FIGHTING HEALTH-RELATED MISINFORMATION IN SOCIAL MEDIA WITH LARGE LANGUAGE MODELS

Moisés Robles Pagán
Electrical and Computer Engineering Department
University of Puerto Rico - Mayagüez
moises.robles@upr.edu

Manuel Rodríguez Martínez
Electrical and Computer Engineering Department
University of Puerto Rico - Mayagüez
manuel.rodriguez7@upr.edu

***Abstract*—Combating disinformation in social media is a critical problem, notably when the disinformation targets healthcare. We explore how to fine-tune Large Language Models (LLM) to counteract health-related disinformation on social media. The fine-tuned base models for this project are T5, BERT, and LlaMa-2. We divide the fine-tuning into two sections: 1) classifying if the text is health-related and 2) verifying if the text contains disinformation. To rebut disinformation we use Retrieval Augmented Generation (RAG) to query trusted medical sources. Our experiment shows that the models can classify health-related with 94% precision, 95% recall, and 90% F1. We also show that we classify disinformation texts with 99% precision, 95% recall, and 97% F1. We present an investigation that can help health experts combat and rebut disinformation on different social media platforms.**

***Index Terms*—Large Language Model, Misinformation, Transformers, Vector Databases**

I. INTRODUCTION

Nowadays, technology has advanced to the point that anyone can find any information in just a few seconds. Social media has been an essential element in the search for information. The issue with this is that anyone can find, search, share, and even write anything, accurate or not. However, this dilemma has caused problems in this modern era. If anyone can share anything, how can you be sure what is true? Users are susceptible to disinformation or misinformation. In this context, misinformation refers to messages with false information dispersed because the author misunderstood facts. In contrast, disinformation refers to messages with false information that are intentionally dispersed. The author of these messages has the intention of forming opinions based on false data. In either case, false information spreads to readers as facts. Most social media platforms recommend that users read from experts or official news outlets. Nevertheless, the overwhelming amount of data makes it complicated to keep up with everything.

Currently, social media such as X (formerly known as Twitter) have “Community Notes” which clarify tweets that are misleading or misinforming. However, this system depends totally on human interaction and is a slow and intricate process. On most occasions, when a “Community Note” is added to a tweet, the disinformation has already been spread. The issue of detecting and preventing the spread of misinformation has not been an easy task, especially in the health field. In recent times, it has been challenging for health officials to achieve the prevention of endemic or pandemics. Most of the time, these officials tend to make educational campaigns for the population. However, social media misinformation can reduce the effectiveness of these campaigns. In addition, this is harder to counteract because these can spread for longer times and reach different users [1]. Some problems these experts have faced in the past years were misinformation about vaccines, users invalidating safe measurements, and other issues.

The Twitter Health Surveillance (THS) system was designed to detect tweets related to health conditions [2]. THS is a prototype system we are building at the University of Puerto Rico, Mayagüez (UPRM). The project is designed as an integrated platform to help health officials collect tweets, determine if they are related to a medical condition, extract metadata from them, and create a warehouse that can be used to analyze the data further. The THS Artificial Intelligence (AI) components used Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) to classify tweets as being medically related, unrelated, or ambiguous. The data they used was from the Twitter API system, then processed through the Hadoop ecosystem and stored in a Hive database.

Searching for training data about this topic is not an easy task. A problem with social media text is the informality, slang terms, or special characters. However, we use the data from the THS project as our primary source for the health classification dataset. On the other hand, the misinformation dataset contains social media posts, articles, websites, and others [3]–[5]. Instead of using the THS architectures, we opted for Large Language Models (LLM).

We employ an LLM in this investigation, to detects health misinformation and provides context for its classification. Also, we did not preprocess the tweets because they could lose the context of the actual meaning when hashtags, mentions, and emojis are removed. We built the prototype using Python, PyTorch, Chroma, Ollama, and other open-source tools. In determining if a text is health-related, our system achieved a 90% F1 score and a 97% F1 if it contained misinformation. Additionally, our preliminary results show that using official health sources with Retrieval Augmented Generation (RAG) helps the LLM rebut correctly with an F1 BertScore of 82%. Hence, our system proved that it is possible to classify and rebut health-related misinformation.

A. Contributions

This paper provides the following original contributions:

- **Leveraging LLMs for Health Misinformation:** Large Language Models are being used for different fields nowadays. However, these do not focus on health misinformation on social media. We present Large Language Models as a solution to classify and rebut health misinformation texts on social media and use research papers extracted from PubMed as context for the LLM.
- **Present a novel solution to misinformation rebuttal:** To rebut misinformation, it is necessary to have an understanding of what needs to be fact-checked. Also, it is important to have the necessary context for the correction. We extracted research papers that were added to a vector database. That setup enable us to use RAG to answer health misinformation with peer-review documents.
- **Pipeline Interface:** Developed a frontend application that showcase the full pipeline, allowing users to view the process.

B. Paper Organization

This paper has the following organization. Section II contains the background on the transformer and the Large Language Models architectures, vector databases, and the Twitter Health Surveillance (THS). For section III, we can observe the system architecture for the data extraction and classification process. Later, in section IV we show the system performance. Ending with section V, we have our conclusion with suggestions for future work.

II. Background

A. Large Language Models (LLM)

The are three different architectures for Large Language Models, encoder-only, decoder-only, and encoder-decoder. Each one has advantages on specific tasks. In Figure 1

1) Encoder-only models: These models are like BERT. This type of model predict by masking specific words in a sentence. They are better for classification and sentiment analysis.

2) Decoder-only models: For these model we have the well known GPT-3. They receive one input and try to predict the entire text. They are good for summarizing and text-generation.

3) Encoder-Decoder models: T5 is an encoder-decoder models. They mask entire sequences of text. Good for translation and question and answering.

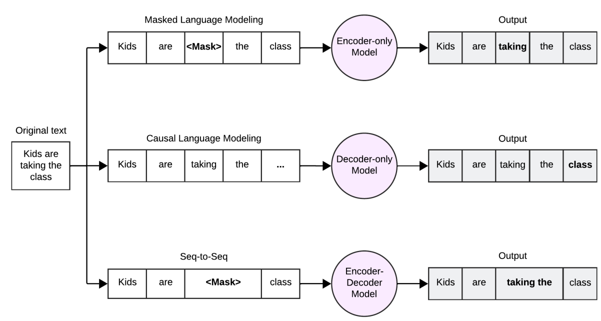

Fig. 1: The transformer architecture

III. System Architectures

This section presents the general architectures of the systems we used for the development and experiments in our research. We illustrate the Extraction, Transformation, and Load (ETL) pipeline we used for the medical research papers. Additionally, we have our misinformation classification process, where we classify the health misinformation and refute it. Finally, we have the system's User Interface (UI).

A. Research Paper ETL Pipeline

Our model must use credible sources of information to rebut misinformation. We identified PubMed [6], an online library that contains peer-reviewed medical literature. We want to extract the papers and store them in a vector database. To extract these papers, we used the BioC API [7], which has access to the PubMed library. However, the API needs the research paper's identifier, known as PubMed Central (PMC) ID. We design a scraper to extract these identifiers from the official PubMed site. The pipeline in Figure 2 shows the processes of data extraction.

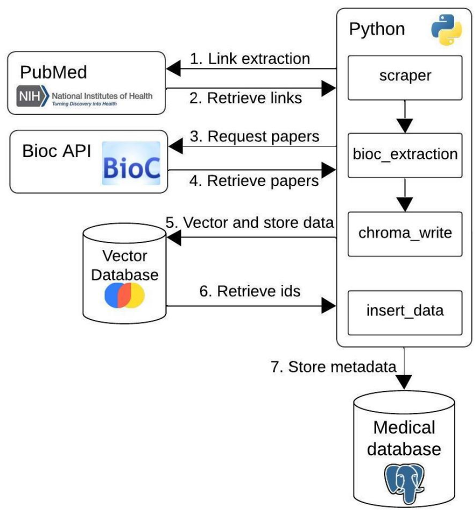

Fig. 2: Medical Data Extraction Pipeline

1) Scraper: The first step of the pipeline was identifying what papers we needed to extract. We selected topics based on the datasets we used, some topics were *allergy*, *bird flu*, *covid*, *monkeypox*, *zika*, *vaccine*, and others. To extract them, we built a scraper in Python using Selenium and BeautifulSoup libraries. We used Selenium to retrieve the web source from PubMed's website, and BeautifulSoup was used to get the links to each paper. Each link contains the PMC ID and we extracted 5,000 identifiers for each topic. These identifiers were grouped by topic and stored locally in Comma Separated Value (CSV) files.

2) BioC API: After retrieving those identifiers, we need to extract the research papers. Using the PubMed API, BioC, we made requests that returned the documents as JSON. Later, the paper's sections -introduction, methodology, results, and others- were combined as one attribute, excluding references. We removed tables, figures, and references from the context to ensure the chunking process worked appropriately. If the data is not preprocessed, when performing RAG, we can retrieve data that is not useful. After that, we stored the result into a new JSON that contains the paper's metadata and context.

3) Vectorizing data: After retrieving the data, we vectorize the papers, Figure 3 shows this process. First, each research paper's context was split into chunks using LangChain. Then, we used an LLM, BAAI [8], to embed these chunks. A universal unique identifier (UUID) was combined with each chunk and stored in a Chroma [9] database. After storing the embedding, we added these UUIDs to their JSON.

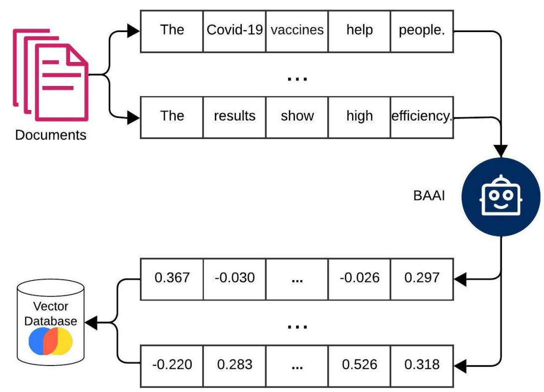

Fig. 3: Data Vectorization Process

4) Store metadata: Now, with all papers vectorized, we upload the metadata into a Postgres database. Duplicates or any research that did not contain at least an abstract were removed. That ensures that there is no repetition or inconsistency when doing the rebuttal. Later, we upload the data into the database following the schema found in Figure 4. The tables in this schema are as follow:

Research: Contains the research paper's metadata. Its attributes are: *title*, paper's title; *context*, the paper's text; *paper_ref*, the reference of the paper; and *fullpaper*, a boolean that is true if the paper contains an abstract, introduction, methodology, discussion, conclusion, and references.

Chunks: Pairs the UUIDs from the paper's chunks and their respective research record.

Keyword: Keywords that allow the reader to know the subjects mentioned in the paper.

Author: Full name of the paper's authors.

Reference: All references present in the paper.

Topic: The topics used to search the papers.

We started the search with 85,000 peer-reviewed papers. After finishing the filtering and data cleaning, we ended with 56,365 different peer-reviewed papers.

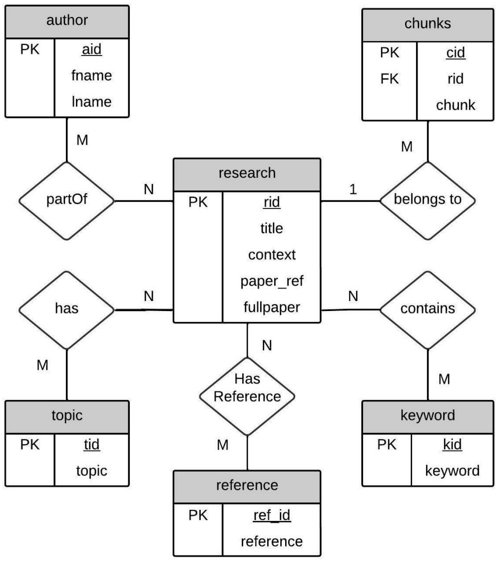

Fig. 4: Research Papers Schema Diagram

B. Misinformation Rebuttal Pipeline

After training the models and storing the context for the rebuttal, we create the model pipeline. The pipeline shown in Figure 5 shows the process of receiving a text, making the classifications, and returning an explanation of why it is misinformation.

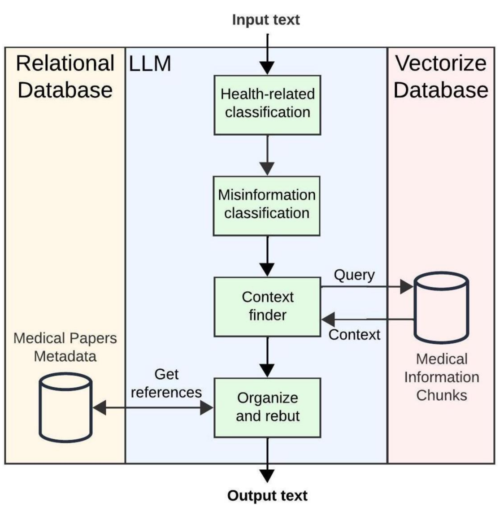

Fig. 5: Misinformation Rebuttal Pipeline

1) Health-related Classification: The first part of the pipeline is determining if the text is related to health. The model determines whether the input is related, unrelated, or ambiguous. If the text is related, we go to the next part of the pipeline. Non-related or ambiguous texts, ends the process because it is out of the model's scope.

2) Misinformation Classification: Then, we validate if the input contains misinformation. The fine-tuned model can only return if the text misinformation or non-misinformation. If the model finds no misinformation, the process is completed. When the result contains misinformation, we start the search for research papers to be used to rebut the message.

3) Context Finder: Before we the query the vector database, we must understand the topic of the text. To automate this process and generate a query as precise as possible, we use Ollama [10]. That tool allows us to execute locally a pre-trained LLM that generates text. Thus, we send a query to Ollama asking it to make a one-sentence query for the vector database related to the input.

Example
Input: "#nih fauci, expected to be grilled tomorrow over ineffective ... universal #influenza vax in 5, maybe 10 years?"
Output: "Flu vaccine effectiveness and future universal influenza vaccination strategies."

The above example shows that Ollama can identify the topics of the original text. The LLM identified the topic and made a query that can help find research papers related to it. That query is sent to the Chroma database to retrieve the research papers' chunks. For this experiment, our model returns eight chunks that are similar in context to the query. We selected this number because it returns an appropriate amount of context without Ollama truncating the text. These chunks are then sent to another model to be analyzed and organized.

4) Organize and Rebut: The final part of our pipeline is using RAG to provide an answer that explains why the original text is misinformation. First, we retrieve the references of the chunks we use for the context. Then, we send the original text with the chunks, as context, to Ollama so the model can generate an explanation that rebuts the misinformation. Ollama returns a 2-3 sentence result that explains the text's misinformation. The final output is a JSON with the classifications, rebuttal, and references used for the rebuttal.

The pipeline enables us to automate the classification process and rebut misinformation using peer-reviewed research. By leveraging fine-tuned models, vector search, and RAG, the architecture provides concise, fact-based responses. A key advantage of this approach is its ability to explain complex content accessible to non-technical readers, giving them a clearer understanding of false information. This can assist professionals in the field to mitigate the spread of lies that can negatively impact public health.

IV. Performance Evaluation

A. System Setup

1) Performance Metrics: To evaluate the models effectiveness we used the following metrics:

- **Precision**: To evaluate the proportion of positives examples that the model classified as positive that are actually positive.
- **Recall**: To evaluate the proportion of the positives examples that the model classified correctly against all positives.
- **F1**: To balance the precision and recall scores.
- **Elapsed Time**: The time needed to complete the models finetuning stage.
- **BERTScore**: Compute a similarity score between two different sentences, to verify that the rebuttal generated by the LLM relates to the classified text [11].

2) Hardware: The node's specification can be found on Table I.

TABLE I: Cluster's Node Specifications

Hardware	Description
Hard Disk	250GB
RAM	87.9GB
Processor	Intel(R) Xeon(R) Silver 4214 @ 2.20GHz
GPU	NVIDIA TESLA V100s 32GB

3) Software: Various software tools were used for the project. The training, ETL pipeline, and REST API were implemented with Python 3.9.19. To finetune the models we used PyTorch 2.0.1 with GPU support enabled for CUDA 11.7. To initialize and use the models we relied on the Transformers 4.34.0 library. We imported Bert, T5, and LLaMa-2 base models from Hugging Face (HF) [12]. To reduce the model memory usage, we use the PEFT 0.12.0 and bitsandbytes library for Low-Rank Adapter (LoRA) [13]. Also, we used Postgres 14 to store our extracted research papers. In addition, we added Chroma 0.4.24 as our vector database. Next, for the RAG process, we installed Ollama [10] and LangChain 0.1.16 to run LlaMa3.1 8B parameter model [14]. Finally, to create the UI to show the results we used React.

TABLE II: LLM Specifications

Model	HF name	Architecture	Parameters
BERT	bert-base-uncased	Encoder Only	110M
T5	t5-base	Encoder-Decoder	220M
LlaMa-2	Llama-2-7b-hf	Decoder-only	7B

B. Datasets

TABLE III: Training Datasets

Dataset	Category	Label	# records	Weights
Health Related Dataset	Unrelated	0	3828	3.25
	Related	1	7848	1.58
	Ambiguous	2	765	16.26
Misinformation Dataset	Misinfo.	0	3638	2.41
	Not Misinfo.	1	5111	1.71

1) Health-related Dataset: The health-related dataset comprises of 12,441 tweets extracted from the THS project [2]. Health professionals labeled the tweets in the dataset as related, unrelated, or ambiguous. As shown in Table III this dataset is imbalanced; thus, we applied class weights to the loss function. That prevent the models from overfitting, mitigating the effects of class imbalance during training.

2) Misinformation Dataset: The misinformation dataset, with 8,749 texts, combines data from different sources such as news, social media, and blogs [3]–[5]. Only English-only records were extracted from these sources. Labels and weights for the dataset can be found on Table III.

3) Data Preprocessing: Before we start finetuning the models, we preprocessed both datasets. To keep as much context as possible, we include links, mentions, and hashtags for the classification process. These elements are important for context because users can convey sentiments that could be relevant to text. However, these elements contain various characters, and the embedding might struggle with out-of-vocabulary or irregular patterns. Thus, we use special tokens to replace elements with patterns that do not contribute to semantic meaning, such as random strings in URLs. Any URL was replaced by *[LINK]*, mentions by *[MENTION]*, and hashtags by *[HASHTAG]*. That allows us to keep the elements and give them some importance in the text. An example of this is shown below.

Example

Input: "listening to the experts talk about #influenza at the @nmnh/@asmicrobiology #flu program like https://t.co/ehn6n"

Output: "listening to the experts talk about [HASHTAG] at the [MENTION]/[MENTION] [HASTAG] program like [LINK]"

C. Fine-tuning

The LLMs used for this paper are pre-trained base models, meaning that the model learned how words relate to each other. Training models from scratch is computationally expensive and requires a large amount of data. Instead of that, we fine-tuned the models to achieved our classification goals. We taught the models to classify two types of texts: health-related and misinformation-related texts.

The models used for this experiment are Bert, T5, and LLaMa-2, Table II shows their specifications. The architectures have their specialty; thus, we performed two processes of fine-tuning: sequence classification and CLM. BERT and LLaMa-2 were only trained on sequence classification, while T5 was trained on sequence and CLM. Because BERT architecture does not allow text generation and LLaMa-2 required more resources than the ones available they were not trained for CLM. Because of the limitations, we fine-tuned the models using LoRA.

1) Parameter-Efficient Fine-Tuning (PEFT): PEFT is a library that reduces the memory usage of a model to be fine-tuned. That library allows us to use LoRA, an adapter that reduces the number of trainable parameters of a model. The LoRA hyperparameters used are found on Table IV.

TABLE IV: LoRA Hyperparameters

Parameters	**Value**
r	16
alpha	32
dropout	0.05
bias	all

2) Training Parameters: The hyperparameters used for the training are shown in Table V.

TABLE V: Fine-tuning Hyperparameters

Parameter	**value**
Learning Rate	5E-6
Batch size	16
Epochs	20
Gradient Accumulation	8
Weight_decay	0.1
Evaluation Step	50
Evaluation Batch	2
Evaluation Accumulation	16
Warm-Up	450
Metric	f1

D. Health-Related Classification Results

We present the results of the health classification process and compare them with the best overall model of the previous THS project [2]. That work concluded that their best model is an LSTM, with no attention, and a GRU layer.

TABLE VI: Health Related Precision Result

Model	**Result**
LSTM GRU NO ATTENTION	0.83
BERT	0.85
LLaMa-2	0.94
T5 (Causal)	0.85
T5 (Sequence)	0.48

1) Precision: Table VI shows the result for the precision metric for the related classification. For clarity, we focus on this class because our project goal is to detect health-related misinformation. The best-performing model here was the LLaMa-2 model, which has a precision of 94%. The model with the lowest score was T5 (Sequence) with a 48%. All other models outperform slightly the THS model.

TABLE VII: Health Related Recall Result

Model	**Result**
LSTM GRU NO ATTENTION	0.89
BERT	0.91
LLaMa-2	0.84
T5 (Causal)	0.95
T5 (Sequence)	0.44

2) Recall: Table VII shows the result for the recall metric for the related classification. When comparing this metric with the THS investigation, there is a noticeable difference. Their results show that the LSTM layer, no attention, and a GRU layer model was the only one with a result over 80% [2]. In contrast, most of our models had a score of at least 80%. Here, our best model was T5 (Causal), with a performance of 95% in recall. Our model with the highest precision, LLaMa-2, ended with 84%.

TABLE VIII: Health Related F1 Result

Model	**Result**
LSTM GRU NO ATTENTION	0.86
BERT	0.88
LLaMa-2	0.89
T5 (Causal)	0.90
T5 (Sequence)	0.46

3) F1: Table VIII shows the result for the F1 metric for the related classification. The results show that T5 (Causal) had the highest F1 at 90%, while T5 (Sequence) the lowest at 46%. In contrast, the THS model, with an 86%, ending in second to last place.

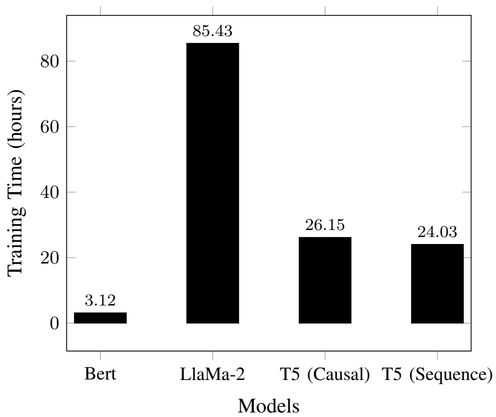

Fig. 6: Health-Related Models Training Time

4) Training Time: In Figure 6, we present our model's training time. BERT trained faster than any other model, which took 3.12 hours. Both T5 models took over 24 hours to fine-tune. Lastly, LLaMa-2 took over three days to train. Based on these results, we can infer that models with fewer parameters train faster. BERT trained 27.4x faster than LLaMa-2.

E. Misinformation Classification Results

In this section, we present the misinformation classification results. However, we did not compare with the THS model because they did not train a model to classify misinformation.

TABLE IX: Misinformation Precision Result

Model	**Result**
BERT	0.90
LLaMa-2	0.98
T5 (Causal)	0.99
T5 (Sequence)	0.99

1) Precision: Table IX shows the result for the precision metric for the misinformation classification. In this case, we focus on the misinformation class because it is our project goal. Our best-performing models were both T5 models with a precision of 99%. Nonetheless, all models had a score of at least 90%.

TABLE X: Misinformation Recall Result

Model	**Result**
BERT	0.94
LLaMa-2	0.95
T5 (Causal)	0.92
T5 (Sequence)	0.85

2) Recall: Table X shows the recall metric's results for the misinformation classification. Here, the model with the best results was LLaMa-2, with a performance of 95%. While the lowest score was T5 (Sequence) ended with 85% in the recall.

TABLE XI: Misinformation F1 Result

Model	**Result**
BERT	0.92
LLaMa-2	0.97
T5 (Causal)	0.96
T5 (Sequence)	0.92

3) F1: Table XI shows the result for the F1 metric for the misinformation classification. The results show that LLaMa-2 had the highest F1 with a 97%. However, all models had an F1 score of at least 90%.

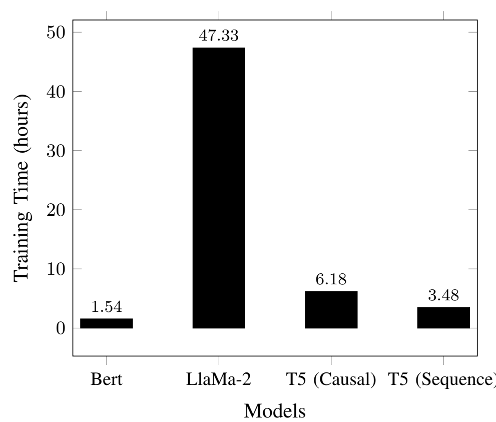

Fig. 7: Misinformation Models Training Time

4) Training Time: In Figure 7, we present the training time for the misinformation models. The model that trained the fastest was BERT, which took 1.54 hours. Next is T5 (Sequence) with 3.48 hours and T5 (Causal) with 6.18 hours. Finally, LLaMa-2 took almost 2 days to train, 30.73x slower than BERT.

F. BERTScore Result

Our model BERTScores' F1 is shown in Table XII. We calculate these values by taking the average score of the 71 texts classified as health-related and misinformation. The table shows that both models, had a 82% F1 score. That means that the generated output are closely related but not exactly the same.

TABLE XII: BERTScore F1 Results

Model	**Result**
LLaMa-3.1	82%
GPT-3.5-turbo	82%

G. Discussion

The results present that most LLMs outperformed the THS model. We applied preprocessing techniques such as replacing links, mentions, and hashtags with special tokens. The score metrics we used to evaluate the classification process were precision, recall, and F1. In our case, we want to focus on F1. In the health field, false negative should be as low as possible. A false negative is missing a misinformation post that could pose a health risk to someone. However, we do not want a high number of false positive. Overclassifying texts as misinformation is also bad, because users might get skeptical of the veracity of the system. Additionally, the process of classification is slow when rebutting, that is why we need a balance.

Our model with the best result for the health-related classification (Table VIII) was T5 (Causal), with a 90% F1 score, while the THS model had 86%. However, the trade-off for this model is that the training is computationally expensive. T5 (Causal) labels are texts instead of numbers, they must be embedded, which requires more processing power. Additionally, the training time (Figure 6) for T5 is more extensive when compared to BERT. Now, BERT had a slightly lower result with 88%. Nonetheless, when we factor in the training time and processing power, this model is more efficient.

The model with the highest F1 score for the misinformation classification (Table XI) was LLaMa-2 with 97%. However, it requires high computational power and is the model with the most extensive training time (Figure 7). However, LLaMa-2 outperform the other models by a slight margin.

For the BERTScore, our results shows that both models had an identical performance (Table XII). A possible reason is that the RAG process gives sufficient context to generate a coherent response. However, both models have their trade-offs. To use GPT-3.5-turbo, users must pay OpenAI to request their API. In contrast, LLaMa-3.1 runs with Ollama, and we need sufficient memory to run the model.

This paper focuses on social media posts, and we know that there are frequent changes in how users interact. Additionally, when new diseases are found or named, we must retrain the models to find new patterns. Retraining can be costly if the model requires excessive resources and extensive training. Thus, we can say that BERT had overall results to help combat health misinformation on social media. That model had an F1 score of 88% in health and 92% in misinformation classification, was the fastest and required the least amount of resources to train.

V. Conclusion

In this paper, we presented how LLMs can be used to refute health misinformation in social media. Additionally, we demonstrated that certain elements within a text—such as mentions, hashtags, and links—play a significant role in shaping its meaning. We also presented how we extracted, processed, stored, and used research papers with LLMs for the misinformation rebuttal. Finally, the research shows that it is possible to fine-tune large models with limited memory using LoRA. Our system was implemented with Python, Postgres, Chroma, and other open-source tools. The research presents the performance results using health-related tweets and misinformation texts from different online sources. Our research preliminary performance results show that we can achieve an F1 score of 90% for health-related classification and 97% for misinformation classification. Additionally, we present that the model can refute misinformation by generating an answer using RAG. The misinformation rebuttal models achieve an F1 BERTScore of 82%. Thus, the system can help health experts combat misinformation and reduce the risk of negatively impacting public health.

VI. Acknowledgment

This research is supported by the United States (US) National Library of Medicine of the National Institutes of Health (NIH) under award number R15LM012275. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

[1] S. T and S. Mathew, "The disaster of misinformation: A review of research in social media," *International Journal of Data Science and Analytics*, vol. 13, pp. 1–15, May 2022.

[2] C. C. Garzón-Alfonso and M. Rodríguez-Martínez, "Twitter health surveillance (ths) system," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 1647–1654.

[3] S. Crone, *Monkeypox misinformation: Twitter dataset*, 2022.

[4] Möbius, *Covid-19 fake news dataset*, Oct. 2023.

[5] S. Siwakoti, K. Yadav, I. Thange, *et al.*, *Localized misinformation in a global pandemic: Report on covid-19 narratives around the world*, Mar. 2021.

[6] *Pubmed - national library of medicine*, Accessed: 2024-05-10.

[7] D. C. Comeau, C.-H. Wei, R. Islamaj Doğan, and Z. Lu, "PMC text mining subset in BioC: about three million full-text articles and growing," *Bioinformatics*, vol. 35, no. 18, pp. 3533–3535, Jan. 2019.

[8] S. Xiao, Z. Liu, P. Zhang, and N. Muennighoff, *C-pack: Packaged resources to advance general chinese embedding*, 2023.

[9] Chroma, *The ai-native open-source embedding database*, accessed: 04.27.2024, 2022.

[10] Ollama, *Get up and running with large language models.* Accessed: 2024-09-29.

[11] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, *Bertscore: Evaluating text generation with bert*, 2020.

[12] *Hugging face – the ai community building the future*, Accessed: 2024-10-18.

[13] E. J. Hu, Y. Shen, P. Wallis, *et al.*, *Lora: Low-rank adaptation of large language models*, 2021.

[14] H. Touvron, T. Lavril, G. Izacard, *et al.*, *Llama: Open and efficient foundation language models*, 2023.

[15] D. Villanueva-Vega and M. Rodriguez-Martinez, "Finding similar tweets in health related topics," in *2021 IEEE International Conference on Digital Health (ICDH)*, 2021, pp. 184–190.