Mini Project Report

on

Hate and Fake Detection in Multilingual Social Media Text

Submitted by

21BDS012 Chava Srinivasa Sai

21BDS027 Kasu Sai Kartheek Reddy

21BDS029 Koppuravuri Abhiram

21BDS056 Rangoori Vinay Kumar

Under the guidance of

Dr . Sunil Saumya

Head Of The Department (DSAI)

DEPARTMENT OF Data Science and Artificial Intelligence

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY DHARWAD

May 8, 2024

Certificate

This is to certify that the project, entitled **Hate and Fake Detection in Multilingual Social Media Text**, is a bonafide record of the Mini Project coursework presented by the students whose names are given below during 2023 - 2024 in partial fulfilment of the requirements of the degree of Bachelor of Technology in Computer Science and Engineering.

Roll No	Names of Students
21BDS012	Chava Srinivasa Sai
21BDS027	Kasu Sai Kartheek Reddy
21BDS029	Koppuravuri Abhiram
21BDS056	Rangoori Vinay Kumar

<Sunil Saumya>
(Project Supervisor)

Contents

List of Figures

List of Tables

1 Abstract

This report represents an existing method for detecting hate speech on social media platforms like X (formerly known as Twitter), it mainly focuses on Natural Language Processing (NLP) and Vector Database. Our RAG model uses a two-stage process to retrieve relevant articles and predict hate speech comments. And it also explore the use of LoRA and its extensions (Quantized LoRA or QLoRA) to fine-tune Large Language Models (LLMs) for hate speech detection. The combination of RAG and QLoRA results in better performance than traditional methods.

2 Introduction

Social media sites increasingly serve as virtual news channels in the digital age, but they lack strong community guidelines. The rise of hate speech and fake news has raised concerns about internet safety and appropriate platform use. To overcome this issue, new technologies must be implemented in order to effectively detect and counteract harmful and misleading information.

In this research, we suggest an innovative approach for detecting hate speech and fake news using advanced natural language processing techniques. Our approach, known as Retrieval Augmented Generation (RAG), improves understanding of hate speech and fake news by analyzing data from Twitter conversations on a variety of topics. Using two stages technique, our model can collect relevant articles and create logical reasoning to predict negative reviews about hate speech and fake news.

Also, this study investigates the value of Low Rank Adaptation (LoRA) and Quantized LoRA (QLoRA) approaches to improve the fine-tuning of Large Language Models. LoRA and QLoRA techniques greatly decrease computational complexity and memory requirements by evaluating LLM weight matrices and adjusting only a subset of parameters, allowing for a more effective adjustment to hate speech and fake news identification duties.

This study improves the area of recognizing and tackling dangerous content online by presenting a complete framework based on the latest NLP algorithms. The experimental results give solid proof for the accuracy of this method, highlighting its potential to combat online hate speech and fake news, eventually ensuring a safer digital environment for everyone.

3 Related Work

While current methods for identifying hate speech on social media provide valuable insights, there are opportunities for improvement.

Deep Learning Approaches:- BiLSTM models have been increasingly employed for hate speech detection due to its ability to capture long-range dependencies in text[6]used BiLSTM models for detecting hate speech on Twitter task, illustrating the effectiveness of bidirectional RNNs in capturing the context and semantics of text. CNNs have been utilized in identifying hate speech on social media platforms, such as Twitter. In a recent study[7] a CNN-powered method was introduced for hate speech detection, incorporating a vectorization technique based on a crowd-sourced and continually updated hate word dictionary. The authors proposed integrating this approach with standard word embeddings to enhance the classification performance of the CNN model. Their methodology showcased superior performance compared to traditional shallow learning techniques .

Researchers have investigated the application of pre-trained language models in identifying hate speech. For instance, a study [8] introduced a BERT-based approach for this purpose on social media platforms, showing better performance than traditional machine learning methods. Likewise, another study[2] utilized BERT for hate speech detection on twitter, emphasizing its capability to comprehend intricate text semantics and adjust to different linguistic situations.

RAG and Vector Databases: Techniques such as NLP, vector databases, and RAG models find advantages beyond hate speech detection.LLMs tend to generate false information, known as hallucination. Usage of zero- or few-shot settings enhanced by retrieval augmentation and re-ranking for Tweet classification on GPT4 for hate speech detection has been done which

involves RAG which we are proposing in our approach [11] There is usage of RAG model and vector databases in Nanjing Yunjin (a traditional Chinese silk weaving craft) question and answer tasks, suggesting that our method can be applied cross-domain[10]. This study [1] uses a vector database-based Retrieval Augmented Generation (RAG) approach to enhance the accuracy and transparency of LLMs as religion question-answering chatbot. This work was done because the chatbots' responses can include content that insults personal religious beliefs, interfaith conflicts, and controversial or sensitive topics. It must avoid such cases without promoting hate speech or offending certain groups of people or their beliefs.

QLORA: Model complexity presents a challenge in hate speech detection tasks. To address this, QLoRA (Quantized Low-Rank Adaptation) is used for efficient fine-tuning of large language model. This approach, as demonstrated in recent research [9] , optimizes model performance while reducing computational overhead. They have finetuned LLaMA and Vicuna on a sub-sample of hate-speech dataset, using QLoRA in their research of finding effectiveness and adaptability of pre-trained and fine-tuned Large Language Models (LLMs) in identifying hate speech. This [4] have used Qlora to reduce the LLM (Llama-2-7B) memory usage without compromising performance. This research aimed to examine the factors essential for counter-speech, which is pivotal in understanding the optimal methods for addressing hate speech online. Various studies evaluate the emotional bases utilized in counter speech, including emotion-empathy, offensiveness, and the level of hostility.

Other surveys on hate speech: Comprehensive surveys provide a crucial foundation for understanding the landscape of hate speech detection. This survey [5] discusses about State-of-the-art hate speech identification methods, popular benchmark datasets of hate speech/offensive language detection specifying their challenges, the methods for achieving top classification scores, and dataset characteristics such as the number of samples, modalities, language(s), number of classes, etc. This literature review [3] systematically reviews textual hate speech detection systems and highlights their primary datasets, textual features, and machine learning models.Our proposed method combines NLP, vector database, RAG model, and QLoRA fine-tuning to provide unique support. This integration aims to increase contextual understanding and improve detection of hate speech on social media.

other methods: Various methods of analyzing hate speech have been explored.Graph convolutional networks (GCNs) to incorporate structural information into social media data, thus improving the accuracy of the analysis for hate speech identification .

4 Data and Methods

The main data was obtained from X (formerly known as Twitter) and YouTube as tweets and comments, respectively, resulting in a code-mixed dataset in Hindi and English. After gathering the data, we manually labelled each record in the dataset as hate or not-hate and fake or not-fake, and each record was cross-checked by other team members. The initial stage in pre-processing was to remove special characters from the text. We then transliterated the code-mixed text into Hindi using IndicXlit and performed translation using IndicTrans2 to convert it into English.

The final English Data, along with the Hate and Fake Labels, was fed into Fine-tuned Large Language Models (LLMs) using Quantized Low Rank Adaptation (QLoRA) configurations. The results were then collected. We employed various LLMs and tracked the progress of all of the data they produced during both the training and testing periods.

We next moved on to the primary phase, which required building the vector database. This requires gathering documents from a variety of sources, including blogs and webpages. We proceeded by identifying the topics that are covered in our dataset, and then scrapped the data from webpages for the chosen topics, and this data is stored in Vector Database.

We used LangChain to extract the top three relavent documents for each query from the vector database. Then, we used the BART model to summarize the data. The summarized text was stored. Finally, we passed the query and the reasons to the Fine-tuned LLMs and got the results.

4.1 Data Description

In this project, we scraped code-mixed data from social media platforms like X (formerly known as Twitter) and YouTube. Subsequently, we pre-processed the data and utilized AI4Bharat tools for transliterating the data into Hindi. Following that, we employed IndicTrans2 to translate the data into English. Finally, we manually labeled the data as either Hate or Non-Hate and Fake or Non-Fake, with the labeling for Fake or Non-Fake based on the claims made.

Table 1
Data Set Distribution

	Hate	**Not-Hate**	**Fake**	**Not-Fake**
Records	5138	2876	4137	3877
Total	8014		8014	

4.2 Fine Tuning

In the recent past, the field of Artificial Intelligence has seen many changes, particularly in the field of NLP, because of Large Language Models (LLMs). These LLMs have revolutionized the way machines understand human-level language. LLMs have many use cases, such as using them as virtual assistants, question-answering chatbots, and fine-tuning them for specific tasks.

While the pre-trained LLM models have impressive language understanding and generation capabilities, if they need to be adapted to perform a particular task, we need to fine-tune them. Fine-tuning is a process where we train the model for a specific task, whether it be for classification or question-answering chatbots for a particular field like medical or engineering.

Fine-tuning LLMs includes various tasks:

- ***Task Definition***: Defining the task that the fine-tuned model will be optimized for.
- ***Data Collection***: Gathering a sufficient amount of labeled or annotated data relevant to the task. The high quality of the data results in efficient fine-tuning.

- ***Model Selection***: Choosing a pre-trained model like GPT, BERT, etc.
- ***Fine-Tuning Process***: Adapting the fine-tuning process based on the task, whether it be text generation, sequence classification, etc.
- ***Safety Tuning***: Implementing measures to prevent the model from performing undesired actions.
- ***Evaluation and Validation***: Assessing the fine-tuned model's performance on the validation data.

By fine-tuning LLMs, programmers may fully use their potential for a variety of applications while still taking advantage of the advantages of pre-trained models and tailoring them to certain tasks and domains.

4.2.1 LoRA

Large Language Models (LLMs) are refined using a technique called Low Rank Adaptation (LoRA). The computational difficulties of optimising LLMs with constrained computer resources are tackled by LoRA. LoRA's main goal is to minimise memory and computational cost while optimising the previously taught LLM for a given task. This is accomplished during fine-tuning by splitting the LLM's weight matrix into low-rank matrices and updating subparts at a time rather than the entire weight matrix at once.

The LoRA technique works as follows:

- ***Parameter Decomposition:*** The weight matrix or the parameters of the pre-trained LLM are decomposed into low-rank matrices using techniques like Singular Value Decomposition (SVD) or Matrix Factorization Techniques.
- ***Subset Update:*** During the fine-tuning process, only a subset of weights of the weight matrix of the LLM are updated. This subset typically includes the top-k singular values and corresponding singular vectors, where k is a hyperparameter chosen based on the available computational resources and the characteristics of the task-specific dataset.

- ***Efficient Adaptation:*** By updating only a subset of parameters, LoRA achieves efficient adaptation to the task-specific data.
- ***Regularization and Optimization:*** In LoRA, regularization techniques like dropout are employed to prevent the model from overfitting.
- ***Performance Benefits:*** Empirical studies have shown that LoRA can significantly reduce the computational cost and memory footprint of fine-tuning large-scale LLMs while achieving competitive performance compared to standard fine-tuning techniques.

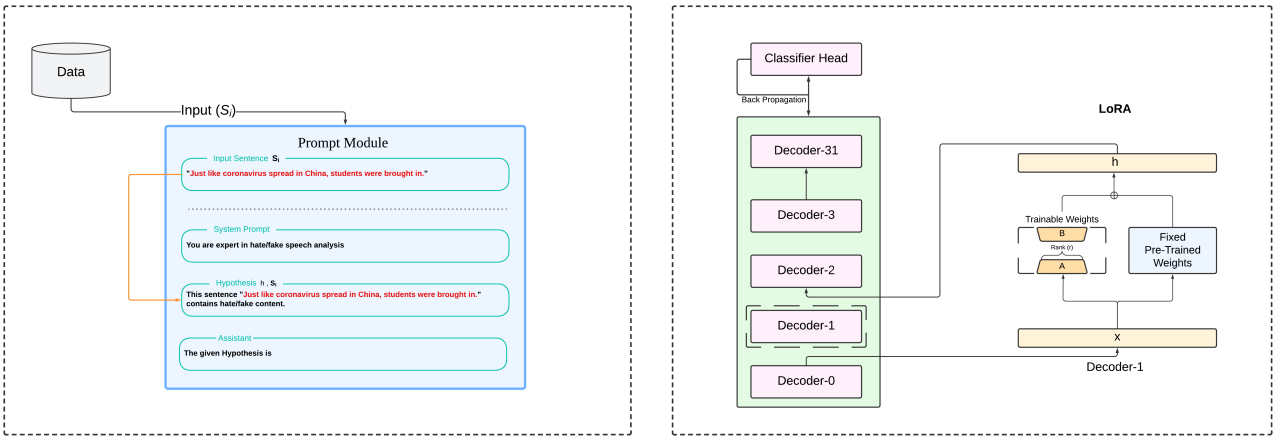

Figure 1. LoRA Architecture

4.2.2 QLoRA

In order to further optimise the efficiency of big language models, QLoRA, a revolutionary method in the field of natural language processing (NLP), not only builds upon the foundation set by Low Rank Adaptation (LoRA), but also includes novel quantization approaches. Through the use of quantization approaches, such as lowering model parameter precision from FP16 to FP8 or FP4, QLoRA seeks to drastically reduce memory and computing demands during the fine-tuning procedure.

Furthermore, QLoRA is a viable way for practitioners and researchers to investigate the possibility of utilising lower accuracy in model parameters without sacrificing performance. This creates prospects for the use of NLP models in situations with restricted computational

resources.

Moreover, the incorporation of quantization methods into the adaptation process leads to shorter inference times and improved scalability of fine-tuning procedures, hence improving the suitability of NLP models for real-time applications.

Furthermore, by encouraging the creation of lightweight NLP solutions that use fewer computing resources and hence lessen the carbon footprint associated with large-scale language model training and deployment, QLoRA adds to the continuing conversation on model efficiency and sustainability.

All things considered, QLoRA is a major development in natural language processing (NLP) as it provides a practical way to maximise model efficiency while upholding performance requirements, solving important issues with resource consumption, environmental effect, and scalability.

Table 2
Hyper parameters used for Training 7B's

Hyper Parameter	**Value**
Rank (LoRA config)	16
LoRA Alpha (LoRA config)	64
Dropout (LoRA config)	0.1, 0.2
Learning Rate	2×10^{-5}
Learning Rate Scheduler	Constant
adam_beta1	0.9
adam_beta2	0.999
adam_epsilon	1.000×10^{-8}
rms_norm_eps	1.000×10^{-5}

4.3 RAG

Retrieval Augmented Generation (RAG) is a cutting-edge methodology in natural language processing that integrates retrieval-based techniques with generative models. RAG optimizes the output of Large Language Models (LLMs). As LLMs are pre-trained up to a specific point

in time, they may lack up-to-date information if not connected to the internet. For instance, ChatGPT 3.5 is trained on data until January 2023, limiting its ability to answer queries beyond that period.

Explaining the Different Parts of RAG (Retrieval Augmented Generation) :

- **Retrieval:** In this phase, relevant information pertaining to the given query is retrieved from the vector database or other additional data sources. The vector database stores embeddings of text, facilitating efficient retrieval of relevant information.
- **Augmentation:** In RAG, the retrieved information enhances the knowledge of the Large Language Model (LLM), providing it with updated information regarding the query. This augmentation equips the LLM with additional insights it might not be aware of initially, enabling it to provide more comprehensive responses.
- **Generation:** During this stage, the LLM generates new text based on access to both the query information and the retrieved data. By leveraging this combined knowledge, the generated response to the user query aims to be more accurate, as the LLM has been provided with updated data through the retrieval process.

By integrating Langchian with our framework, we have used RecursiveCharacterTextSplitter. it is a tool in Langchain which is designed to split the text into individual characters using recursive approach, it is particularly useful when the task is to break down the text into smallest units which are typically characters.The key concept behind it is recursion where it will breakdown the problem into smaller and smaller sub parts and then it will divide those sub parts as well until a base case is arrived. In RecursiveCharacterTextSplitter the base case is to get the single character, so when we are using the RecursiveCharacterTextSplitter it will stop only when the text is splitted into the characters.

The data was scraped from the web pertaining to the specified queries and subsequently stored in a PDF file. Following this, the PDF file was loaded, and its text content was extracted.

The embeddings were then generated from the text using an embeddings model, and these embeddings were stored in Weaviate. Weaviate, an open-source vector search engine, facilitates the construction, exploration, and ranking of large datasets of text data using machine learning methods. Designed to handle unstructured data like natural language text, Weaviate employs semantic search techniques to retrieve relevant information based on query context.

The vector embeddings were stored in Weaviate, and through **vector_db.similarity_search**, the top three most similar chunks related to the query were retrieved. The data from all documents was combined and passed to the BART summary model, resulting in a summary comprising 2 to 5 sentences. Finally, the generated summary was stored for further use.
We then passed the relevant documents to the summarizing model BART and stored the output data for later use.

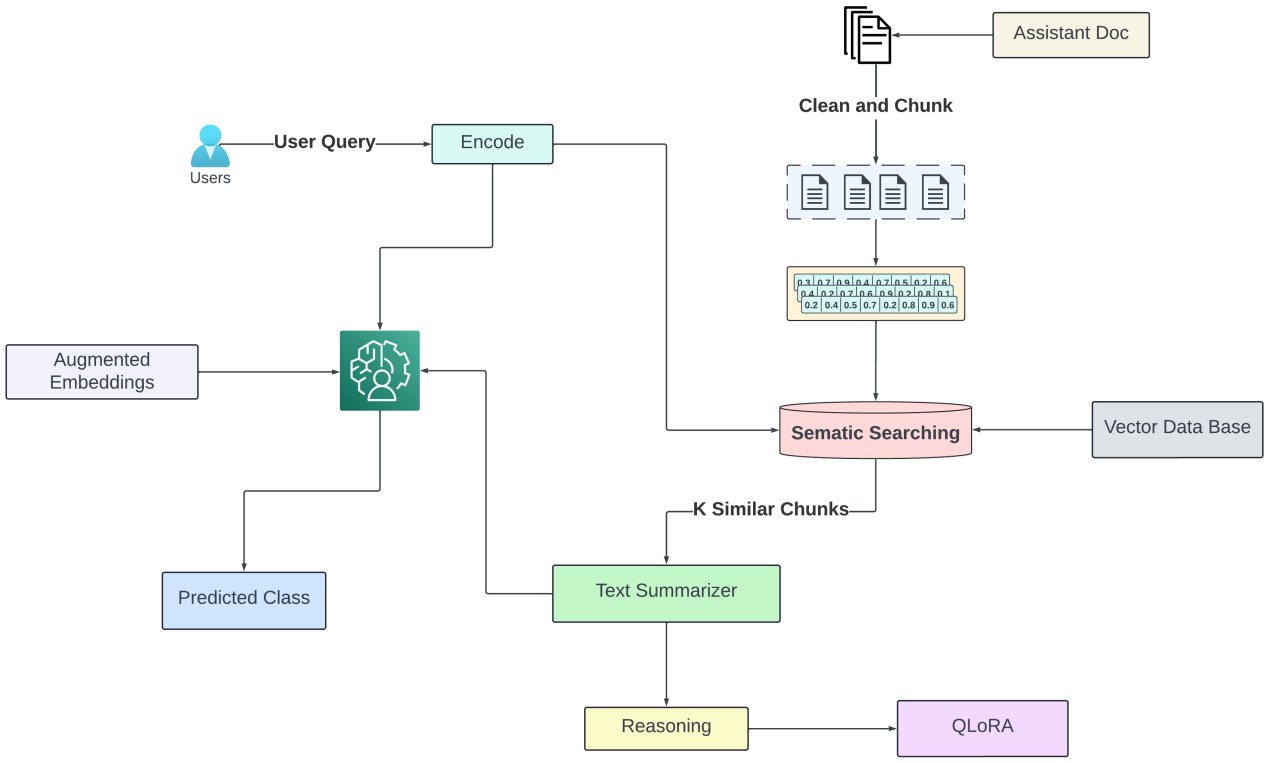

Figure 2. The WorkFlow of RAG Framework

4.3.1 QLoRA with RAG

In this project, we utilized QLoRA with LLMS. Our input format consisted of a query, expressed as 'Query: [query text]', followed by the reason, extracted from the documents based on the query, presented as 'Reason: [reason text]'. As previously discussed, employing the RAG Approach, we provided updated data to the model to enhance its understanding of the contextual information related to the query.

5 Results and Discussions

In this project, we compared the results of LLMs using QLoRA and RAG with QLoRA. We observed that RAG with QLoRA achieved better performance than QLoRA alone. This improvement can be attributed to the additional contextual information provided to the RAG model, allowing it to better understand the query context and classify the results more effectively.

Table 3
Comparison of Hate Data Results: QLoRA vs. RAG with QLoRA

Model Name	**Hate Macro F1 with QLoRA**	**Hate Macro F1 with RAG and QLoRA**
Mistral 7B	72.3	**72.8**
DeepSeek 7B	72.3	**70.9**
Zephyr Beta 7B	69.6	**70.8**
Zephyr Alpha 7B	67.1	**69.7**

6 Conclusion

In this study, we implemented a novel approach combining RAG (Retrieval-Augmented Generation) with QLoRA (Quantized Low Rank Adaptation) to enhance the performance of language models on a specific task. Initially, we employed QLoRA to fine-tune the language models using both the provided data and labels, which resulted in improved model performance. Subsequently,

Table 4
Comparison of Fake Data Results: QLoRA vs. RAG with QLoRA

Model Name	**Fake Macro F1 with QLoRA**	**Fake Macro F1 with RAG and QLoRA**
Zephyr Beta 7B	77.3	**78.2**
Zephyr Alpha 7B	74.7	**78.4**
Mistral 7B	77.3	**78.2**
DeepSeek 7B	74.7	**78.4**

leveraging RAG, we extracted relevant data from a vector database based on the queries posed. These extracted data, along with the reasons, were then passed through the fine-tuned language models, again utilizing the QLoRA approach.
The comparative analysis of the results demonstrated the effectiveness of our proposed methodology. Specifically, the integration of RAG with QLoRA yielded superior performance compared to utilizing QLoRA alone. This outcome underscores the significance of incorporating advanced techniques for both retrieval and generation tasks in natural language processing.

In conclusion, our project illustrates the effectiveness of combining RAG and QLoRA in a complementary manner, presenting encouraging developments in language model performance improvement. These results support the ongoing attempts to overcome difficulties in natural language creation and interpretation tasks by utilizing cutting-edge approaches.

References

[1] Enis Karaarslan Alan, Ahmet Yusuf and Omer Aydin. A rag-based question answering system proposal for understanding islam: Mufassirqas llm. 2024. doi: 2401.15378.

[2] Umar Farooq Umair Arshad Waseem Shahzad Ali, Raza and Mirza Omer Beg. Hate speech detection on twitter using transfer learning. 2022. doi: 10.1136/bmj.b2535.

[3] Fatimah Alkomah and Xiaogang Ma. A literature review of textual hate speech detection methods and datasets. 2022. doi: 2403.15449.

[4] Ghadi Alyahya and Abeer Aldayel. Hatred stems from ignorance! distillation of the persuasion modes in countering conversational hate speech. 2024. doi: 2403.15449.

[5] Anusha Chhabra and Dinesh Kumar Vishwakarma. A literature survey on multimodal and multilingual automatic hate speech identification. 2023. doi: 2403.15449.

[6] Mohd Fazil Vineet Kumar Sejwal Mohammed Ali Alshara Reemiah Muneer Alotaibi Ashraf Kamal Khan, Shakir and Abdul Rauf Baig. Bichat: Bilstm with deep cnn and hierarchical attention for hate speech detection. volume 151, pages 4335 – 4344, 2022. doi: 10.1136/bmj.b2535.

[7] Michael Bodnar Nikolas Schmidt Kupi, Maximilian and Carlos Eduardo Posada. dictnn: A dictionary-enhanced cnn approach for classifying hate speech on twitter. volume 151, pages 181–193, 2021. doi: 2103.08780.

[8] Reza Farahbakhsh Mozafari, Marzieh and Noel Crespi. A bert-based transfer learning approach for hate speech detection in online social media. volume 151, pages 928–940, 2020. doi: 10.1136/bmj.b2535.

[9] Aadish Sharma Nasir, Ahmad and Kokil Jaidka. Llms and finetuning: Benchmarking cross-domain performance for hate speech detection. pages 1131–1135, 2023.

[10] Lu Lu Minglu Liu Chengxuan Song Xu, Liang and Lizhen Wu. Nanjing yunjin intelligent question-answering system based on knowledge graphs and retrieval augmented generation technology. 2024.

[11] Daniel Skala Daniela Jašš Samuel Sučík Andrej Švec Šuppa, Marek and Peter Hraška. Bryndza at climateactivism 2024: Stance, target and hate event detection via retrieval-augmented gpt-4 and llama. 2024.