**Honours Project  
Arman Doğru (300168151)  
University of Ottawa  
School of Electrical Engineering and Computer Science**

An Approach to Fake News Detection Using Machine Learning

**CSI4900  
April 23, 2024**

# **TABLE OF CONTENTS**

[**Abstract 4**](#_gyt5prbx6lih)

[**Introduction 4**](#_tj0yr47perr7)

[The Task and Its Importance 4](#_shhws7rkgeb7)

[Overview of Deep Learning and Language Models in Fake News Detection 5](#_kjyzk6i7o1vk)

[Literature Review and Identification of Gaps 5](#_pj7bg6b9v2ux)

[Motivation and Research Questions 5](#_uftg6jt1uxn)

[Problem Statement 5](#_u5cgtaqseaff)

[**Literature and Background 6**](#_yjd4vtcgxy4d)

[Detecting Fake News 6](#_3grc0wrawad)

[Transfer Learning in Fake News Detection 6](#_is3b0j2gw3z5)

[Domain-Specific Approaches in Detecting Fake News 6](#_5jeh0op00kz5)

[Large Language Models and Deep Learning Models 6](#_xa0ig58ulta)

[Figure 1: Architecture of BERT 7](#_vf5pw85v6aki)

[**Methodology and Experimental Setup 7**](#_nxfrz8gbwb9m)

[System Overview 7](#_62pseb3fob7s)

[Models Selection and Rationale 8](#_iey7qw47vpea)

[Mistral 7B (LLM) 8](#_10oxaqd1go7p)

[Figure 2: Sliding Window Attention 8](#_2cx1hfha8jae)

[Figure 3: Comparison of Mistral 7B with Llama. 8](#_y58omtvrj6bp)

[Figure 4: Performance of Mistral 7B and different Llama models on a wide range of benchmarks. 9](#_6oepqbhdkt2w)

[Fine Tuning 9](#_8k3vxk3stfop)

[Data Handling 9](#_wygbyp3v4ly)

[Datasets 9](#_9l9xbswtj6yi)

[Figure 5: Distribution of claim lengths after preprocessing 10](#_elurxb23qyr8)

[Preprocessing Steps 10](#_e22j7nddd6v5)

[Dataset Information 11](#_8out1y9kqui2)

[Categories and Vocab Size: 11](#_7c2mmyb3v3mn)

[Labels Distribution: 11](#_aukk3g9b9zfv)

[Key Observations: 11](#_8stubgz8eltu)

[Experimental Setup 11](#_rhe9o2bbf3jk)

[Mistral Hyperparameters and Training 11](#_xyzo8cjarbnm)

[Quantization and LoRA Configuration 12](#_u1b0hqqn6f8s)

[Quantization Configurations: 12](#_3d9ycibakhv5)

[LoRA Configurations: 12](#_7gnck196n1tp)

[Training Process Enhancements 13](#_a9y3uytfil88)

[Custom Training Loop: 13](#_m2zokjp8b3ei)

[Gradient Accumulation and Learning Rate Scheduler: 13](#_mvnun3vth7er)

[Cross-Validation and Data Splits 13](#_wqpf307afe7y)

[Computational Requirements 13](#_y2ho2qxz33ko)

[Model Parameters 14](#_3w5p98iqbm52)

[LSTM and GRU Models in Fake News Detection 14](#_ix8nun5cxq50)

[LSTM Models: 14](#_tqzegf4emqp)

[1. LSTM Model Architecture 14](#_gykj2lfcnogj)

[2. Bidirectional LSTM Model Architecture 15](#_6dp1udg48ya2)

[GRU Models: 16](#_suij4iax9wbl)

[Training and Implementation: 17](#_lzx2vjb3ylr4)

[Results Handling 17](#_p9d03ciotane)

[Evaluation Metrics 17](#_3o661x47mlil)

[Feature Selection 17](#_t46p5m45j0o2)

[Environmental Setup 17](#_fmkhpvcmsn2f)

[MISTRAL 7B Results 18](#_jhqaivea086c)

[Figure 6: Test Scores of Models Across Various Trainings 18](#_vsdy6th2t0v3)

[Overall Performance 19](#_3u6zk6d7w8o1)

[Domain-Specific Trainings 19](#_6br616icpasa)

[Training with Varied Data Sizes 19](#_12qwgdx276j)

[LSTM and GRU Models 19](#_msfdrfve9677)

[Figure 7: Baseline LSTM Performance Over the Epochs 20](#_a5qrtufzizoa)

[Figure 8: ML model performances across the domains 20](#_y69fflaao2y1)

[Training Performance Over Epochs 21](#_4yr7a8ajwzkd)

[Predictor Performances For Benchmark 21](#_qtt5n28ltp0d)

[Figure 9: Predictor Performances across the categories 21](#_yncfr3h08l9)

[Analysis by Category 22](#_q2p848v0ie2j)

[Discussion 22](#_key53h4iqir)

[**Failures and Lessons Learned 23**](#_g1gnbx1i0k7f)

[Lessons Learned: 23](#_v2j0os4lchkk)

[**Conclusion and Future Work 24**](#_rvl2zvw9hc6k)

[Conclusion 24](#_4ms2evkjzuqq)

[Future Work 24](#_184xq3pc2n3g)

[**References 25**](#_5dgsvjotdma2)

# Abstract

In the evolving landscape of digital information dissemination, the detection of fake news presents a significant challenge, accentuated by the rapid spread of misinformation across various media platforms. This research leverages the capabilities of the Mistral7B Large Language Model (LLM) to identify and classify fake news within distinct domains: Crime, Health, Politics, Science, and Social Media. Our approach integrates advanced machine learning techniques, specifically focusing on domain-specific adaptations of Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU), to enhance the precision and reliability of fake news detection.

Central to our methodology was the fine-tuning of the Mistral7B model across these varied domains, underpinned by a robust experimental setup that included preprocessing, model training, and rigorous performance evaluations. The project aimed to explore the efficacy of domain-specific training compared to a baseline model that processed information without fine-tuning for particular content areas. The results indicate that while the Mistral7B model exhibits high accuracy in structured domains like Health, challenges remain in more dynamic environments such as Social Media, where informal and diverse language usage prevails.

This study not only contributes to the theoretical and practical understanding of fake news detection using deep learning but also sets the stage for future work to explore more efficient computational strategies, the integration of multimodal data, and the expansion into multilingual model applications. The findings underscore the necessity of tailored approaches in the machine learning landscape to combat the nuances and complexities of fake news across different global contexts.

# Introduction

## The Task and Its Importance

In today's digital age, the dissemination of false news has become a major issue, weakening media credibility and disseminating misinformation at an alarming rate. The task at hand was to fine-tune a Large Language Model (LLM), specifically the Mistral7B, for the classification of fake news across several domains or categories, including crime, health, politics, science, and social media. This activity is critical because it seeks to improve the correctness and reliability of information distributed to the public.

## 

## Overview of Deep Learning and Language Models in Fake News Detection

Deep learning has transformed the area of artificial intelligence, providing strong methods for processing and comprehending massive information. Language models (LMs), a subset of deep learning models, excel at comprehending, producing, and interpreting human language. In the case of false news identification, LMs examine text to distinguish between authentic and counterfeit material. The use of models like as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRU) alongside LLMs demonstrates the breadth of approaches that can be used to address this issue.

## Literature Review and Identification of Gaps

While various studies have investigated the use of LMs in fake news detection, there is still a gap in domain-specific techniques. Most present systems use a generalist approach, using a single model to detect fake news regardless of subject matter. This broad approach frequently overlooks nuanced distinctions in language and presentation, which can vary dramatically between news categories.

## Motivation and Research Questions

The key objective for this research is to see if a domain-specific strategy, in which each model is tailored to a distinct news category, can outperform a baseline universal detection model that has not been fine-tuned for various domains. The study questions emphasize on the effectiveness of domain-specific training in improving model performance, as well as a comparison of LLM, LSTM, and GRU models in fake news identification.

## Problem Statement

The problem is characterized as the requirement for better accuracy and reliability in false news detection across several news domains through the use of advanced machine learning techniques and model architectures.

# 

# Literature and Background

## Detecting Fake News

Detecting fake news entails spotting deception in media content, which is crucial given the potential for serious societal consequences. Various machine learning models, such as decision trees, support vector machines, and neural networks, have been used for this purpose. Recent advances have centered on deep learning models, which are capable of comprehending complicated patterns in big datasets. LLMs, LSTMs, and GRU models stand out for their capacity to analyze sequential data, making them ideal for text analysis tasks like fake news identification. For example, Zhou and Zafarani (2018) employed LSTM models to capture the temporal features of news propagation, effectively distinguishing between true and fraudulent news.

## Transfer Learning in Fake News Detection

Transfer learning is the application of knowledge obtained in one problem domain to another, but similar problem. It has been particularly useful in detecting fake news because it can use pre-trained models to obtain higher accuracy with less datasets. A common strategy is to fine-tune models such as BERT (Bidirectional Encoder Representations from Transformers) for specific applications. For example, Lee et al. (2019) found that fine-tuning BERT on false news datasets significantly improved detection accuracy compared to models trained from scratch.

## Domain-Specific Approaches in Detecting Fake News

Domain-specific techniques customize models to specific content areas, making them more sensitive to the nuances of language used in various circumstances. This strategy, called domain adaptation, enables models to perform better on specialized tasks. Smith and Tan (2020) conducted an important study comparing the effectiveness of generic versus domain-adapted models in political and medical news, finding considerable improvements in accuracy when models were properly trained on domain-relevant data.

## Large Language Models and Deep Learning Models

Large Language Models (LLMs) such as Llama, BERT, Mistral and their variants have revolutionized many aspects of natural language processing, including fake news detection. These models are capable of understanding context and nuance in text due to their deep learning architecture. For instance, Llama V4 and BERT have been adapted for various NLP tasks with substantial success in domains requiring an understanding of complex sentence structures and contextual information.

##### Figure 1: Architecture of BERT

# Methodology and Experimental Setup

We compared the performance of LLMs (Large Language Models) with other recurrent models such as LSTMs and GRUs. The primary focus of this section is to detail the methodology used for training LLMs across various domains, including health, social media, and politics. We employed the Mistral LLM due to its robust pretrained models tailored for different domains and scenarios, aiming to understand the benefits and constraints of each model within a specific context.

## System Overview

We utilized the Mistral LLM architecture, a series of high-capacity transformer models developed for text classification and sequence modeling tasks. These models offer several advantages over traditional recurrent neural networks, including the ability to handle longer sequences with more complex patterns and dependencies, which is crucial for detailed text analysis in diverse domains like politics and health.

## Models Selection and Rationale

### Mistral 7B (LLM)

The Mistral 7B model represents a significant advancement in transformer-based architectures, setting new benchmarks in natural language processing tasks. Distinctive for its Sliding Window Attention mechanism, it achieves an impressive attention span through layered attention that compounds the range of context it can consider, enabling it to address approximately 131K tokens theoretically. In practice, this allows for twice the speed of traditional attention methods with a sequence length of 16K. Moreover, Mistral 7B incorporates a Rolling Buffer Cache, a memory-efficient approach that maintains a fixed-size cache, substantially reducing memory usage during processing.

##### Figure 2: Sliding Window Attention

Mistral 7B’s efficacy is demonstrated across various NLP benchmarks, particularly excelling in code, mathematics, and reasoning challenges, outperforming LLaMA 2 13B and competing effectively with LLaMA 1 34B. The model’s utility is further enhanced through a fine-tuning process, which adapts it to specialized tasks like disinformation detection, by adjusting its top layers to new data nuances. This combination of innovative attention mechanisms, memory optimization, and fine-tuning capabilities positions Mistral 7B at the forefront of current large language models.

##### Figure 3: Comparison of Mistral 7B with Llama.

In our investigations, we centered our attention on the Mistral7B language model for its state-of-the-art performance on a breadth of NLP benchmarks. Mistral7B's prowess in handling extensive datasets was a key factor in our selection. To gain a comprehensive view of the landscape, we included traditional deep learning neural networks such as LSTMs and GRUs in our comparisons. We approached the training in two distinct ways: For the LSTMs and GRUs, we initiated training from scratch, meticulously setting up parameters tailored to our datasets. Meanwhile, for Mistral7B, we optimized the pre-existing knowledge by fine-tuning its topmost layers, enabling it to better understand the intricacies involved in detecting disinformation.

##### Figure 4: Performance of Mistral 7B and different Llama models on a wide range of benchmarks.

## Data Handling

### Datasets

We worked with multiple datasets, each curated to include a balanced mix of real and fake news articles across three domains: health, politics, and social media.

* Crime:
  + FA-KES
  + Snopes
* Health:
  + Covid-19 Fake News Infodemic Research Dataset
  + Covid FNIR
* Politics:
  + FakeNews
  + ISOT
  + LIAR
  + Politifact
* Science:
  + Climate Dataset
* Social Media:
  + GossipCop

The datasets were preprocessed to remove outliers and balance class distribution, with the final dataset sizes ranging from 2,000 to 30,000 articles per domain. We have picked health, politics and social media for our baseline datasets.

##### Figure 5: Distribution of claim lengths after preprocessing

### Preprocessing Steps

* **Data Cleaning:** Removal of duplicate entries and rows with missing values to maintain data quality.
* **Text Processing**: All text data underwent normalization including converting to lowercase and removing punctuations. We employed tokenization using BERT’s tokenizer and removed stopwords using the NLTK corpus to refine the text data further.
* **Vectorization**: The text data was then vectorized, preparing it for input into the neural networks.
* **Outlier Detection:** Outliers were identified and removed, particularly targeting articles over 500 characters, which accounted for approximately 17% of our data.

### Dataset Information

#### Categories and Vocab Size:

* Politics
  + Total Articles: 10,000
  + Vocab Size: 16,769
* Social Media
  + Total Articles: 10,000
  + Vocab Size: 24,194
* Health
  + Total Articles: 7,474
  + Vocab Size: 10,700

#### Labels Distribution:

* True News Articles: 13,911
* Fake News Articles: 13,563

#### Key Observations:

* The dataset is fairly balanced across the true and fake labels, promoting a robust training and testing process.
* The vocabulary size indicates a rich diversity of language use across different domains, with social media showing the highest variability, followed by politics and health.

## Experimental Setup

### Mistral Hyperparameters and Training

| Domain | Batch Size | Learning Rate | Lora Dropout |
| --- | --- | --- | --- |
| Baseline | 32 | 2e-5 | 0.05 |
| Health | 32 | 1e-5 | 0.05 |
| Politics | 16 | 2e-5 | 0.05 |
| Social Media | 64 | 2e-5 | 0.05 |

We utilized a grid search approach to determine the optimal hyperparameters for each model, including learning rate, batch size, and number of epochs. Typically, models were trained over 10 epochs with early stopping implemented to prevent overfitting.

### Quantization and LoRA Configuration

#### Quantization Configurations:

4-bit Quantization:

* + The choice of 4-bit quantization helps reduce the model size by compressing the model weights to take only 4 bits per weight instead of the traditional 32-bit floating point. This drastically reduces the memory footprint.
  + Type ('nf4'): This stands for Noise-Free 4-bit quantization which is designed to be near lossless. This means it aims to retain the precision of the model while reducing its size.
  + Double Quantization: This involves quantizing already quantized weights further, potentially reducing the model size more aggressively without a significant accuracy drop. It helps in fine-tuning the balance between model size and performance.
  + Computation dtype as torch.bfloat16: Using bfloat16 allows for maintaining the dynamic range of the operations similar to float32 but at half the precision, reducing computational demand and enhancing speed, especially on supported hardware.

These settings contribute to reducing model size and computational overhead, making the model more efficient during both training and inference, while attempting to preserve as much of the original model's accuracy as possible.

#### LoRA Configurations:

1. Dimensionality Reduction (r=16):
   * This refers to the rank of the low-rank matrices in LoRA (Low-Rank Adaptation), which adjust the model's pre-trained weights. Lower ranks like 16 reduce the number of parameters significantly, hence reducing complexity and computational load.
2. Scaling Factor (lora\_alpha=8):
   * The scaling factor controls the extent to which the LoRA layers can adapt the pre-existing model weights. A scaling factor of 8 strikes a balance between allowing enough flexibility for adaptation and not deviating too much from the pre-trained weights.
3. Dropout Rate:
   * Setting a dropout rate (e.g., 0.05 or 5%) helps in preventing overfitting by randomly dropping units during the training process, which aids in generalizing better on unseen data.

These LoRA settings are designed to finely tune the balance between performance and overfitting by controlling how much the model adapts from pre-trained states during fine-tuning.

### Training Process Enhancements

#### Custom Training Loop:

* Class Weights in Binary Classification:
  + The custom trainer class handles class imbalance by using class weights in the loss computation. It uses binary cross-entropy with logits as the loss function, which is appropriate for binary classification tasks. This helps in emphasizing the minority class during training, thus addressing class imbalance effectively.

### Gradient Accumulation and Learning Rate Scheduler:

* Gradient Accumulation:
  + By accumulating gradients over multiple steps, the model manages GPU memory better, allowing for larger batches or maintaining batch size on limited memory without compromising on the ability to process larger datasets.
* Learning Rate Scheduler:
  + A learning rate scheduler adjusts the learning rate during training, usually lowering it as training progresses. This helps in fine-tuning the model more effectively as it converges, reducing the learning rate to prevent overshooting minima in the loss landscape.

These techniques enhance the efficiency of the training process and contribute to building a robust model by managing resources effectively and adapting the learning process according to the training dynamics.

### Cross-Validation and Data Splits

We employed a stratified k-fold cross-validation technique to ensure that each fold was representative of the overall dataset. Data splits for training, validation, and testing were 70%, 20%, and 10%, respectively.

### Computational Requirements

The platform we have used to run our training scripts was Kaggle and Google Colab. Given the extensive size of the datasets, training was conducted on high-performance GPU servers equipped with two T4 GPUs with 15GB VRAM each. This setup was crucial for managing the computational load and ensuring efficient processing times.

### Model Parameters

| **Model** | **Number of Layers** | **Hidden Layers Size** | **Attention Heads** | **Trained Parameters** |
| --- | --- | --- | --- | --- |
| MISTRAL 7B | 24 | 1024 | 16 | 7 billion |

## LSTM and GRU Models in Fake News Detection

### LSTM Models:

Long Short-Term Memory (LSTM) models are a type of recurrent neural network (RNN) designed to handle sequence prediction problems with input data that has time-dependent structures. This capability makes LSTMs ideal for natural language processing tasks, including fake news detection. Unlike standard feedforward neural networks, LSTMs have feedback connections that allow them to process not just single data points, but entire sequences of data. For our project, LSTM models were utilized to analyze textual data through the layers of networks to capture the temporal dependencies within the text that could indicate the presence of fake news.

We implemented a simple LSTM model where the network architecture consisted of an embedding layer followed by an LSTM layer and a dropout layer to prevent overfitting. This was followed by a dense layer with a sigmoid activation function to classify the input as real or fake news. The model's effectiveness hinges on its ability to learn long-term dependencies in text data, such as the context surrounding specific claims or statements within an article, which can be critical in distinguishing between real and fake information.

### 1. LSTM Model Architecture

The LSTM model is designed for sequence prediction problems and is especially suited for text because it can capture long-term dependencies in text sequences that are crucial for understanding context in language. Here's the architecture breakdown:

* Embedding Layer: Transforms the input sequence into a dense vector of fixed size. Here, the vocabulary size is set to 10,000 words, and each word is represented by a 40-dimensional vector.
* Dropout Layer: Applied after the embedding layer to prevent overfitting by randomly setting the fraction (30% here) of input units to 0 at each update during training time.
* LSTM Layer: Processes the sequence input by traversing the text with 100 LSTM units. This layer captures dependencies in text data over long distances effectively.
* Dropout Layer: Again, a dropout of 30% is used after the LSTM layer to reduce overfitting.
* Dense Layer: A fully connected layer that outputs the prediction probability of the input being real or fake news. The sigmoid activation function is used to compress the output between 0 and 1.

| **Layer (type)** | **Output Shape** | **Param #** |
| --- | --- | --- |
| Embedding | (None, 5000, 40) | 400,000 |
| Dropout | (None, 5000, 40) | 0 |
| LSTM | (None, 100) | 56,400 |
| Dropout | (None, 100) | 0 |
| Dense | (None, 1) | 101 |

### 2. Bidirectional LSTM Model Architecture

The Bidirectional LSTM extends the standard LSTM by processing the sequence data in both forward and backward directions. This is particularly useful in text processing where the context from both the past and future of a given point is crucial for effective understanding.

* Embedding Layer: Similar to the LSTM model, converting words to vectors.
* Bidirectional LSTM Layer: A wrapper around the LSTM layer that runs two LSTMs: one in the forward direction of the input sequence and another in the reverse direction. This layer doubles the output dimension.
* Dropout Layer: Helps in preventing overfitting by ignoring randomly selected neurons during training, which helps in reducing the chance of creating model dependencies.
* Dense Layer: Final prediction output as in the LSTM model.

| **Layer (type)** | **Output Shape** | **Param #** |
| --- | --- | --- |
| Embedding | (None, 5000, 40) | 400,000 |
| Bidirectional | (None, 200) | 112,800 |
| Dropout | (None, 200) | 0 |
| Dense | (None, 1) | 201 |

### GRU Models:

Gated Recurrent Units (GRU) are another form of RNN similar to LSTMs but with a simpler structural design, which often allows them to be more efficient computationally. Like LSTMs, GRUs are designed to help overcome the vanishing gradient problem in traditional RNNs, making them effective for tasks requiring learning from large sequences of data. GRUs achieve this by using gating mechanisms that regulate the flow of information. These gates are capable of learning which data in a sequence is important to keep or discard, which optimizes the model's ability to capture essential information for making accurate predictions.

* Embedding Layer: Maps each token to a 40-dimensional vector, same as LSTM.
* GRU Layer (100 units, return sequences): GRU processes the input while keeping the sequential integrity, allowing stacking another GRU.
* Dropout (0.5): Helps mitigate overfitting by ignoring nodes randomly.
* Second GRU Layer (64 units): Further processes the information without returning sequences to prepare for the output.
* Dropout (0.5): Additional dropout layer before the output.
* Dense Layer (ReLU activation, 64 units): Introduces non-linearity to the model aiding complex patterns recognition.
* Output Dense Layer (sigmoid activation, 1 unit): Final prediction of fake or real news.

| **Layer (type)** | **Output Shape** | **Param #** |
| --- | --- | --- |
| Embedding | (None, 5000, 40) | 400,000 |
| GRU | (None, 200) | 112,800 |
| Dropout | (None, 200) | 0 |
| GRU | (None, 200) | 112,800 |
| Dropout | (None, 200) | 0 |
| Dense | (None, 1) | 201 |

### Training and Implementation:

Both LSTM and GRU models were trained from scratch using our preprocessed datasets, which included a diverse range of news articles categorized into health, politics, and social media domains. The training process involved several steps to optimize model performance, such as vectorization of text using TF-IDF encoding, sequence padding, and embedding to transform textual data into a format suitable for neural network processing.

The models were evaluated based on their accuracy, precision, recall, and F1-score metrics, with each model undergoing a rigorous training regimen involving multiple epochs and batch processing to ensure robust learning. The LSTM model was particularly noted for its high precision, whereas the GRU model excelled in handling bidirectional context, which was reflected in its slightly better recall rates.

### **Results Handling**

#### **Evaluation Metrics**

For each model and dataset, we tracked precision, recall, accuracy, F-score, G-mean. The results for each epoch were recorded in Weights and Biases then ported to an Excel sheet, providing a detailed account of the model's performance over time.

#### **Feature Selection**

Features were selected based on their frequency and relevance to the fake news detection task. The MISTRAL7B model was particularly effective in utilizing its vast number of parameters to discern relevant features autonomously.

### **Environmental Setup**

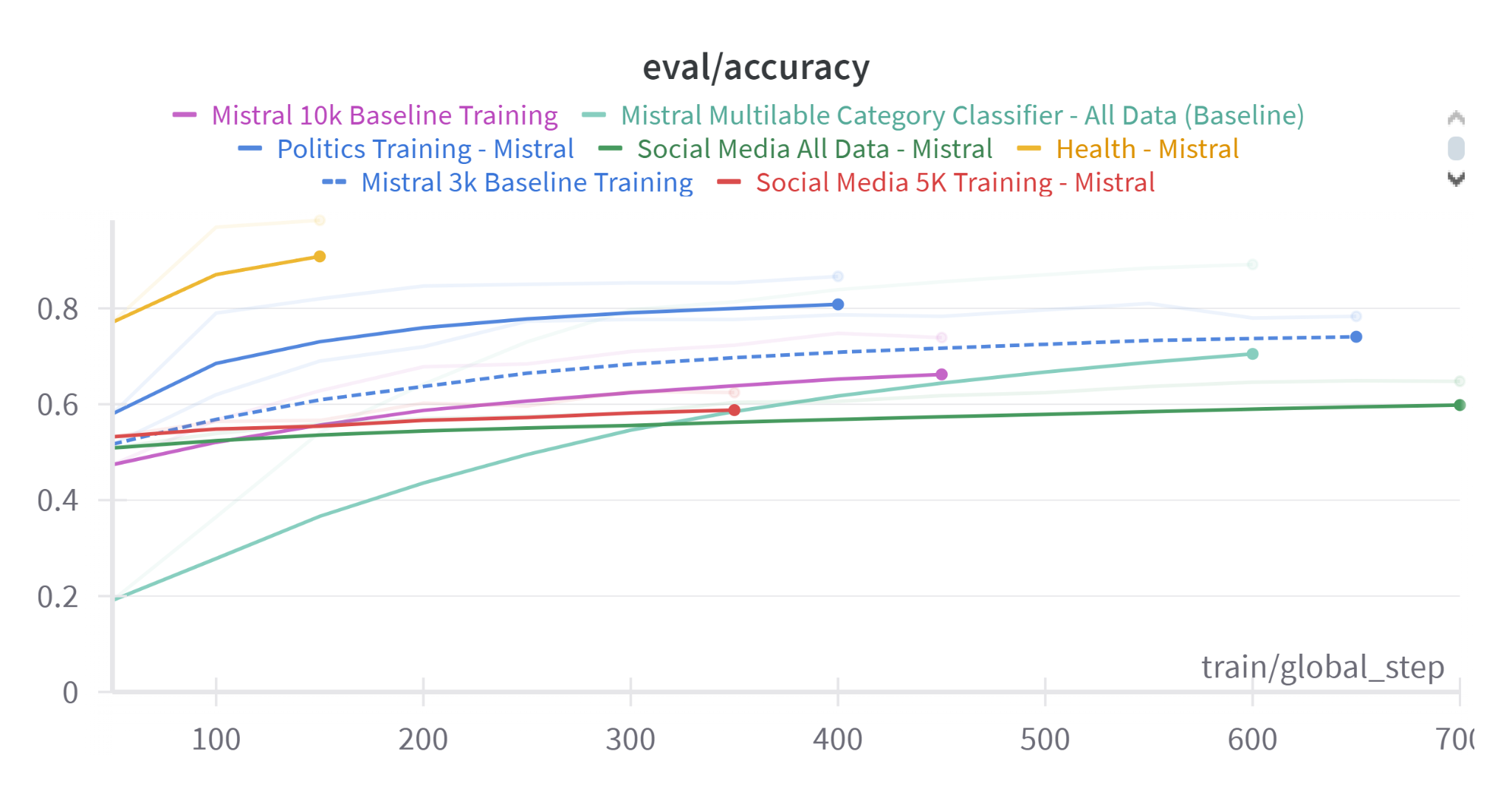
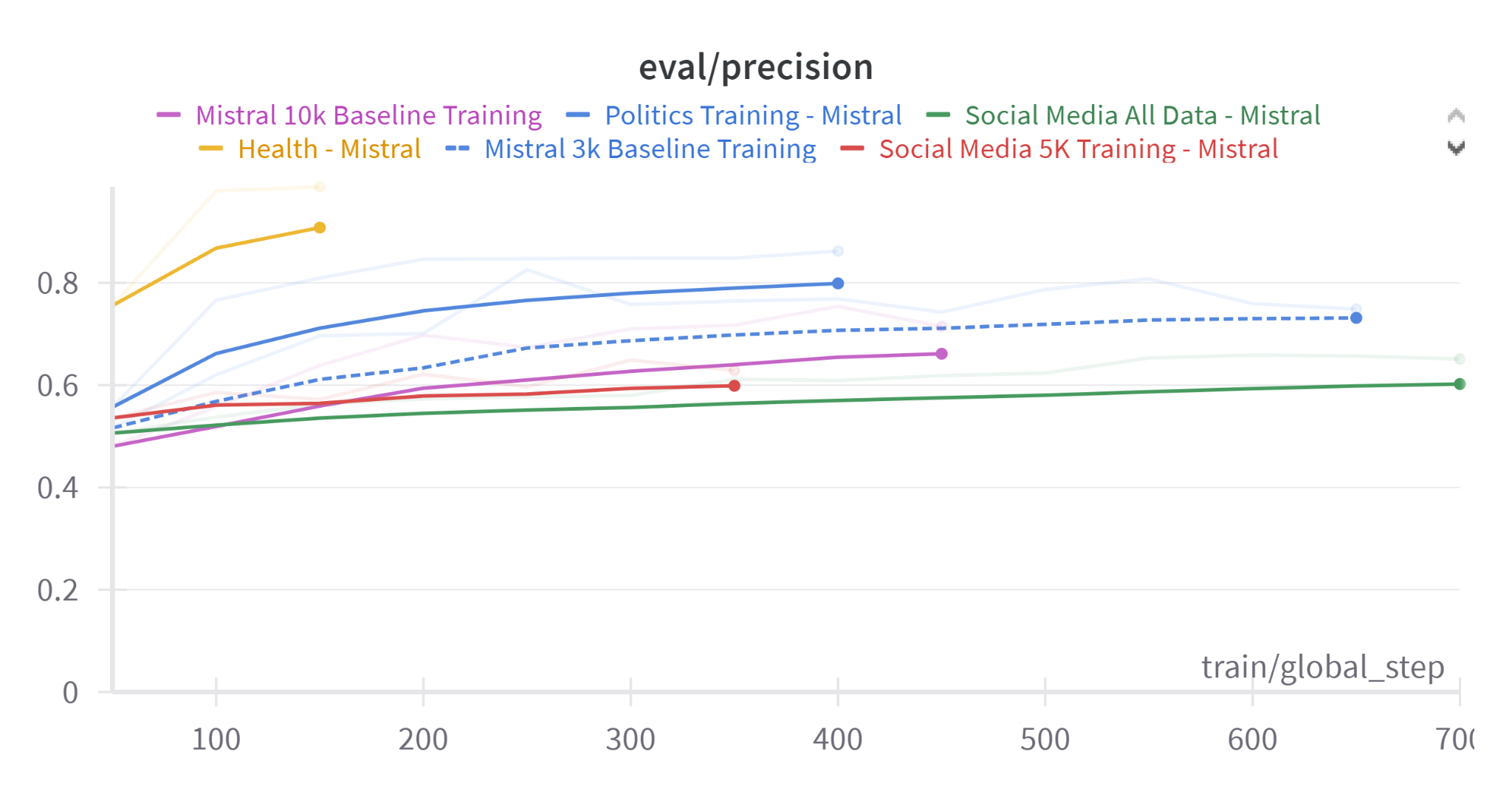
Our experiments were conducted on a Kaggle server setup consisting of:

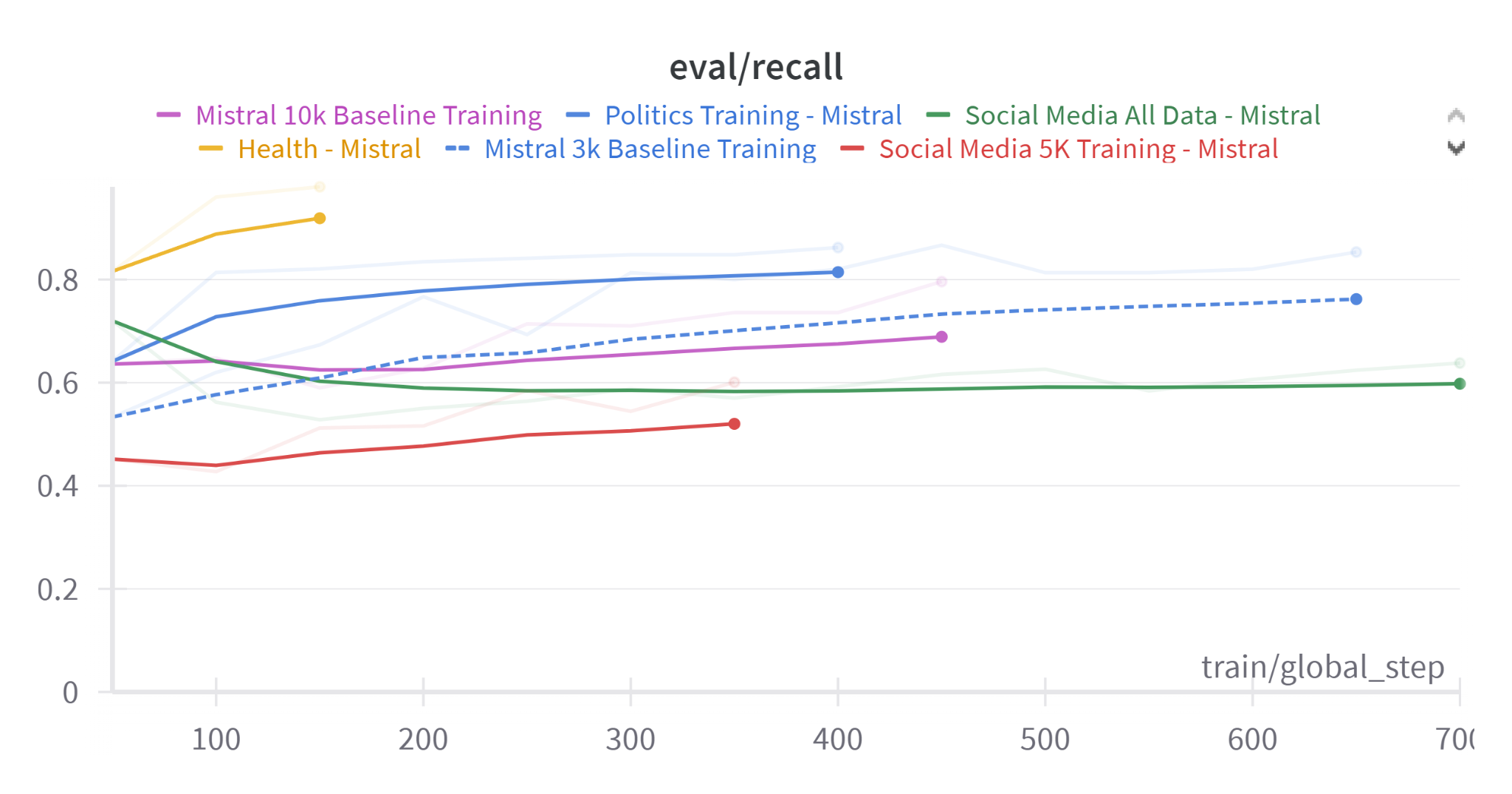
* GPU: NVIDIA T4 (x2)
* VRAM: 15GB (x2)
* Storage: 50GB

## 

## MISTRAL 7B Results

We have experimented with different data samples to see how the model would perform on different sizes of data, this allowed us to make discoveries about our datasets.





##### Figure 6: Test Scores of Models Across Various Trainings

| **Model** | **Test Accuracy** | **Test Recall** | **Test Precision** |
| --- | --- | --- | --- |
| Mistral Multilabel Category Classifier - All Data (Baseline) | 0.855 |  |  |
| Mistral 10k Baseline Training | 0.628 | 0.648 | 0.717 |
| Mistral 3k Baseline Training | 0.773 | 0.693 | 0.825 |
| Social Media All Data - Mistral | 0.649 | 0.866 | 0.807 |
| Social Media 5K Training - Mistral | 0.626 | 0.584 | 0.628 |
| Politics Training - Mistral | 0.866 | 0.862 | 0.862 |
| Health - Mistral | 0.983 | 0.98 | 0.987 |

#### Overall Performance

The baseline performance of the Mistral 3,000 and 10,000 data points respectively demonstrates that general purpose fakenews detection using LLMs lead up to a really big vocabulary size hence a really inaccurate predictions.

#### Domain-Specific Trainings

1. Health - Mistral: Demonstrated exceptional accuracy of 98.3%, highlighting the model's strength in processing and understanding structured, technical content.
2. Politics Training - Mistral: Achieved robust accuracy at 86.6%, reflecting the model's capability to handle nuanced political discourse.
3. Social Media All Data - Mistral: Showed lower accuracy at 64.9%, indicative of the challenges posed by the informal and diverse language used in social media platforms. Indicating that this dataset might be the reason behind the low accuracy on mistral

#### Training with Varied Data Sizes

* Mistral 10k Baseline Training: Showed a moderate accuracy of 62.8%, suggesting challenges in generalization with more data.
* Mistral 3k Baseline Training: Resulted in better performance with an accuracy of 77.3%, possibly due to more focused learning on a smaller, possibly less varied dataset.

### **LSTM and GRU Models**

##### Figure 7: Baseline LSTM Performance Over the Epochs

| **Domain** | **Model** | **Test Accuracy** | **Test Precision** | **Test Recall** | **Test Loss** |
| --- | --- | --- | --- | --- | --- |
| Politics | LSTM | 90.03% | 91.37% | 90.01% | 0.3069 |
|  | GRU | 90.06% | 94.51% | 85.57% | 0.2670 |
|  | LSTM\_B | 88.09% | 92.13% | 89.18% | 0.3168 |
| Social | LSTM | 76.70% | 76.89% | 79.09% | 0.4825 |
|  | GRU | 77.09% | 78.62% | 75.33% | 0.4859 |
|  | LSTM\_B | 77.52% | 76.62% | 73.42% | 0.4930 |
| Health | LSTM | 93.35% | 91.49% | 96.40% | 0.1835 |
|  | GRU | 93.84% | 94.31% | 94.02% | 0.1842 |
|  | LSTM\_B | 92.62% | 92.25% | 93.95% | 0.2082 |

##### Figure 8: ML model performances across the domains

#### Training Performance Over Epochs

Generally, both models showed improvements in accuracy, precision, and recall as the number of epochs increased, confirming their ability to learn effectively over time.

**Politics:**

* + LSTM and GRU models approached a 90% accuracy threshold, with GRU slightly outperforming LSTM in precision.

**Social Media:**

* + Both models struggled compared to other domains, yet they showed gradual improvements, indicating an issue with the dataset.

**Health:**

* + Accuracy nearing or exceeding 93%, underscoring the models' effectiveness in structured data environments.

### Predictor Performances For Benchmark

| **Category** | **Model** | **Precision (False)** | **Recall (False)** | **F1-Score (False)** | **Precision (True)** | **Recall (True)** | **F1-Score (True)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Health** | **GaussianNB** | 0.64 | 0.83 | 0.73 | 0.77 | 0.55 | 0.64 |
|  | **BernoulliNB** | 0.64 | 0.93 | 0.76 | 0.88 | 0.49 | 0.63 |
|  | **MLP** | 0.78 | 0.78 | 0.78 | 0.79 | 0.78 | 0.78 |
|  | **RFC** | 0.79 | 0.87 | 0.83 | 0.86 | 0.78 | 0.82 |
| **Politics** | **GaussianNB** | 0.64 | 0.83 | 0.73 | 0.77 | 0.55 | 0.64 |
|  | **BernoulliNB** | 0.64 | 0.93 | 0.76 | 0.88 | 0.49 | 0.63 |
|  | **MLP** | 0.77 | 0.79 | 0.78 | 0.79 | 0.77 | 0.78 |
|  | **RFC** | 0.79 | 0.87 | 0.83 | 0.86 | 0.77 | 0.81 |
| **Social** | **GaussianNB** | 0.64 | 0.83 | 0.73 | 0.77 | 0.55 | 0.64 |
|  | **BernoulliNB** | 0.64 | 0.93 | 0.76 | 0.88 | 0.49 | 0.63 |
|  | **MLP** | 0.77 | 0.78 | 0.78 | 0.78 | 0.78 | 0.78 |
|  | **RFC** | 0.79 | 0.87 | 0.83 | 0.86 | 0.78 | 0.82 |

##### Figure 9: Predictor Performances across the categories

#### Analysis by Category

The detailed breakdown of predictor performances across the Health, Politics, and Social categories revealed significant variances:

**Health:**

* + Best performance with accuracies ranging from 69% to 82%, reflecting strong adaptability to structured, factual data.

**Politics:**

* + Slightly lower performance than health but still robust, with accuracies reaching up to 82%.

**Social:**

* + The most challenging category with accuracies not exceeding 83%, highlighting the complexity of processing informal and varied language.

### Discussion

The MISTRAL7B model's performances have provided substantial insights into the complexities of text classification across varied domains. The excellent results in the health sector can be attributed to the alignment of technical language with the model's capabilities in parsing and understanding detailed, factual information. The moderate outcomes in the politics domain suggest a need for models to better grasp the subtleties and variability in political language, whereas the relatively lower performance in the social media domain underscores the ongoing challenges in dealing with casual, diverse language usages.

The overall varying accuracies are mostly being caused by having completely different vocabularies and claim structures for each dataset. For instance social media dataset contains shorter claims with much more varied vocabulary whereas the health dataset contains similar claim structures and smaller vocabulary size. We believe that there is much more to test and see for future work, using more computational power and larger datasets, we could understand a lot more about how LLMs perform with different domains.

# 

# 

# 

# Failures and Lessons Learned

**Computational Challenges**: One of the significant hurdles we encountered was managing the computational demands of training large language models like Mistral7B. The high memory and processing power required often led to long training times and difficulty in optimizing model parameters efficiently.

**Downsampling of Datasets:** Due to these computational constraints, we were compelled to downsample our datasets. While this made it feasible to complete the training cycles within reasonable time frames, it potentially compromised the diversity and representativeness of our training data, which may have affected the model's ability to generalize across real-world scenarios.

**Model Performance Variances:** Throughout our experiments, we observed that while the Mistral7B model excelled in structured domains like Health, its performance was less satisfactory in more dynamic and varied contexts such as Social Media. This disparity highlighted the limitations of our current approach in handling informal and diverse language patterns effectively.

#### Lessons Learned:

1. **Resource Allocation**: Ensuring adequate computational resources and optimizing data handling can significantly enhance training efficiency and model performance.
2. **Balanced Data Handling:** It's crucial to maintain a balance between manageable dataset sizes and maintaining the quality and diversity of the training data.
3. **Model Adaptability:** Developing strategies for better domain adaptation can help improve model robustness, especially in less structured environments.

# 

# 

# Conclusion and Future Work

## Conclusion

Our research with the Mistral7B model on detecting fake news across various domains has yielded promising results, particularly in structured categories like Health, where the model's deep learning capabilities effectively parsed and understood complex datasets. However, the challenges encountered in more informal domains such as Social Media underscore the need for further refinement of our approaches.

## Future Work

1. **Enhanced Computational Strategies**: Exploring more efficient computational strategies, such as distributed computing or advanced data sampling techniques, to manage the high demands of training large models.
2. **Domain-Specific Model Tuning:** Further research into domain-specific tuning and training methods could help improve performance across challenging areas like Social Media.
3. **Integration of Multimodal Data:** Incorporating multimodal data sources, such as images and videos, could enhance the model's accuracy and applicability in real-world scenarios where fake news often combines text with visual content.
4. **Expansion to Multilingual Models**: Extending the models to handle multiple languages could vastly increase their utility in global contexts, where fake news is not limited to English-speaking populations.
5. **Exploring New Model Architectures:** Investigating emerging model architectures that may offer better handling of diverse and informal text data could lead to significant improvements in fake news detection.

This research has laid a solid foundation for future explorations and highlighted critical areas for improvement that, if addressed, could substantially enhance the reliability and accuracy of fake news detection systems globally.

# References

*The architecture of bert*. Transformers for Natural Language Processing. (n.d.-a). https://subscription.packtpub.com/book/data/9781800565791/2/ch02lvl1sec09/the-architecture-of-bert

*The architecture of bert*. Transformers for Natural Language Processing. (n.d.-b). https://subscription.packtpub.com/book/data/9781800565791/2/ch02lvl1sec09/the-architecture-of-bert

Hauzenberger, L. (2024, January 16). *Multilabel classification using mistral-7b on a single GPU with quantization and Lora*. Medium. https://medium.com/@lukas.hauzenberger/multilabel-classification-using-mistral-7b-on-a-single-gpu-with-quantization-and-lora-8f848b5237f3

Huggingface. (n.d.). *Blog/lora-for-sequence-classification-with-roberta-llama-mistral.md at main · Huggingface/blog*. GitHub. https://github.com/huggingface/blog/blob/main/Lora-for-sequence-classification-with-Roberta-Llama-Mistral.md

Kinsella, B. (2024, February 27). *Mistral introduces two new high performance llms and neither are open-source*. Synthedia. https://synthedia.substack.com/p/mistral-introduces-two-new-high-performance