Fighting Health-Related Misinformation In Social Media With Large Language Models

Moisés Robles Pagán and Manuel Rodríguez Martínez

University of Puerto Rico - Mayagüez, Mayagüez PR 00680, USA,
moises.robles@upr.edu, 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 and use 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 divided 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 a system that can help health experts combat and rebut disinformation on different social media platforms.

1 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, 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, unable to keep up with the disinformation effort. 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 manage pandemics. Often, these officials need to make educational campaigns for the population. However, social media misinformation can reduce the effectiveness of these campaigns [1]. Some problems these experts have faced in the past years were misinformation about vaccines, and social distancing measures.

In this paper, we present enhancements to the Twitter Health Surveillance (THS) system to detect and rebut misinformation. THS is a prototype we are building at the University of Puerto Rico, Mayagüez (UPRM), designed to detect tweets related to health conditions [2]. THS is 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. THS has Artificial Intelligence (AI) components based on Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU), used to classify tweets as being medically related.

In this work, we use prior data from the THS project as our primary source for the health classification dataset. Likewise, the misinformation dataset contains social media posts, articles, websites, and others [3–5]. However, we explore Large Language Models (LLM) to detect health misinformation,

provide context for its classification decisions, and provide a rebuttal to the tweet. A key feature is the use of curated medical datasets and Retrieval Augmented Generation (RAG) to help the LLM generate a rebuttal to the misinformation. We built our new prototype components using PyTorch, Chroma, Ollama, and other open-source tools. For the task of determining if a text is health-related, our system achieved a 90% F1 score. In addition, it achieved a 97% F1 for misinformation detection. Additionally, our initial results show that querying official health sources with RAG helps the LLM rebut appropriately with an F1 BertScore of 82%. This illustrates that it is possible to use LLM to fight health-related misinformation.

1.1 Contributions

This paper provides the following original contributions:

- **Using LLMs for Health Misinformation Detection:** We present LLM as a solution to classify and rebut health misinformation texts on social media and use research papers extracted from the NIH PubMed site 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. We extracted biomedical research papers and added them to a vector database. This setup enable us to use RAG to rebut health misinformation with peer-review documents.
- **Pipeline Interface:** We developed a frontend application that showcase the full pipeline, allowing users to interact with THS.

1.2 Paper Organization

Section II contains background on LLM architectures, vector databases, and THS. In section III, we present the system architecture for the data extraction, classification process, and user interface. Later, in section IV we show the system performance. Ending with section V, we have our conclusions.

2 Background

2.1 Large Language Models

Natural language processing (NLP) has always been an intricate field because of the complexity of human language. The meaning of a message can vary because of homonyms, tone, context, and other factors that affect the message delivered. Previous NLP techniques such as Recurrent Neural Network (RNN) could help in understanding a sentence's context in the short term [6]. However, these struggle when trying to understand longer texts. In contrast, the transformer architecture [7], differs from others because it uses self-attention to understand the relationship between words and positions within a sentence. This enables the model to break ambiguities in sentences. The transformer led to the introduction of LLM [8]. These models are trained with large amounts of data to replicate human-like patterns or generate text based on statistical relationships between words.

LLM Architectures LLM have different architectures, designed for performing specific tasks:

- **Encoder-only models:** These models are like BERT [9]. This type of model predicts text by masking specific words in a sentence. They are better for classification and sentiment analysis.
- **Decoder-only models:** These models include the well known GPT-3 [10]. They receive one input and try to predict the entire text. They are good for summarization and text-generation.
- **Encoder-Decoder models:** An example is T5 [11]. These mask entire sequences of text, and are good for translation and question answering.

Challenges and Limitations An issue with LLM is that they answer based on statistical relationships between words, and occasionally, their output could make no sense. They could generate a result that is not factual or valid; this phenomenon is called an *hallucination*. Without additional context, they cannot differentiate between fact and fallacy.

2.2 Vector Databases

We can reduce the hallucinations by providing textual context relating to the input text (e.g., premise). A solution to this is finding data from official, curated sources on the topic associated with the premise text. That data can be stored in a vector database, which stores text data in a multi-dimensional vector representation. Some database engines support vector searches natively, like Chroma [12], while others have extensions, like Postgres with its pgvector module [13]. These vector databases turn images, documents, and others data into numerical vectors known as *embeddings*. These embeddings are numerical vectors capturing semantic meaning [14].

When querying the database, it will return text chunks that can function as context for an LLM to analyze. This becomes additional input that the LLM can use to generate an answer. Thus, an LLM can use the retrieved chunks to generate a fact-checked answer. The work in [15] tested LLM to answer a question that contained images and text. They evaluate the GPT3 base model, the base model with prompt engineering, and a GPT3 with RAG. Their experiments showed that the base model's average success ratio was 40%, the second model was 58%, and the model with RAG ended with 75%. Said experiment illustrates that an LLM can have high performance when it receives the necessary context.

2.3 Misinformation in Social Media

Misinformation has been dangerous during critical events like natural disasters or health crises. For instance, during the COVID-19 pandemic, false claims appeared saying that the vaccine had microchips or that it was intended for population control [16], which led to high health risks or even deaths [1] because people refused to get vaccinated out of fear.

There have been various research studies on reducing the propagation of misinformation. Misinformation was modeled as a game-theoretic problem in [17], where some players spread fake news, and others tried to stop it. They created an agent at the network level to combat misinformation in a simulation. However, they could not conclude the efficiency of their model due to the lack of discernible patterns in the simulation. On the other hand, the authors in [18] used LSTM and BERT to classify misinformation from the different news sources. They showed that BERT outperformed LSTM, achieving an accuracy of 64.88% against 60.59%.

When combatting misinformation, the difficulty arises when determining what is spreading and how experts can correct it. It requires credible sources or an expert on the field, to verify the truth. These tasks are time-consuming and the rebuttal must be expressed so that any audience can understand it. LLM can combat misinformation by classifying it and rebutting it. This AI approach can reduce the dependency on experts and have a system that can act in real-time to mitigate misinformation.

2.4 Twitter Health Surveillance (THS)

The THS system classified tweets related to health issues [2, 19]. The system utilized LSTM and GRU to classify tweets as being medical related, unrelated, or ambiguous. The THS data extraction pipeline can be found on Figure 1. THS extracted data from the Twitter API and processed it through the Apache ecosystem. Later, a preprocessing phase for each tweet occurred, which removed hashtags, mentions, emojis, and web links. The classification agent trained with the resulting plain text. This version used recurrent neural network (RNN) and 1-d convolutional neural networks (CNN) because of their advantages with sequential data. The authors tested various architectures, and the one with the highest result was an LSTM layer (with no attention) followed by a and a GRU layer. However, these experiments removed special elements from the original text, which could remove valuable insights.

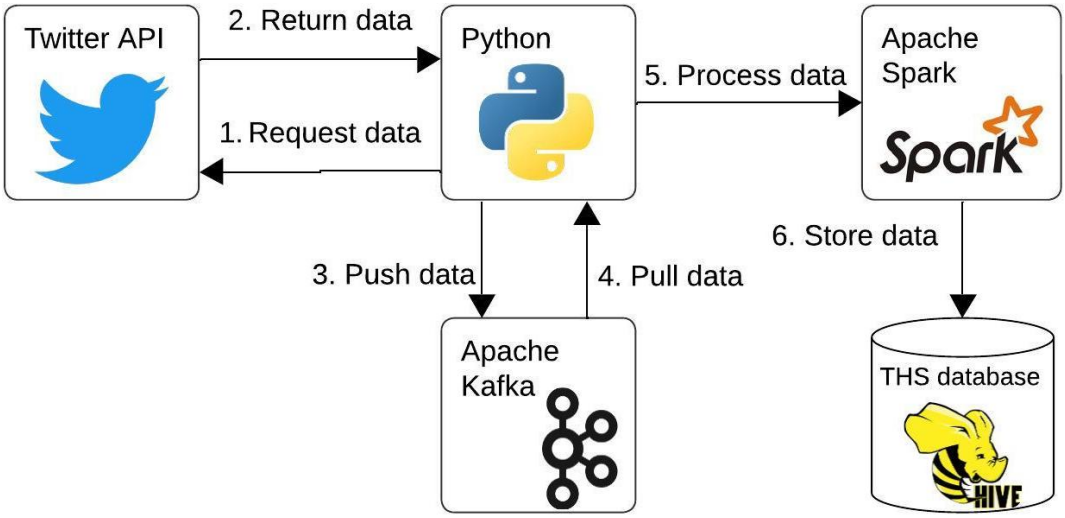

Fig. 1: THS Architecture

3 System Architecture

3.1 ETL Pipeline for Biomedical Papers

Our model must use credible sources of information to rebut misinformation. We identified NIH PubMed [20], an online library that contains peer-reviewed medical literature. We worked to extract relevant papers and store them in a vector database. To extract these papers, we used the BioC API [21], which has access to the PubMed library. The API needs the research paper's identifier, known as PubMed Central (PMC) ID. We designed a web scraper to get these identifiers. The pipeline in Figure 2 shows the processes of data extraction.

Scraper The first step of the pipeline was identifying what papers we needed to extract. We selected topics based on our datasets: *allergy*, *covid*, *monkeypox*, *zika*, *vaccine*, and others. We built our scraper using Selenium and BeautifulSoup libraries to extract the links for each paper, and get the PMC ID from the link. We extracted 5000 PMC IDs for each topic.

BioC API Using the PubMed API, BioC, we made requests with the PMC ID that returned each document as JSON text. Later, the paper's sections -introduction, methodology, results, and others- were extracted, excluding references. We removed tables, figures, and references from the context to ensure the chunking process worked appropriately. After that, we stored the result into a new JSON that contains the paper's metadata and context.

Vectorizing data Next, we vectorized the papers, as shown in Figure 3. First, each paper's context was split into chunks using LangChain. Then, we used an LLM, BAAI [22], to embed these chunks. A universal unique identifier (UUID) is combined with each chunk and stored in a Chroma database. After that, we added these UUIDs to their JSON.

Metadata Store 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. Next, we upload the metadata into a Postgres database using the following schema:

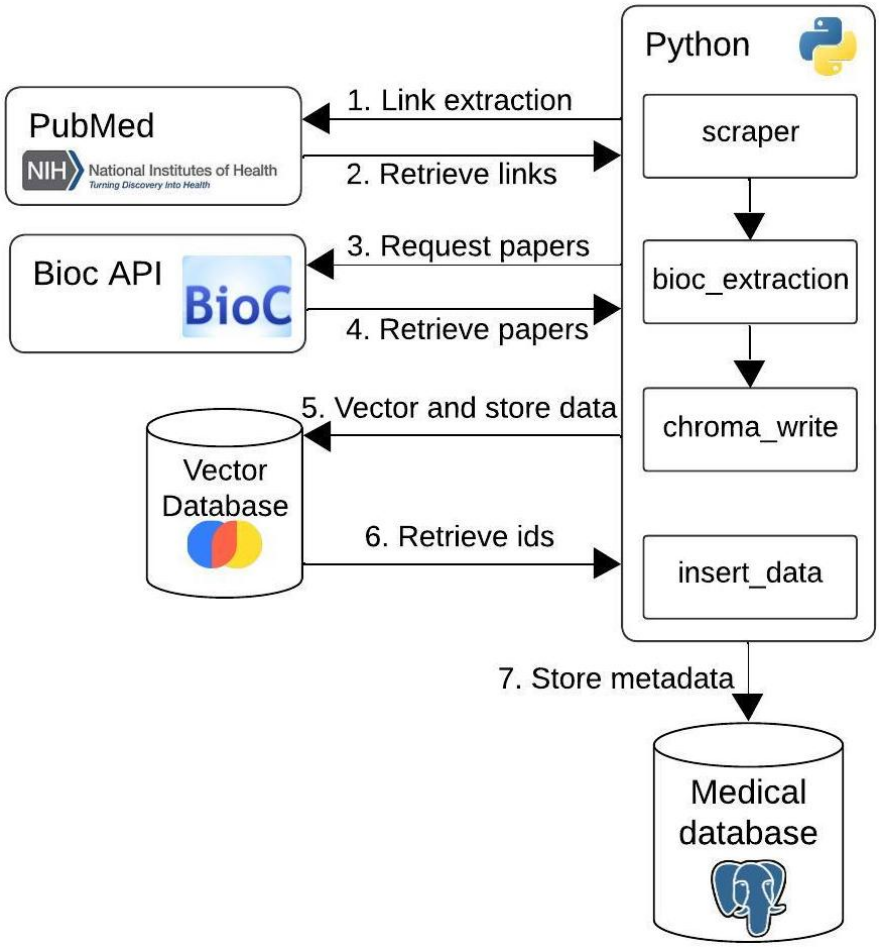

Fig. 2: Medical Data Extraction Pipeline

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 false if the paper only contains an abstract.
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.

3.2 Misinformation Rebuttal Pipeline (MRP)

After training the models and storing the context for the rebuttal, we create the model pipeline. The pipeline shown in Figure 4 shows the process of receiving a text, making the classifications, and returning an explanation of why it is misinformation.

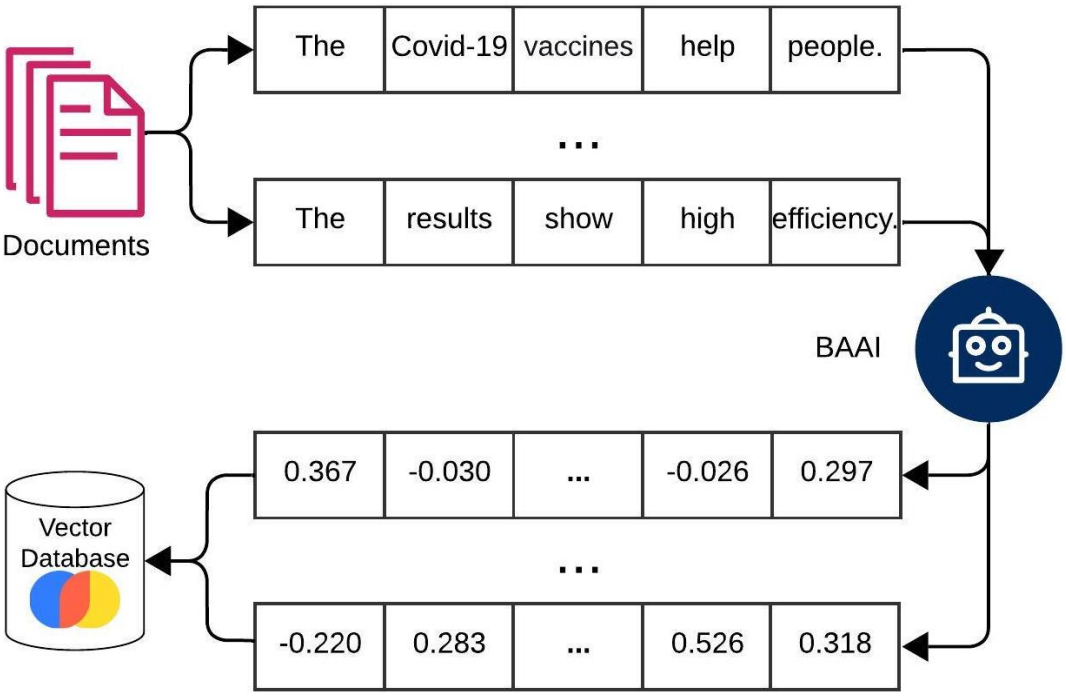

Fig. 3: Data Vectorization Process

Health-related Classification The first part of the pipeline is determining if the text is related to health. If the text is related, we go to the next part of the pipeline. Non-related or ambiguous texts end the process because it is out of the model's scope.

Misinformation Classification Next, we validate if the input contains misinformation. The model only returns if the text misinformation or not. If the model finds no misinformation, the process is completed. When the result contains misinformation, we start the search for context for the rebuttal.

Context Finder Before we 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 [23] which runs the LLama 3.1 LLM. We send a query to Ollama asking it to make a one-sentence query for the vector database related to the input tweet.

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. That output is sent to the Chroma database to retrieve the research papers' chunks. In our current setup, our model returns eight chunks. 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.

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

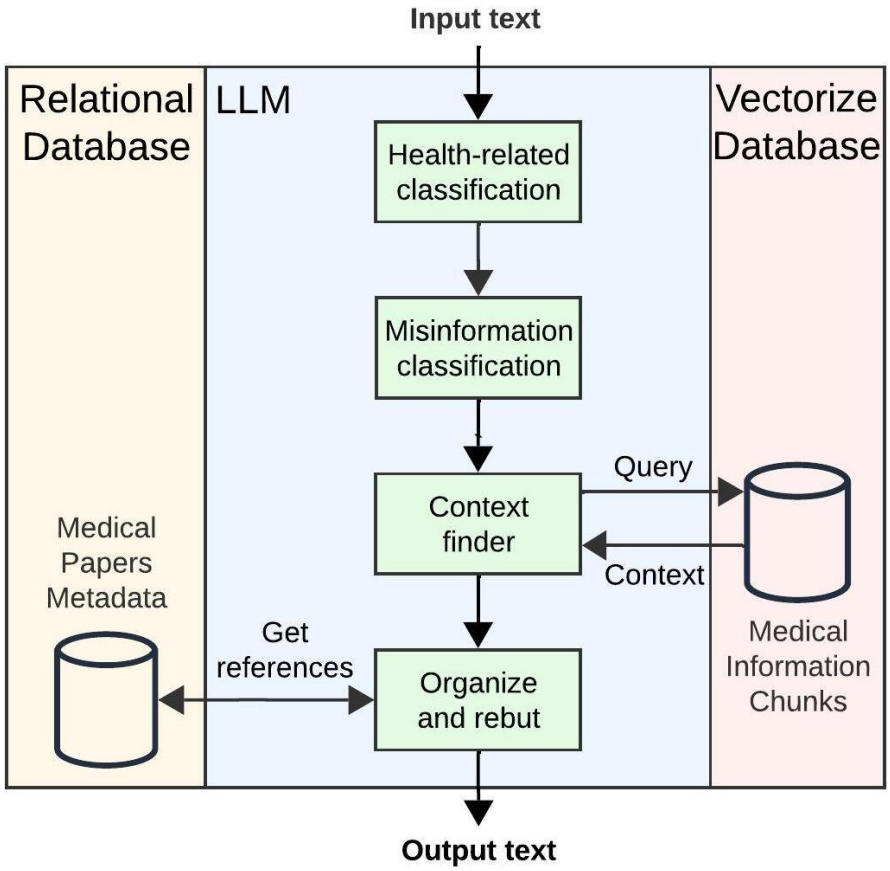

Fig. 4: Misinformation Rebuttal Pipeline

context. Then, we send the original tweet text, and the retrieved chunks as context to Ollama so the model can generate an explanation that rebuts the misinformation. The final output is a JSON with the classifications, a 2-3 sentence rebuttal generated by Ollama, and references used for the rebuttal.

The pipeline automates the classification process and rebuts misinformation using peer-reviewed research. By leveraging fine-tuned models, vector search, and RAG, the architecture provides concise, fact-based responses. Also, this approach has the ability to explain complex content in a manner accessible to non-technical readers.

3.3 User Interface (UI)

We built a UI where users can interact with tweets and see the classification/rebuttal. When we select a tweet, a screen appears showing the original text, the classifications, and rebuttal; this is shown in Figure 5. The top of the screen shows the input text, number 1 shows the health classification, and number 2 shows the misinformation classification. Below that is the rebuttal and references generated by the model.

4 Performance Evaluation

4.1 System Setup

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

Fig. 5: THS Frontend - Rebuttal View

- **Precision**: The proportion examples that the model classified as positive that are actually positive.
- **Recall**: The proportion of the positives examples that the model classified correctly against all positives.
- **F1**: Balance between the precision and recall scores.
- **Elapsed Time**: The time needed to complete the models fine-tuning stage.
- **BERTScore**: How well the rebuttal generated by the LLM relates to the classified text [24].

Hardware The system was built on a cluster's node. The specifications are: an Intel(R) Xeon(R) Silver 4214 @ 2.20GHz processor, with 87.9GB of RAM, 250GB of Hard Disk, and a NVIDIA TESLA V100s 32GB GPU.

Software 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 the models from Hugging Face (HF) [25]. To fine-tune and reduce the model memory usage, we use PEFT 0.12.0 and bitsandbytes libraries for Low-Rank Adapter (LoRA) [26]. Postgres 14 was our relational database and Chroma 0.4.24 was our vector database. Next, for the RAG process, we installed Ollama [23] and LangChain 0.1.16 to run LlaMa3.1 8B parameter model [27]. Finally, to create the UI to show the results we used React.

Table 1: 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

4.2 Datasets

Health-related Dataset The health-related dataset comprises of 12,441 tweets extracted from the THS project [2]. As shown in Table 2 this dataset is imbalanced. Thus, we applied class weights to the loss function, to prevent overfitting.

Misinformation Dataset The misinformation dataset, with 8,749 texts, combines data from different sources such as news, blogs, and others [3–5]. Details about the dataset are found on Table 2.

Table 2: Training Datasets

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

Data Preprocessing 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 the text. However, 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. The special character replacement is shown in the example 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]"

4.3 Fine-tuning

The LLMs used for this paper are pre-trained base models. 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 we used are BERT, T5, and LLaMa-2, Table 1 shows their specifications. BERT and LLaMa-2 were only trained on sequence classification, while T5 was trained on sequence and Causal Language Modeling (CLM). BERT does not allow for text generation and LLaMa-2 required more resources than the ones available, hence, they were not trained for CLM. We fine-tuned the models using LoRA, and the hyper-parameters used for LoRA and for the training are found on Table 3.

Table 3: Fine-tuning & LoRA hyperparameters

Type	Parameter	Value
LoRA	r	16
	alpha	32
	dropout	0.05
	bias	all
Fine-tuning	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

4.4 Health-Related Classification Results

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

Table 4: 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

Precision Table 4 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 LLaMa-2, with a score of 94%. Most models outperform the THS model.

Table 5: 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

Recall Table 5 shows the result for the recall metric for the related classification. The THS paper had the LSTM, no attention, and GRU model as 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 recall of 95%.

Table 6: 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

F1 Table 6 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.

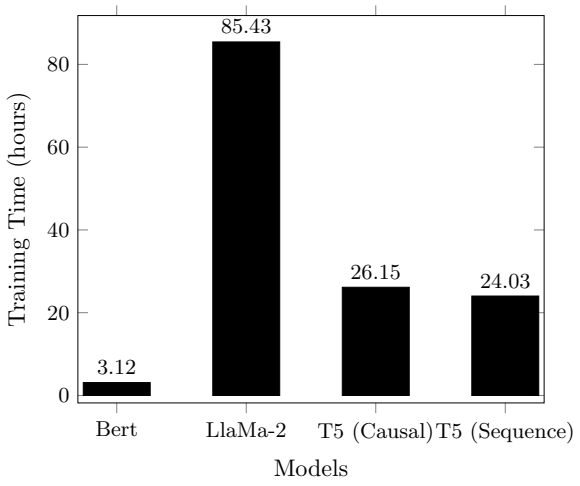

Fig. 6: Health-Related Models Training Time

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.

4.5 Misinformation Classification Results

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

Table 7: Misinformation Precision Result

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

Precision Table 7 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 8: Misinformation Recall Result

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

Recall Table 8 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%.

Table 9: Misinformation F1 Result

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

F1 Table 9 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%.

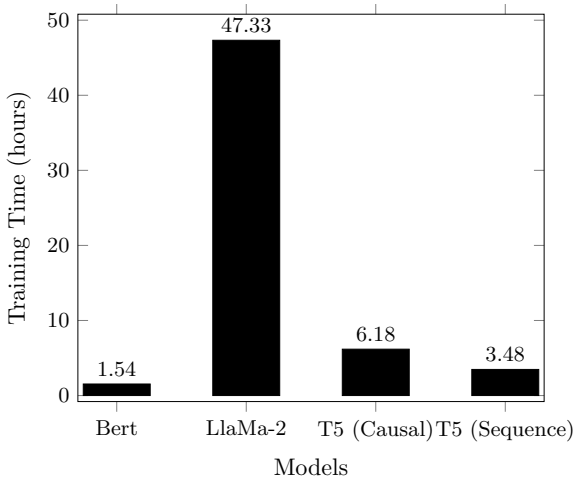

Fig. 7: Misinformation Models Training Time

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.

4.6 BERTScore Result

Our model BERTScores' F1 is shown in Table 10. 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 to the tweet.

Table 10: BERTScore F1 Results

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

4.7 Discussion

The results indicate 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. Over-classifying 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. Thus, we can see that BERT had overall good results to help combat health misinformation on social media. BERT had an F1 score of 88% in health and 92% in misinformation classification, and was the fastest to train.

5 Conclusion

We presented how LLMs can be used to refute health misinformation in social media. We also presented how we extracted, processed, stored, and used medical research papers with LLMs for the

misinformation rebuttal. The research presents the performance results using health-related tweets and misinformation texts from different online sources. Our initial 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 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.

References

1. S. T, S. Mathew, International Journal of Data Science and Analytics **13**, 1 (2022). DOI 10.1007/s41060-022-00311-6
2. C.C. Garzón-Alfonso, M. Rodríguez-Martínez, in *2018 IEEE International Conference on Big Data (Big Data)* (2018), pp. 1647–1654. DOI 10.1109/BigData.2018.8622504
3. S. Crone. Monkeypox misinformation: Twitter dataset (2022). DOI 10.34740/KAGGLE/DS/2352848. URL `https://www.kaggle.com/ds/2352848`
4. Möbius. Covid-19 fake news dataset (2023). URL `https://www.kaggle.com/datasets/arashnic/covid19-fake-news/data?select=NewsFakeCOVID-19.csv`
5. S. Siwakoti, K. Yadav, I. Thange, N. Bariletto, L. Zanotti, A. Ghoneim, J. N. Localized misinformation in a global pandemic: Report on covid-19 narratives around the world (2021). URL `https://esoc.princeton.edu/publications/localized-misinformation-global-pandemic-report-covid-19-narratives-around-world`
6. A. Sherstinsky, Physica D: Nonlinear Phenomena **404**, 132306 (2020). DOI 10.1016/j.physd.2019.132306. URL `http://dx.doi.org/10.1016/j.physd.2019.132306`
7. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin. Attention is all you need (2023). URL `https://arxiv.org/abs/1706.03762`
8. H. Naveed, A.U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, A. Mian. A comprehensive overview of large language models (2024). URL `https://arxiv.org/abs/2307.06435`
9. J. Devlin, M. Chang, K. Lee, K. Toutanova, CoRR **abs/1810.04805** (2018). URL `http://arxiv.org/abs/1810.04805`
10. T.B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D.M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, D. Amodei, CoRR **abs/2005.14165** (2020). URL `https://arxiv.org/abs/2005.14165`
11. C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P.J. Liu, Journal of Machine Learning Research **21**(140), 1 (2020). URL `http://jmlr.org/papers/v21/20-074.html`
12. Chroma. The ai-native open-source embedding database (2022). URL `https://www.trychroma.com/`. Accessed: 04.27.2024
13. pgvector. pgvector: Open-source vector similarity search for postgres (2024). URL `https://github.com/pgvector/pgvector`. Accessed: 10.1.2024
14. P.N. Singh, S. Talasila, S.V. Banakar, in *2023 IEEE International Conference on ICT in Business Industry & Government (ICTBIG)* (2023), pp. 1–7. DOI 10.1109/ICTBIG59752.2023.10455990
15. T. Sun, A. Somalwar, H. Chan, in *2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring)* (2024), pp. 1–5. DOI 10.1109/VTC2024-Spring62846.2024.10683437
16. M.S. Islam, A.H.M. Kamal, A. Kabir, D.L. Southern, S.H. Khan, S.M.M. Hasan, T. Sarkar, S. Sharmin, S. Das, T. Roy, M.G.D. Harun, A.A. Chughtai, N. Homaira, H. Seale, PLOS ONE **16**(5), 1 (2021). DOI 10.1371/journal.pone.0251605. URL `https://doi.org/10.1371/journal.pone.0251605`
17. T. Yilmaz, Ö. Ulusoy, IEEE Transactions on Computational Social Systems **10**(6), 3321 (2023). DOI 10.1109/TCSS.2022.3208793
18. A. Harbola, M. Manchanda, D. Negi, in *2023 International Conference on Innovative Data Communication Technologies and Application (ICIDCA)* (2023), pp. 1073–1077. DOI 10.1109/ICIDCA56705.2023.10100054
19. D. Villanueva-Vega, M. Rodriguez-Martinez, in *2021 IEEE International Conference on Digital Health (ICDH)* (2021), pp. 184–190. DOI 10.1109/ICDH52753.2021.00033
20. Pubmed - national library of medicine. URL `\url{https://pubmed.ncbi.nlm.nih.gov}`. Accessed: 2024-05-10

21. D.C. Comeau, C.H. Wei, R. Islamaj Doğan, Z. Lu, Bioinformatics **35**(18), 3533 (2019). DOI 10.1093/bioinformatics/btz070. URL https://doi.org/10.1093/bioinformatics/btz070
22. S. Xiao, Z. Liu, P. Zhang, N. Muennighoff. C-pack: Packaged resources to advance general chinese embedding (2023)
23. Ollama. Get up and running with large language models. URL \url{https://ollama.com}. Accessed: 2024-09-29
24. T. Zhang, V. Kishore, F. Wu, K.Q. Weinberger, Y. Artzi. Bertscore: Evaluating text generation with bert (2020). URL https://arxiv.org/abs/1904.09675
25. Hugging face – the ai community building the future. URL \url{https://huggingface.co}. Accessed: 2024-10-18
26. E.J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen. Lora: Low-rank adaptation of large language models (2021). URL https://arxiv.org/abs/2106.09685
27. H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, G. Lample. Llama: Open and efficient foundation language models (2023). URL https://arxiv.org/abs/2302.13971