https://doi.org/10.3115/v1/N15-1111.
[42] Sarker, Abeed, Ari Z. Klein, Janet Mee, Polina Harik, and Graciela Gonzalez-Hernandez. 2019. "An Interpretable Natural Language Processing System for Written Medical Examination Assessment." *Journal of Biomedical Informatics* 98 (October): 103268.
[43] Shengyu, Zhang, Dong Linfeng, Li Xiaoya, Zhang Sen, Sun Xiaofei, Wang Shuhe, Li Jiwei, et al. 2023. "Instruction Tuning for Large Language Models: A Survey." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2308.10792.
[44] Stahl, Maja, Leon Biermann, Andreas Nehring, and Henning Wachsmuth. 2024. "Exploring LLM Prompting Strategies for Joint Essay Scoring and Feedback Generation." In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, 283–98.
[45] Suen, King Yiu, Victoria Yaneva, Le An Ha, Janet Mee, Yiyun Zhou, and Polina Harik. 2023. "ACTA: Short-Answer Grading in High-Stakes Medical Exams." In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, edited by Ekaterina Kochmar, Jill Burstein, Andrea Horbach, Ronja Laarmann-Quante, Nitin Madnani, Anaïs Tack, Victoria Yaneva, Zheng Yuan, and Torsten Zesch, 443–47. Stroudsburg, PA, USA: Association for Computational Linguistics.
[46] Sung, Chul, Tejas Indulal Dhamecha, and Nirmal Mukhi. 2019. "Improving Short Answer Grading Using Transformer-Based Pre-Training." In *Lecture Notes in Computer Science*, 469–81. Lecture Notes in Computer Science. Cham: Springer International Publishing.
[47] Taghipour, Kaveh, and Hwee Tou Ng. 2016. "A Neural Approach to Automated Essay Scoring." In . Stroudsburg, PA, USA: Association for Computational Linguistics. https://doi.org/10.18653/v1/D16-1193.
[48] Thirunavukarasu, Arun James, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. 2023. "Large Language Models in Medicine." *Nature Medicine* 29 (8): 1930–40.
[49] Tianci, Xue, Wang Ziqi, and Ji Heng. 2023. "Parameter-Efficient Tuning Helps Language Model Alignment." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2310.00819.
[50] Tobias, Kerner. 2024. "Domain-Specific Pretraining of Language Models: A Comparative Study in the Medical Field." *ArXiv [Cs.LG]*. arXiv. http://arxiv.org/abs/2407.14076.
[51] Tonmoy, S. M. Towhidul Islam, S. M. Mehedi Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. 2024. "A Comprehensive Survey of Hallucination Mitigation Techniques in Large Language Models." *ArXiv [Cs.CL]*. arXiv. https://www.amanchadha.com/research/2401.01313.pdf.
[52] Touvron, Hugo, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, et al. 2023. "LLaMA: Open and Efficient Foundation Language Models." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2302.13971.
[53] Tsichlis, Jason T., Andrew M. Del Re, and J. Bryan Carmody. 2021. "The Past, Present, and Future of the United States Medical Licensing Examination Step 2 Clinical Skills Examination." *Cureus* 13 (8): e17157.
[54] Uto, Masaki. 2021. "A Review of Deep-Neural Automated Essay Scoring Models." *Behaviormetrika* 48 (2): 459–84.
[55] Vaswani, Shazeer, and Parmar. 2017. "Attention Is All You Need." *Advances in Neural Information Processing Systems*. https://proceedings.neurips.cc/paper/7181-attention-is-all-you-need.
[56] Vsevolodovna, Ruslan Magana. 2024. *Ai-Medical-Chatbot: Free Doctor Consultation with Artificial Intelligence by Using Generative AI*. Github. https://github.com/ruslanmv/ai-medical-chatbot.
[57] Wang, Wenhui, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. "MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers." Edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. *Neural Information Processing Systems* abs/2002.10957 (February): 5776–88.
[58] Xiao, Changrong, Wenxing Ma, Qingping Song, Sean Xin Xu, Kunpeng Zhang, Yufang Wang, and Qi Fu. 2024. "Human-AI Collaborative Essay Scoring: A Dual-Process Framework with LLMs." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2401.06431.

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

[59] Xu, J., Y. Jiang, B. Yuan, and S. Li. 2023. "Automated Scoring of Clinical Patient Notes Using Advanced NLP and Pseudo Labeling." *2023 5th International*. https://ieeexplore.ieee.org/abstract/document/10405427/.
[60] Yaneva, V., K. Y. Suen, J. Mee, and M. Quranda. 2024. "Automated Scoring of Clinical Patient Notes: Findings From the Kaggle Competition and Their Translation into Practice." *Proceedings of The*. https://aclanthology.org/2024.bea-1.8/.
[61] Yaneva, Victoria, Janet Mee, L. Ha, Polina Harik, M. Jodoin, and A. Mechaber. 2022. "The USMLE® Step 2 Clinical Skills Patient Note Corpus." *North American Chapter of the Association for Computational Linguistics*, 2880–86.
[62] Yang, Haoran, Yumeng Zhang, Jiaqi Xu, Hongyuan Lu, Pheng Ann Heng, and Wai Lam. 2024. "Unveiling the Generalization Power of Fine-Tuned Large Language Models." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2403.09162.
[63] Yim, Wen-Wai, Ashley Mills, Harold Chun, Teresa Hashiguchi, Justin Yew, and Bryan Lu. 2019. "Automatic Rubric-Based Content Grading for Clinical Notes." In *Proceedings of the Tenth International Workshop on Health Text Mining and Information Analysis (LOUHI 2019)*, edited by Eben Holderness, Antonio Jimeno Yepes, Alberto Lavelli, Anne-Lyse Minard, James Pustejovsky, and Fabio Rinaldi, 126–35. Hong Kong: Association for Computational Linguistics.
[64] Yuntao, Bai, Jones Andy, Ndousse Kamal, Askell Amanda, Chen Anna, Dassarma Nova, Dawn Drain, et al. 2022. "Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2204.05862.
[65] Zhang, Dan, Sining Zhoubian, Yisong Yue, Yuxiao Dong, and Jie Tang. 2024. "ReST-MCTS*: LLM Self-Training via Process Reward Guided Tree Search." Edited by A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang. *Neural Information Processing Systems* abs/2406.03816 (June): 64735–72.
[66] Zhang, Tianyi, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2020. "Revisiting Few-Sample BERT Fine-Tuning." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2006.05987.
[67] Zhou, Hongjian, Fenglin Liu, Boyang Gu, Xinyu Zou, Jinfa Huang, Jinge Wu, Yiru Li, et al. 2023. "A Survey of Large Language Models in Medicine: Progress, Application, and Challenge." *ArXiv [Cs.CL]*. arXiv. http://arxiv.org/abs/2311.05112.
[68] Zhou, Jianing, Vyom Nayan Thakkar, Rachel Yudkowsky, Suma Bhat, and William F. Bond. 2022. "Automatic Patient Note Assessment without Strong Supervision." In *Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis (LOUHI)*, edited by Alberto Lavelli, Eben Holderness, Antonio Jimeno Yepes, Anne-Lyse Minard, James Pustejovsky, and Fabio Rinaldi, 116–26. Abu Dhabi, United Arab Emirates (Hybrid): Association for Computational Linguistics.