CM-CliniBOT: A Code-Mixed Hindi-English Clinical Chatbot

Akshat Saxena[1], **Dipankar Srirag**[2] **and Aditya Joshi**[3]
[1]Indian Institute of Kanpur, India
[2,3]University of New South Wales, Sydney, Australia
asaxena23@iitk.ac.in$_1$ d.srirag@student.unsw.edu.au$_2$ aditya.joshi@unsw.edu.au$_3$

Abstract

Leveraging the benefits of **Natural Language Processing** in the medical domain has been the major focus of the researchers now. There have been several models, that have been created for the same purpose and continuous work is being done to increase the accuracy, as well as the efficiency of it. Our work will also follow the same outlines. Despite the advancements, there is one problem that still needs to be solved, and that is the problem of language diversity. In this work, we present a *Retrieval Augmented Generation* model, which would perform two major tasks concurrently: **(i)** Take up code-mixed Hindi-English queries and understand their meaning and **(ii)** Come up with the most accurate and relevant response to help the patient with their query. To tackle the issue, we created a comprehensive dataset having code-mixed queries along with the responses, as there is no such dataset present over the internet, made for working out the same issue. The code and dataset are publicly available at this link[1]

Our innovative approach aims to bridge the language gap in medical NLP applications, potentially improving patient outcomes and accessibility to medical information. Future work will refine the model and expand the dataset to cover more diverse linguistic patterns.

1 Introduction

Natural Language Processing has transformed many domains, enabling machines to understand the human and technical language helping to carry out the task more efficiently and correctly. One such domain is the medical domain, where NLP has significantly improved by assisting in clinical decision-making. However, the major problem that still needs to be solved is to enhance the bilingual and polylingual adaptability of models, since, the majority of the research focuses mainly on monolingual context, primarily in English, which creates a problem for some patients who find trouble in articulating their problem in English. Any miscommunication in the medical field, where precision and clarity are paramount, can cause severe consequences. Thus it seeds the need for building up models that understand code-mixed languages.

Code-Mixing: Code-mixing, or code-switching, refers to the phenomenon where a speaker alternates between two or more languages within a single sentence or discourse. In this study, we focus on code-mixing between Hindi and English, with the potential for future work to explore other language combinations. For example, an English sentence such as "Doctor, could you share the prescription with me?" translates to a code-mixed Hindi-English version: "Doctor, kya aap mere saath prescription share kar sakte hain?" This sentence incorporates words from both English and Hindi vocabularies.

Retrieval-Augmented Generation: Recent advancements in Retrieval-Augmented Generation (RAG) models have demonstrated superior performance in tasks requiring detailed domain-specific knowledge while ensuring factual correctness. This approach is particularly beneficial in the medical domain, where precision and accuracy are crucial.

There is an ongoing issue of language diversity, and to make NLP more accessible and effective for medical purposes we must devise a solution to this problem. Most of the earlier research focuses on advancing the medical-based RAG model's efficiency, and only a few bridge the

[1]https://huggingface.co/ProElectro07/CM-CliniBOT
https://huggingface.co/datasets/ProElectro07/CM-Medical-Utterances

gap in handling code-mixed sentences. Current NLP is not adequately equipped to understand code-mixed sentences, which are frequently used in countries like India, where people convey their problems to medical professionals in the same format.

Our research primarily focuses on solving this issue to leverage, as well as enhance the model's capability to understand the query. For the same, we will be working with the **Mistral-7B v0.1 Instruct**[2] model. The primary objectives will be to:

i) Understand the patient's code-mixed Hindi-English sentences.
ii) Generate the most relevant responses to solve the patient's query in English.

Our research addresses the crucial need to overcome the monolingual problem in medical NLP applications. By developing a RAG framework, we aim to improve healthcare accessibility among people facing lingual problems. In this way, we can overcome the significant language barrier in this domain, thus enhancing the patient's experience.

2 Related Work

In recent years, significant efforts have been directed toward developing models specifically tailored for medical applications, with the primary goal of improving diagnostic accuracy and clinical decision-making. Large language models such as Med-PaLM, BioBERT, and ClinicalBERT have been meticulously designed and trained on vast corpora of medical texts to address the unique challenges of healthcare. These models have shown substantial promise in understanding and generating medical content, thus contributing to enhanced accuracy in medical tasks such as disease diagnosis, treatment recommendations, and patient care.

However, the journey toward achieving higher accuracy in healthcare has not been limited to models explicitly developed for medical purposes. A growing body of research has explored the integration of general-purpose models with domain-specific knowledge through techniques like Retrieval-Augmented Generation (RAG). This approach involves augmenting the model's generative capabilities with relevant external knowledge retrieved from large datasets, which has been demonstrated to significantly boost the accuracy of model predictions in the medical domain. For instance, the study presented in (Xiong et al., 2024) exemplifies how incorporating retrieval mechanisms into generative models can lead to substantial improvements in the accuracy of biomedical information processing.

Further advancements have been made in creating toolkits and frameworks designed to facilitate the development of RAG models tailored to healthcare applications. The work by (Li et al., 2024) introduces a comprehensive toolkit aimed at streamlining the creation of RAG models to build reliable models for various medical tasks. Additionally, innovative applications of RAG have extended beyond traditional healthcare, as seen in (Vakayil et al., 2024), which explores the use of RAG models to provide psychological support to survivors of sexual harassment. Another notable study, (Yang et al., 2024), applies RAG techniques to address a broad range of general healthcare needs, highlighting the versatility and effectiveness of this approach.

In parallel with these advancements, researchers have also tackled the challenge of code-switching, and much work has been done. (Ghosh et al., 2024) demonstrates the integration of visual information with code-mixed queries, thereby facilitating more accurate and comprehensive information retrieval.

To support these endeavors, several datasets have been created specifically to address the complexities of code-mixed language in healthcare. For instance, datasets like (Banerjee et al., 2018), (Dowlagar and Mamidi, 2023)(Telugu–English Code-Mixed) provide a rich dataset that captures the nuances of code-mixed language, thereby enabling more effective model development and fostering greater accessibility in healthcare communication.

3 Dataset

3.1 Data Collection

To work out our model, we needed two datasets. One of the datasets was required to train the model,

[2] https://huggingface.co/filipealmeida/Mistral-7B-Instruct-v0.1-sharded

to understand code-mixed sentences, while the other was required to evaluate the model. For the training of our model, we required a dataset having conversations between a patient and a doctor in Hinglish format, but since there was no dataset present over the internet for our case, we searched for a dataset online having a conversation between a doctor and a patient, in English so that we could further annotate it to convert the English queries into Hinglish ones. We found a comprehensive dataset for the case.

For the evaluation part of our model, we split our training dataset to get our testing dataset part. Since we are designing the RAG model mainly for COVID purposes, we extracted the queries that were COVID-specific, and thereafter, created a testing dataset out of it.

Although the dataset we are working with is quite small, further research can be conducted on the same issue with a larger dataset. This will increase the robustness of the model, leading to better results.

3.2 Data Annotation

For our purpose, we needed a dataset having code-mixed Hindi-English queries. To achieve that, we manually converted the English queries into Hinglish queries. To convert the English queries into Hinglish ones, we employed a method inspired by the approach described in (Ghosh et al., 2024). This method involves using prompts to generate code-mixed text from monolingual English text. Specifically, we designed prompts that guide the transformation of English queries into a natural code-mixed format, incorporating commonly used Hindi terms and phrases. This approach ensured that our dataset accurately reflects the linguistic patterns widely used in medical consultations in multilingual societies like India.

4 Methodology

To create such a model, we need to fulfill three major tasks:

i) Model Training
ii) Retrieval of documents
iii) Response Generation for Code-Mixed Queries

Let's expand each step and delve deeper into it:

4.1 Model Training

To understand the Code-Mixed Hindi-English queries, we need to initially train the model to understand such sentences by adding up the Hindi tokens in its vocabulary. We chose a sharded version of **MISTRAL-7B v0.1-INSTRUCT** for the model selection to leverage its adaptability on instruction-based tasks. The method we chose here was incremental training, that is we loaded the adapters learned from a session to the model, to prepare it for the next learning session. We trained the model two times to fulfill the two different objectives of **i)** understanding the Hindi tokens and **ii)** learning to respond to the patient like a doctor.

4.1.1 Understanding the Hindi tokens

For the first three epochs, we trained the model by providing all the labeled data to the model, the code-mixed query, its English translation, and the Doctor's response from our dataset. This step was crucial to make the model learn the Hinglish tokens.

4.1.2 Learning to respond

Now for the other four epochs, we trained the model by providing only two labeled data, the Code-Mixed query, and the Doctor's response from our dataset. We need to add this step as, during the actual operating process, we won't be providing the English translation of the code-mixed queries, so it becomes significant for the model to understand the query on its own, which is fulfilled by this step.

The number of training epochs in our study was not predetermined, but dynamically determined based on the loss function's behavior. Training continued as long as the loss showed a consistent decreasing trend and was terminated when a significant spike in loss was observed. This approach optimized model performance while mitigating overfitting risks. Moreover, one more problem of teaching the model to generate a response like a doctor was simultaneously managed from both steps. Furthermore, we needed to replicate the functioning of a typical RAG model.

4.2 Document Retrieval

After completing model training, setting up the retriever component was crucial. In this study, we utilized the Abstracts of 10,000 Covid Research Papers[3] database, which comprises abstracts of 10,000 research papers on the subject. For document embeddings, we employed the **BAAI** model.

Chunking Strategy: We tried out different chunking strategies with different chunk sizes, keeping in mind the context window of the model, and concluded that specifically in the medical domain, we do not have to mix up the information, thus combining two different sources of information, which could potentially decrease the quality of information or context that was getting retrieved, and thus to eliminate this, we proceeded by taking each abstract as a chunk rather than breaking it further into smaller chunks.

We established a vector database to store these embeddings using FAISS, chosen for its robust functionality in similarity search. Inspired by the approach detailed in (Xiong et al., 2024), we designed an ensemble retriever. This setup combines FAISS's retrieval capabilities, which effectively identify the most semantically similar documents, with the BM-25 algorithm, which retrieves documents based on token frequency. This combination enhances the retriever's performance by integrating both semantic similarity and term frequency-based retrieval methods. Here, we checked for different weights, to find the best settings for our model.

Multi-Query Retriever: Besides, we also introduced the multi-querying method into the retriever. In this method, the model generates some queries similar to the actual query and generates a final embedding from all the queries it received and generated, thus enabling the embedder to capture the information accurately by introducing several variations of the query.

4.3 Response Generation for Code-Mixed Queries

Our initial strategy employed a single-step process for handling code-mixed queries:

- We designed a prompt template that instructed our language model to:
  1. Understand the code-mixed query.
  2. Generate a response based on retrieved-context.
- The model used was Mistral IN v0.1, which has a context window of 8192 tokens.
- The prompt included:
  i) Instructions for query understanding.
  ii) Instructions for response generation.
  iii) The code-mixed query itself.
  iv) Retrieved relevant documents.

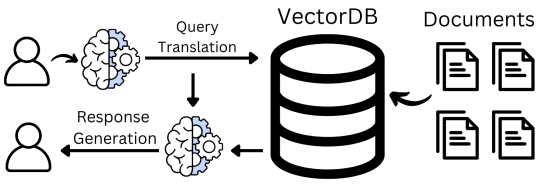

Figure 1: Image demonstrating our refined approach

5 Challenges Encountered

We faced several issues with this approach:

1. **Context Window Limitations**: The combined size of the prompt, query, and retrieved documents often exceeded the 8192 token limit.
2. **Incomplete Query Comprehension**: Due to truncation caused by token limitations, the model sometimes failed to understand the query fully.
3. **Inaccurate Responses**: Incomplete understanding led to generated responses that were often inaccurate or irrelevant.
4. **Prompt Optimization Difficulties**: Attempts to modify the prompt to fit within token limits resulted in degraded performance scores.

6 Strategy Refinement

To address these challenges, we developed a two-stage approach, and designed the prompt template[A] accordingly.

[3] https://www.kaggle.com/datasets/anandhuh/covid-abstracts

6.1 Query Translation

- **Input**: Code-mixed query
- **Process**: Translate the query in English
- **Output**: English translation of the original query
- **Rationale**:
  i) **Reduces the overall token count**: Optimizes processing by minimizing token usage.
  ii) **Enhances alignment with English-trained document embedders**: Improves the effectiveness of embeddings by better matching the training context.
  iii) **Boosts compatibility with similarity-matching algorithms**: Increases the accuracy of similarity measures by aligning with algorithmic expectations.

6.1.1 Response Generation

- **Input**: Translated English query from the previous chain.
- **Process**:
  1. Retrieve relevant documents based on the English summary.
  2. Generate a response using the retrieved context.
- **Output**: Generated response to the original query.
- **Rationale**:
  i) Improves retrieval accuracy due to monolingual matching.
  ii) Enhances response relevance and accuracy.
  iii) Better utilizes the model's training on English data.

6.2 Theoretical Basis for Refinement

The refinement strategy is grounded in several key observations:

1. **Vector Representation**: When we encode data, we obtain vector representations. Document embedders are typically trained on monolingual (often English) corpora.
2. **Embedding Space Mismatch**: Code-mixed queries create vectors that may not align well in the embedding space of monolingual documents. This misalignment can lead to poor similarity matching.
3. **Similarity Measures**: Most similarity algorithms are optimized for monolingual comparisons. They may fail to capture the semantic relevance when comparing code-mixed queries to monolingual documents.
4. **Model Training**: While our model was fine-tuned on code-mixed data, the underlying pre-training and most of its knowledge are based on monolingual (primarily English) text. This makes it more effective at processing and generating responses in a single language.

6.3 Expected Benefits

By implementing this two-stage approach, we anticipate:

- Improved retrieval accuracy
- More relevant and accurate responses
- Better utilization of context window
- Enhanced scalability to different code-mixed language pairs

7 Model Evaluation

We evaluated our model primarily using the BERTScore metric, which takes into account both recall and precision of the generated responses. Additionally, we assessed our RAG model using the BLEU and ROUGE scores.

We compared our model's responses with those of ChatGPT 4o. While we observed that our benchmark model outperformed our model in terms of BERT score, but the margin was not substantial. The higher BERT score of model can be primarily attributed to the training dataset and testing dataset as well, which we will discuss further. The results are shown in Table 1.

7.1 Reasons for Low BLEU and ROUGE Scores

The BLEU and ROUGE metrics evaluate predictions based on exact word matches between the predicted and reference sentences. While useful for monolingual text, this approach can be overly strict

Model	ROUGE			BLEU			BERT Score		
	R1	R2	RL	B1	B2	B3	Precision	Recall	F1-score
ChatGPT 4o	21.82	1.76	13.15	24.67	1.12	0.15	84.19	82.55	83.36
CM-CliniBOT (Trained)	19.93	1.82	11.43	19.66	1.26	0.22	83.61	82.60	83.09
CM-CliniBOT (Untrained)	18.32	1.66	10.31	15.18	0.93	0.17	82.31	82.44	82.36

Table 1: ROUGE, BLEU, and BERT scores for various models.

for code-mixed language processing. Code-mixed queries often involve nuanced language switches and varied vocabulary, which these metrics do not adequately account for.

Sometimes, the model generates sentences that use different words or phrases but convey the same meaning as the reference. These semantically correct outputs may score poorly on BLEU and ROUGE due to the lack of exact word overlap. Moreover, BLEU is generally used for language translation, while Rouge is used for summarization tasks, which may also indicate the cause of low scores.

In contrast, the **BERTScore** metric calculates the F1 score based on the semantic meaning of tokens and words rather than exact matches. This allows it better to capture the contextual relevance and correctness of the responses. Despite the lower BLEU and ROUGE scores, a high BERT score indicates that the model produces semantically meaningful responses since it focuses more on finding the overlap between the meanings of two different words. In summary, while BLEU and ROUGE scores may not fully reflect the model's performance in handling code-mixed queries, the BERTScore and manual evaluation indicate that the model effectively produces meaningful and correct responses. Further, we analyzed based on human evaluation.

7.2 Human Evaluation and Behavioral Assessment

The primary objective of our project was to develop a model capable of generating responses that effectively communicate comprehensive information in a patient-friendly manner. During our evaluation, it became evident that state-of-the-art models such as ChatGPT-4o achieved higher BERT scores compared to our model. These benchmark models excel in generating responses that are closer in structure and content to the reference texts, which translates into higher performance on metrics that measure semantic similarity and alignment with expected outputs.

During manual evaluation, we observed significant differences between the responses generated by our model and those produced by state-of-the-art models like ChatGPT. Although these benchmark models are known for their linguistic proficiency, our analysis revealed that their responses often lacked the depth and empathy necessary for effective patient communication. Specifically, their replies were frequently brief and tended to address only isolated aspects of the patient's query, without offering a holistic view of the medical condition being discussed.

While our model did not match the BERT scores of these models, it was designed with a focus on providing detailed and contextually relevant responses. This design choice resulted in outputs that were more thorough and informative, addressing various aspects of the patient's query. Despite the lower BERT scores, our model aimed to ensure that responses were well-rounded and covered a broader spectrum of information.

7.3 Retriever Evaluation

In this section, we introduce a novel evaluation method to assess the quality of documents retrieved by various retrievers. Traditional evaluation methods often require extensive manual effort to verify document relevance, making the process both time-consuming and prone to subjectivity. To address these challenges, we developed an automated scoring system that efficiently evaluates document relevance through an ensemble multi-query retriever, designed to retrieve four documents in a hierarchical order.

Our scoring system assigns weighted scores based on the position of the document in the retrieval hierarchy: 0.4 for the first document,

0.3 for the second, 0.2 for the third, and 0.1 for the fourth, furthermore, the total score is scaled to 100. These scores are aggregated only if the documents are deemed relevant to the patient's query. Relevance is determined through a large language model (LLM)-based approach that evaluates factual similarity to the query, assigning labels of "Relevant" or "Irrelevant." A document receives the full score if labeled relevant, and a score of 0 if labeled irrelevant. We designed the prompt[A] accordingly.

The evaluation process was applied across several retrievers, as summarized in Table 2. The ensemble retriever, incorporating both FAISS and BM25 retrievers, was anticipated to perform well due to its diversified retrieval strategy. However, the evaluation revealed that the FAISS retriever achieved the highest score.

Retriever	Score
Multi Query Retriever	53.99
Ensemble Retriever	52.99
FAISS Retriever	**55.19**
BM25 Retriever	52.39

Table 2: Table denoting the retrieval score

Interestingly, while our primary focus was on the ensemble multi-query retriever, the FAISS retriever emerged as the top performer with a score of 55.19. This result suggests that the FAISS retriever's indexing and nearest-neighbor search capabilities are highly effective in this context, even when compared to a more complex ensemble method.

This outcome highlights the importance of empirical evaluation in retriever selection and underscores the potential for further performance optimization. Although the multi-query retriever showed competitive results, the FAISS retriever's superior performance indicates that a more refined indexing strategy can yield better retrieval outcomes. Moving forward, we anticipate that fine-tuning the FAISS retriever or integrating its strengths into a hybrid model could enhance overall performance. Additionally, our scoring system's flexibility allows it to be adapted for different use cases, depending on the number and nature of documents retrieved.

8 Limitations and Future Work

The main constraint we faced was the size of our training dataset. Due to resource limitations, the dataset used for training and fine-tuning was kept relatively small. Consequently, the model was not exposed to sufficiently diverse data, which impacted its accuracy in generating summaries.

Despite these limitations, our work provides a foundation for future research in this area. We encourage researchers to build upon our findings and address the following areas:

- Expanding the training dataset to include more diverse code-mixed queries.
- Work upon the document embedders to handle the code-mixed queries, increase the RAG model's performance, and increase the quality of documents retrieved.
- Exploring advanced techniques for low-resource learning in multilingual settings.
- To try out different languages, and make the model adaptable to work out more languages.
- By selecting, or modifying the pre-existing model for a greater context window, attempts could be made to eliminate the summarization part, and stuff up all the instructions in a single prompt, making it more convenient to work with.

By overcoming these limitations, future work could lead to more robust models capable of handling a wider range of code-mixed queries with higher accuracy. In addition to it, our research mainly focuses on generating English responses to code-mixed Hindi-English queries, further work could be done to explore generating Hinglish and taking in the audio or images as input as well to enhance the effectiveness.

9 Conclusion

In conclusion, our research addresses the challenges of teaching models to handle lingually diverse data, particularly code-mixed Hindi-English queries. We identified the problem of effectively embedding such documents and proposed a comprehensive, step-by-step approach to mitigate or eliminate these issues.

Our technique summarizes the code-mixed

Hindi-English queries before feeding them into the model. This summarized input allows the model to process the query more efficiently and accurately, leading to more relevant responses based on the most pertinent documents retrieved.

Many existing models and document embedding techniques are designed primarily for monolingual data, limiting their ability to handle code-mixed queries effectively. By focusing on this gap, our research contributes to the development of more robust and versatile language models capable of managing code-mixed inputs. This advancement is crucial for improving the performance and applicability of natural language processing systems in multilingual contexts.

Furthermore, our approach paves the way for future research in enhancing model capabilities for other types of mixed-language data, ultimately fostering more inclusive and comprehensive language technologies.

A Appendix

A.1 Prompt Templates

We used one shot prompting for our case, and provided examples of the part we wanted it to output, as it showed better results then.

A.1.1 Hindi-English Translation

```
You are an experienced linguistic
expert and an assistant to a doctor
who only understands English. You
will be given code-mixed Hindi-English
queries and you need to translate
them into English.
Example:
Patient: Doctor sahib, mumjhe bukhar
aur khansi ho rahi hai, kya yeh covid
ka lakshan ho sakta hai?
Response: Doctor, I have fever and
cough, are these symptoms of COVID?
Now, please translate the patient's
query below in a similar manner,
in English:
Patient's question: {input}
Provide an accurate translated query:
```

A.1.2 Response Generation

```
You are an experienced COVID-19
doctor, who will help out the
patient with their queries.
Example:
Doctor: Yes, fever and cough can
indeed be symptoms of COVID-19. I
recommend you get tested for COVID-19
as soon as possible. In the meantime,
please isolate yourself at home,
rest, and drink plenty of fluids.
Monitor your symptoms closely, and
if they worsen, especially if you
have difficulty breathing, seek
medical attention immediately.
Patient's question: {question}
Additional context that might help
with the patient's query: {context}
Provide a brief, concise reply just
like the doctor in the above example:
```

A.1.3 Retriever Evaluation

```
You are an expert in evaluating the
relevance of documents to a query.
Given the query and the 4 documents
below:
Query: {question}
Documents: {context}
Task:
For each document, label it as
"Relevant" if it helps to answer
the query, or "Irrelevant" if it
does not.
Example:
Relevant
Relevant
Irrelevant
Irrelevant
Please provide the labels only, with
each of the 4 document's label on a
new line like the above format:
```

References

Suman Banerjee, Nikita Moghe, Siddhartha Arora, and Mitesh M Khapra. 2018. A dataset for building code-mixed goal oriented conversation systems. *arXiv preprint arXiv:1806.05997*.

Suman Dowlagar and Radhika Mamidi. 2023. A code-mixed task-oriented dialog dataset for medical domain. *Computer Speech & Language*, 78:101449.

Akash Ghosh, Arkadeep Acharya, Prince Jha, Sriparna Saha, Aniket Gaudgaul, Rajdeep Majumdar, Aman

Chadha, Raghav Jain, Setu Sinha, and Shivani Agarwal. 2024. Medsumm: A multimodal approach to summarizing code-mixed hindi-english clinical queries. In *European Conference on Information Retrieval*, pages 106–120. Springer.

Mingchen Li, Halil Kilicoglu, Hua Xu, and Rui Zhang. 2024. Biomedrag: A retrieval augmented large language model for biomedicine. *arXiv preprint arXiv:2405.00465*.

Sonia Vakayil, D Sujitha Juliet, Sunil Vakayil, et al. 2024. Rag-based llm chatbot using llama-2. In *2024 7th International Conference on Devices, Circuits and Systems (ICDCS)*, pages 1–5. IEEE.

Guangzhi Xiong, Qiao Jin, Zhiyong Lu, and Aidong Zhang. 2024. Benchmarking retrieval-augmented generation for medicine. *arXiv preprint arXiv:2402.13178*.

Rui Yang, Yilin Ning, Emilia Keppo, Mingxuan Liu, Chuan Hong, Danielle S Bitterman, Jasmine Chiat Ling Ong, Daniel Shu Wei Ting, and Nan Liu. 2024. Retrieval-augmented generation for generative artificial intelligence in medicine. *arXiv preprint arXiv:2406.12449*.